ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Ένα Σημασιολογικό Πλαίσιο για την Περιγραφή και Ανακάλυψη Μοτίβων Ασφάλειας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Ματθαίου Χουλιάρα

Επιβλέπων: Κρητικός Κυριάκος

Μέλη εξεταστικής επιτροπής: Καρύδα Μαρία, Κωστούλας Θεόδωρος

Σάμος, Οκτώβριος 2026

Η σελίδα αυτή είναι σκόπιμα λευκή.

Πρόλογος

Η ασφάλεια των πληροφοριακών συστημάτων αποτελεί σήμερα αναπόσπαστο μέρος της διαδικασίας ανάπτυξης λογισμικού. Η συνεχής εξέλιξη των τεχνολογιών, η αυξανόμενη πολυπλοκότητα των εφαρμογών και η εμφάνιση νέων απειλών καθιστούν αναγκαία την ενσωμάτωση της ασφάλειας ήδη από τα πρώτα στάδια σχεδίασης και ανάπτυξης ενός συστήματος και όχι μόνο ως μια συμπληρωματική διαδικασία μετά την ολοκλήρωσή του.

Μία από τις προσεγγίσεις που μπορούν να συμβάλουν προς αυτή την κατεύθυνση είναι η αξιοποίηση των μοτίβων ασφάλειας (security patterns). Τα μοτίβα ασφάλειας αποτυπώνουν επαναχρησιμοποιήσιμες λύσεις σε προβλήματα ασφάλειας που εμφανίζονται συχνά κατά την ανάπτυξη λογισμικού και μπορούν να λειτουργήσουν ως ένα χρήσιμο μέσο μεταφοράς γνώσης και εμπειρίας μεταξύ ερευνητών, αναλυτών και προγραμματιστών.

Ωστόσο, η σχετική γνώση βρίσκεται κατανεμημένη σε διαφορετικές βιβλιογραφικές πηγές και παρουσιάζεται με διαφορετικούς τρόπους, γεγονός που μπορεί να δυσκολεύει τον εντοπισμό του κατάλληλου μοτίβου για ένα συγκεκριμένο πρόβλημα. Η ανάγκη για μια περισσότερο οργανωμένη, σημασιολογικά δομημένη και εύκολα αναζητήσιμη αναπαράσταση αυτής της γνώσης αποτέλεσε το βασικό κίνητρο για την εκπόνηση της παρούσας διπλωματικής εργασίας.

Στο πλαίσιο αυτό, η εργασία επικεντρώνεται στη σχεδίαση και ανάπτυξη ενός σημασιολογικού πλαισίου για την περιγραφή, οργάνωση και αναζήτηση μοτίβων ασφάλειας. Για τον σκοπό αυτό αξιοποιούνται τεχνολογίες του Σημασιολογικού Ιστού και μια οντολογία, μέσω της οποίας τα μοτίβα ασφάλειας μπορούν να συσχετίζονται με έννοιες όπως οι απειλές που αντιμετωπίζουν, τα προστατευόμενα αγαθά, οι ιδιότητες ασφάλειας και οι φάσεις ανάπτυξης λογισμικού στις οποίες εφαρμόζονται.

Παράλληλα, αναπτύσσεται μία διαδικτυακή εφαρμογή μέσω της οποίας ο χρήστης μπορεί να εξερευνά και να διαχειρίζεται τη γνώση που περιλαμβάνεται στην οντολογία, καθώς και να αναζητά μοτίβα ασφάλειας βάσει των χαρακτηριστικών και των αναγκών ενός προβλήματος. Στόχος της εργασίας είναι, επομένως, να συνδυάσει τη θεωρητική γνώση γύρω από τα μοτίβα ασφάλειας με μία πρακτική εφαρμογή που διευκολύνει την πρόσβαση, την οργάνωση και την αξιοποίησή τους κατά τη διαδικασία ανάπτυξης ασφαλούς λογισμικού.

Ευχαριστίες

Η ολοκλήρωση της παρούσας διπλωματικής εργασίας δεν θα ήταν δυνατή χωρίς τη συμβολή και την υποστήριξη ανθρώπων που στάθηκαν δίπλα μου σε όλη τη διάρκεια αυτής της προσπάθειας.

Θα ήθελα αρχικά να εκφράσω τις θερμές μου ευχαριστίες στον επιβλέποντα της εργασίας μου, **κ. Κυριάκο Κρητικό**, για την καθοδήγηση, τις παρατηρήσεις και τις υποδείξεις του καθ' όλη τη διάρκεια της εκπόνησης της εργασίας. Η επιστημονική του υποστήριξη και η κριτική του προσέγγιση συνέβαλαν ουσιαστικά στη βελτίωση τόσο της ίδιας της εργασίας όσο και του τρόπου με τον οποίο αντιμετώπισα το αντικείμενό της.

Ιδιαίτερα θα ήθελα να ευχαριστήσω τη γυναίκα μου, **Κυριακή**, για την υπομονή, την κατανόηση και τη συνεχή στήριξή της σε όλη αυτή την πορεία. Η παρουσία της δίπλα μου, ιδιαίτερα στις στιγμές που οι υποχρεώσεις και οι απαιτήσεις της εργασίας γίνονταν δυσκολότερες, υπήρξε ανεκτίμητη.

Τέλος, η μεγαλύτερη και πιο ξεχωριστή ευχαριστία ανήκει στα δύο αγαπημένα μας παιδιά, **τον Ιωνά και τον Θησέα**. Αποτελούν καθημερινά τη μεγαλύτερη πηγή δύναμης, χαράς και έμπνευσής μου και, με τον δικό τους μοναδικό τρόπο, έδωσαν ακόμη μεγαλύτερη αξία στην ολοκλήρωση αυτής της προσπάθειας**.** Σε αυτούς τους ανθρώπους αφιερώνω, με αγάπη και ευγνωμοσύνη, την ολοκλήρωση αυτής της εργασίας.

© 2026

του

Ματθαίου Χουλιάρα

Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

Η σελίδα αυτή είναι σκόπιμα λευκή.

Πίνακας περιεχομένων

1 Εισαγωγή **1**
1.1 Ασφάλεια λογισμικού και μοτίβα ασφάλειας 1
1.1.1 Σημασιολογική και οντολογική αναπαράσταση μοτίβων ασφάλειας 2
1.2 Αντικείμενο διπλωματικής 2
1.3 Στόχος και συνεισφορά της προτεινόμενης λύσης 3
1.4 Δομή της διπλωματικής 3
2 Θεωρητικό Υπόβαθρο **5**
2.1 Γλώσσες Προσδιορισμού Τομέα / Εννοιολογικά Μοντέλα 5
2.1.1 Ημιδομημένες Γλώσσες (Semi-Structural Languages) 5
2.1.2 Ημιτυπικές Γλώσσες (Semi-Formal) 6
2.1.3 Τυπικές (Formal) 15
2.1.4 Σύγκριση γλωσσών (UML και OWL) 21
2.1.5 Σύγκριση Γλωσσικής Αναπαράστασης και Σκοπού 22
2.2 Μοτίβα Ασφάλειας 25
2.2.1 Ορισμός 25
2.2.2 Συσχέτιση Μοτίβων Ασφάλειας με τη Διαδικασία Ανάπτυξης Λογισμικού 31
2.3 Σχετικές Ερευνητικές Εργασίες 33
2.3.1 Ταξινόμηση Μοτίβων Ασφάλειας 33
2.3.2 Περιγραφή Μοτίβων Ασφάλειας 34
2.3.3 Ανίχνευση Μοτίβων Ασφάλειας 38
2.3.4 Αποθετήρια και Βάσεις Γνώσεις Μοτίβων Ασφάλειας 38
3 Προτεινόμενη λύση **40**
3.1 Περίγραμμα 45
3.2 Ανάλυση Απαιτήσεων 46
3.2.1 Λειτουργικές απαιτήσεις 47
3.2.2 Μη Λειτουργικές απαιτήσεις 49
3.3 Μοντελοποίηση Περιπτώσεων Χρήσης 51
3.4 Εννοιολογική Σχεδίαση 51
3.5 Δημιουργία Οντολογίας 56
3.5.1 Ιεραρχία κλάσεων 56
3.5.2 Datatype Properties και Object Properties 58
3.5.3 Individuals σχετικών εννοιών 60
3.5.4 Security pattern individuals 61
3.5.5 Επικύρωση της οντολογίας 63
3.6 Αρχιτεκτονική 65
3.6.1 Επίπεδο Παρουσίασης 66
3.6.2 Επίπεδο Επιχειρησιακής Λογικής 66
3.6.3 Επίπεδο Πρόσβασης Δεδομένων 67
3.6.4 Σημασιολογική Βάση Γνώσης και Αναπαράσταση στη Μνήμη 67

3.6.5 Ροή Ανάγνωσης 68
3.6.6 Ροή Μεταβολής 71
3.6.7 Διαχείριση Εξαρτήσεων 74
3.7 Υλοποίηση 75
3.7.1 Αρχικοποίηση της εφαρμογής και σύνδεση με την οντολογία 75
3.7.2 Φόρτωση της OWL οντολογίας και δημιουργία του RDF γράφου 76
3.7.3 Ανάκτηση μοτίβων μέσω SPARQL 76
3.7.4 Μετατροπή των σημασιολογικών αποτελεσμάτων στο μοντέλο της εφαρμογής 77
3.7.5 Δημιουργία και επεξεργασία security patterns 79
3.7.6 Λογική διαγραφή 81
3.7.7 Διεπαφή SPARQL 81
3.7.8 Διεπαφή χρήστη και πολυκριτηριακή αναζήτηση μοτίβων 82
3.7.9 Συνολική ροή υλοποίησης 83
4 Εγκατάσταση και Επίδειξη της Εφαρμογής 85
4.1 Σκοπός και περιβάλλον εκτέλεσης 85
4.2 Προετοιμασία και εκκίνηση 85
4.3 Περιήγηση στη βιβλιοθήκη και αναζήτηση μοτίβων 86
4.4 Αναλυτική προβολή σημασιολογικής περιγραφής 88
4.5 Δημιουργία, επεξεργασία και λογική διαγραφή 88
4.6 Εξειδικευμένη διερεύνηση μέσω SPARQL 88
4.7 Σύνοψη της επίδειξης 89
5 Αποτίμηση Προτεινόμενης Λύσης 90
5.1 Ικανοποίηση Απαιτήσεων 90
5.2 Σύγκριση με Σχετικές Ερευνητικές Εργασίες 92
5.3 Πλεονεκτήματα και Περιορισμοί της Προτεινόμενης Λύσης 92
6 Συμπεράσματα και Μελλοντική Εργασία 94
6.1 Συμπεράσματα 94
6.2 Μελλοντική Εργασία 95
Βιβλιογραφία 97
Παράρτημα Ι - Επιλεγμένα Αποσπάσματα Κώδικα 101
Ι.1 Αρχικοποίηση της εφαρμογής και σύνδεση με την οντολογία 101
Ι.2 Φόρτωση της OWL οντολογίας και δημιουργία του RDF γράφου 102
Ι.3 Ανάκτηση ενεργών μοτίβων ασφάλειας μέσω SPARQL 103
Ι.4 Δημιουργία νέου μοτίβου ασφάλειας στην OWL οντολογία 104
Ι.5 Λογική διαγραφή μοτίβου ασφάλειας 105
Ι.6 Περιορισμοί ασφαλούς εκτέλεσης SPARQL επερωτήσεων 106

Λίστα Σχημάτων/Εικόνων

Εικόνα 1: Παράδειγμα εφαρμογής περιορισμού OCL σε στερεότυπο UML Profile, όπου το στερεότυπο «Counter» επεκτείνει τη μετακλάση NamedElement και υπόκειται σε τυπικό περιορισμό OCL (Eclipse OCL Documentation). 7

Εικόνα 2: Διάγραμμα κλάσης που απεικονίζει τις σχέσεις μεταξύ των κλάσεων οντολογίας και των ατόμων, σε ένα OUP. 10

Εικόνα 3: Διάγραμμα κλάσης UML που απεικονίζει ιδιότητες οντολογίας. 13

Εικόνα 4: Απεικόνιση Individuals και Statements σε ένα διάγραμμα αντικειμένων UML. 14

Εικόνα 5: Χρησιμοποιημένη αρχή XSLT: επεκτάσεις εργαλείων UML για την ανάπτυξη της οντολογίας. 14

Εικόνα 6: Σχέση μεταξύ OWL, RDF, SPARQL και μηχανισμών συλλογιστικής στο σημασιολογικό πλαίσιο της παρούσας εργασίας. 18

Εικόνα 7: Πλαίσιο τεσσάρων επιπέδων των τεχνολογιών XML, RDF, RDFS και OWL. 23

Εικόνα 8: Διάγραμμα περιπτώσεων χρήσης των βασικών λειτουργιών της εφαρμογής. 51

Εικόνα 9: UML διάγραμμα κλάσεων του επιχειρησιακού μοντέλου ενός μοτίβου ασφάλειας και των βασικών σημασιολογικών του συσχετίσεων. 53

Εικόνα 10: Ιεραρχία των βασικών κλάσεων και των κατηγοριών μοτίβων ασφάλειας στην οντολογία, όπως απεικονίζεται στο Protégé. 57

Εικόνα 11: Η κλάση SecurityPattern και η θέση της στην οντολογική δομή όπως παρουσιάζεται στο Protégé. 58

Εικόνα 12: Ορισμός της Object Property mitigatesThreat στο Protégé, με domain την SecurityPattern και range την Threat. 60

Εικόνα 13: Παράδειγμα βιβλιογραφικής πηγής και των μοτίβων ασφάλειας που τεκμηριώνονται από αυτή μέσω της σχέσης documentsPattern. 61

Εικόνα 14: Παράδειγμα πλήρους σημασιολογικής περιγραφής του μοτίβου Authentication Enforcer στο Protégé. 62

Εικόνα 15: Inferred ιεραρχία κλάσεων μετά την ταξινόμηση της οντολογίας με τον HermiT reasoner. 64

Εικόνα 16: Πολυεπίπεδη αρχιτεκτονική της εφαρμογής και ροή επικοινωνίας με τη σημασιολογική βάση γνώσης. 65

Εικόνα 17: Διάγραμμα ακολουθίας για την περιήγηση και αναζήτηση μοτίβων ασφάλειας. 69

Εικόνα 18: Διάγραμμα ακολουθίας για την αναλυτική προβολή ενός μοτίβου ασφάλειας. 70

Εικόνα 19: Διάγραμμα ακολουθίας για την εκτέλεση read-only SPARQL επερώτησης. 71

Εικόνα 20: Διάγραμμα ακολουθίας για τη δημιουργία νέου μοτίβου ασφάλειας. 72

Εικόνα 21: Διάγραμμα ακολουθίας για την επεξεργασία μοτίβου ασφάλειας. 73

Εικόνα 22: Διάγραμμα ακολουθίας για τη λογική διαγραφή μοτίβου ασφάλειας. 74

Εικόνα 23: Αναλυτική παρουσίαση της σημασιολογικής πληροφορίας του Authorization Enforcer στη διεπαφή της εφαρμογής. 78

Εικόνα 24: Φόρμα δημιουργίας security pattern με περιγραφικά στοιχεία και επιλογές που προέρχονται από την οντολογία. 80

Εικόνα 25: Εκτέλεση read-only SPARQL SELECT query μέσω της ειδικής διεπαφής της εφαρμογής. 82

Εικόνα 26: Κύρια σελίδα προβολής και αναζήτησης της βιβλιοθήκης security patterns. 83

Εικόνα 27: Στιγμιότυπο πραγματικών αποτελεσμάτων αναζήτησης με Risk Level. 87

Εικόνα 28: Στιγμιότυπο πραγματικών αποτελεσμάτων αναζήτησης με Related Pattern. 87

Λίστα Πινάκων

Πίνακας 1: Βασικές σημασιολογικές σχέσεις του εννοιολογικού μοντέλου των μοτίβων ασφάλειας. 54

Πίνακας 2: Βασικές περιγραφικές ιδιότητες (Datatype Properties) των μοτίβων ασφάλειας στην οντολογία. 59

Πίνακας 3: Βασικά στοιχεία και αρμοδιότητες της αρχιτεκτονικής της εφαρμογής. 75

Πίνακας 4: Αποτίμηση της ικανοποίησης των βασικών απαιτήσεων της εφαρμογής. 91

Ακρωνύμια

API	Application Programming Interface
CASE	Computer-Aided Software Engineering
CIA	Confidentiality, Integrity, Availability
CTI	Cyber Threat Intelligence
DL	Description Logic
DOI	Digital Object Identifier
DSL	Domain-Specific Language
GoF	Gang of Four
IoT	Internet of Things
IRI	Internationalized Resource Identifier
J2EE	Java 2 Platform, Enterprise Edition
KR	Knowledge Representation
MDA	Model-Driven Architecture
MOF	Meta-Object Facility
MVC	Model-View-Controller
NDMM	Non-uniform Distributed Matrix Matching
NIST	National Institute of Standards and Technology
OCL	Object Constraint Language
ODM	Ontology Definition Metamodel
OMG	Object Management Group
OMM	Ordered Matrix Matching
OUP	Ontology UML Profile
OWA	Open World Assumption
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	RDF Schema
SCADA	Supervisory Control and Data Acquisition
SGML	Standard Generalized Markup Language
SLR	Systematic Literature Review
SOA	Service-Oriented Architecture

SPARQL	SPARQL Protocol and RDF Query Language
SPD	Security Pattern Detection
STRIDE	Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege
UI	User Interface
UML	Unified Modeling Language
UNA	Unique Name Assumption
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XMI	XML Metadata Interchange
XML	eXtensible Markup Language
XSLT	eXtensible Stylesheet Language Transformations

Περίληψη

Η ανάπτυξη ασφαλών πληροφοριακών συστημάτων αποτελεί μία σύνθετη διαδικασία, καθώς απαιτεί την αναγνώριση πιθανών απειλών και την επιλογή κατάλληλων μηχανισμών προστασίας ήδη από τα πρώτα στάδια του κύκλου ζωής του λογισμικού. Προς αυτό τον σκοπό τα *μοτίβα ασφάλειας* (security patterns) παρέχουν επαναχρησιμοποιήσιμες και τεκμηριωμένες λύσεις σε επαναλαμβανόμενα προβλήματα ασφάλειας. Ωστόσο, η γνώση που αφορά τα διαθέσιμα μοτίβα βρίσκεται κατανεμημένη σε διαφορετικές βιβλιογραφικές πηγές και συχνά περιγράφεται με διαφορετική ορολογία και επίπεδο λεπτομέρειας, γεγονός που δυσχεραίνει τον αποτελεσματικό εντοπισμό και την αξιοποίησή τους.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η σχεδίαση και ανάπτυξη ενός σημασιολογικού πλαισίου για τη δομημένη περιγραφή, οργάνωση, διασύνδεση και αναζήτηση μοτίβων ασφάλειας. Για τον σκοπό αυτό σχεδιάστηκε *οντολογία* βασισμένη στις τεχνολογίες του *Σημασιολογικού Ιστού* και στη γλώσσα *OWL*, μέσω της οποίας τα μοτίβα ασφάλειας αναπαρίστανται ως σημασιολογικά συσχετισμένες οντότητες. Εκτός από τα περιγραφικά χαρακτηριστικά τους, τα μοτίβα συνδέονται με έννοιες όπως οι απειλές που αντιμετωπίζουν, τα προστατευόμενα αγαθά, οι ιδιότητες ασφάλειας που υποστηρίζουν, οι φάσεις ανάπτυξης λογισμικού στις οποίες εφαρμόζονται, το επίπεδο εφαρμογής τους και οι μεταξύ τους σχέσεις. Ιδιαίτερη έμφαση δόθηκε επίσης στην τεκμηρίωση της προέλευσης των μοτίβων μέσω βιβλιογραφικών πηγών, ώστε η γνώση που περιλαμβάνεται στη σημασιολογική βάση να παραμένει ανιχνεύσιμη και επαληθεύσιμη.

Παράλληλα, αναπτύχθηκε διαδικτυακή εφαρμογή η οποία αξιοποιεί την οντολογία ως σημασιολογική βάση γνώσης και παρέχει λειτουργίες προβολής, αναζήτησης και διαχείρισης μοτίβων ασφάλειας. Η εφαρμογή υλοποιήθηκε με τεχνολογίες ASP.NET και Razor, ενώ η πρόσβαση και επεξεργασία των *RDF/OWL* δεδομένων πραγματοποιείται με τη βιβλιοθήκη dotNetRDF. Η *σημασιολογική αναπαράσταση* επιτρέπει την αναζήτηση των μοτίβων όχι μόνο βάσει απλών λεκτικών χαρακτηριστικών αλλά και βάσει των εννοιών και των σχέσεων που τα χαρακτηρίζουν μέσω της γλώσσας επερωτήσεων *SPARQL*.

Το αποτέλεσμα της εργασίας αποτελεί ένα ενιαίο πλαίσιο στο οποίο η βιβλιογραφικά τεκμηριωμένη γνώση γύρω από τα μοτίβα ασφάλειας οργανώνεται σε μηχανικά επεξεργάσιμη μορφή και καθίσταται προσβάσιμη μέσω μιας διαδικτυακής εφαρμογής. Με τον τρόπο αυτό επιδιώκεται να διευκολυνθεί η αναζήτηση, η κατανόηση και η επιλογή κατάλληλων μοτίβων ασφάλειας και να υποστηριχθούν αναλυτές, σχεδιαστές και προγραμματιστές κατά τη διαδικασία *ανάπτυξης ασφαλέστερων συστημάτων λογισμικού*.

Λέξεις Κλειδιά: *Μοτίβα Ασφάλειας, Σημασιολογικός Ιστός, Οντολογία, OWL, RDF, SPARQL, Σημασιολογική Αναπαράσταση και Αναζήτηση, Ασφαλής Ανάπτυξη Λογισμικού*

Abstract

The development of secure information systems is a complex process, as it requires the identification of potential threats and the selection of appropriate protection mechanisms from the early stages of the software development lifecycle. Towards this end, *security patterns* provide reusable and documented solutions to recurring security problems. However, knowledge concerning existing security patterns is distributed across different literature sources and is often described using different terminology and levels of detail, making their effective discovery and reuse more difficult.

The aim of this thesis is the design and development of a *semantic framework* for the structured description, organization, interconnection, and discovery of security patterns. For this purpose, an *ontology* was designed using Semantic Web technologies and the Web Ontology Language (*OWL*), allowing security patterns to be represented as semantically related entities. Apart from their descriptive characteristics captured, the patterns are associated with concepts such as the threats they mitigate, the assets they protect, the security properties they support, the software development phases in which they can be applied, their level of application, and their relationships with other patterns. Particular emphasis was also placed on bibliographic provenance, so that the knowledge contained in the semantic knowledge base remains traceable and verifiable.

In parallel, a web application was developed that exploits the ontology as its semantic knowledge base and supplies functionality for browsing, searching, and managing security patterns. The application was implemented using ASP.NET and Razor technologies, while *RDF/OWL* data access and manipulation are facilitated through the dotNetRDF library. The semantic representation enables patterns to be discovered not only through textual characteristics but also through the concepts and relationships that characterize them via the use of the *SPARQL* query language.

The outcome of this work constitutes an integrated framework in which the literature-based knowledge regarding security patterns is organized in a machine-processable form and made accessible through a web application. The proposed approach aims to facilitate the discovery, understanding, and selection of suitable security patterns and to support analysts, designers, and software developers during the *development of more secure software systems*.

Keywords: *Security Patterns, Semantic Web, Ontology, OWL, RDF, SPARQL, Semantic Representation and Search, Secure Software Development*

1 *Εισαγωγή*

1.1 Ασφάλεια λογισμικού και μοτίβα ασφάλειας

Η ασφάλεια αποτελεί θεμελιώδη μη λειτουργική ιδιότητα, η οποία πρέπει να λαμβάνεται υπόψη κατά την ανάπτυξη ενός συστήματος. Η ασφαλής κατασκευή λογισμικού μπορεί να υποστηριχθεί, μεταξύ άλλων, μέσω της αξιοποίησης προτύπων ασφάλειας (security standards) και μοτίβων ασφάλειας (security patterns).

Τα πρότυπα ασφάλειας είναι έγγραφα που παρέχουν απαιτήσεις, προδιαγραφές, κατευθυντήριες οδηγίες ή χαρακτηριστικά, τα οποία μπορούν να χρησιμοποιούνται με συνέπεια, ώστε προϊόντα, διεργασίες και υπηρεσίες να είναι κατάλληλα για τον σκοπό τους (NIST, n.d.). Σε αντίθεση με αυτά, τα μοτίβα ασφάλειας εστιάζουν σε περισσότερο συγκεκριμένες και επαναχρησιμοποιήσιμες λύσεις για επαναλαμβανόμενα προβλήματα ασφάλειας που εμφανίζονται κατά την ανάπτυξη ή τη λειτουργία ενός συστήματος λογισμικού.

Ειδικότερα ένα μοτίβο ασφάλειας (security pattern) αποτελεί τεκμηριωμένη και επαναχρησιμοποιήσιμη λύση σε ένα επαναλαμβανόμενο πρόβλημα ασφάλειας, η οποία μπορεί να εφαρμοστεί σε συγκεκριμένο πλαίσιο ανάπτυξης ή λειτουργίας ενός συστήματος λογισμικού (Jafari & Rasoolzadegan, 2020). Στην παρούσα εργασία, ο όρος δεν περιορίζεται αποκλειστικά στην ασφάλεια πληροφοριών, αλλά καλύπτει γενικότερα ζητήματα ασφάλειας λογισμικού, πληροφοριακών συστημάτων και εφαρμογών, όπως ο έλεγχος πρόσβασης, η αυθεντικοποίηση, η επικύρωση εισόδου, η κρυπτογραφία, η καταγραφή συμβάντων και η ασφαλής διαμόρφωση.

Ένα μοτίβο ασφάλειας ενσωματώνει τεχνογνωσία σε θέματα ασφάλειας με τη μορφή δομημένης λύσης, παρουσιάζοντας το πρόβλημα που αντιμετωπίζεται, το πλαίσιο στο οποίο εμφανίζεται, τη λύση που προτείνεται και τους πιθανούς συμβιβασμούς που προκύπτουν από την εφαρμογή της. Στην ουσία, περιγράφει ένα συγκεκριμένο επαναλαμβανόμενο πρόβλημα ασφάλειας που εμφανίζεται σε συγκεκριμένα πλαίσια και παρουσιάζει ένα γενικό και δοκιμασμένο σχήμα για την αντιμετώπισή του.

Η χρήση μοτίβων ασφάλειας προσφέρει σημαντικά πλεονεκτήματα, καθώς επιτρέπει την επαναχρησιμοποίηση δοκιμασμένων λύσεων, τη συστηματική αντιμετώπιση συχνών προβλημάτων ασφάλειας και τη δημιουργία κοινού λεξιλογίου μεταξύ αναλυτών, σχεδιαστών, προγραμματιστών και ειδικών ασφάλειας.

1.1.1 Σημασιολογική και οντολογική αναπαράσταση μοτίβων ασφάλειας

Οι οντολογικές προσεγγίσεις στα μοτίβα ασφάλειας έχουν ως στόχο την προσθήκη σαφούς σημασιολογίας στις έννοιες και στις σχέσεις που τα περιγράφουν. Η οντολογική αναπαράσταση μπορεί να καταστήσει την περιγραφή και τη χρήση των μοτίβων περισσότερο ακριβή και να υποστηρίξει την ανάπτυξη εργαλείων ικανών να παρουσιάζουν σχετικά μοτίβα στους συμμετέχοντες στη διαδικασία ανάπτυξης λογισμικού, ανάλογα με το πρόβλημα και το στάδιο ανάπτυξης στο οποίο βρίσκονται.

Παράλληλα, οι οντολογικές προσεγγίσεις συμβάλλουν στη δημιουργία καλύτερα οργανωμένων καταλόγων μοτίβων ασφάλειας, καθώς κάθε μοτίβο μπορεί να συσχετίζεται σημασιολογικά με έννοιες, όπως οι απειλές που αντιμετωπίζει, τα προστατευόμενα αγαθά, οι ιδιότητες ασφάλειας που υποστηρίζει, η κατηγορία στην οποία ανήκει και οι φάσεις ανάπτυξης στις οποίες μπορεί να εφαρμοστεί.

1.2 Αντικείμενο διπλωματικής

Αντικείμενο της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη ενός σημασιολογικού πλαισίου για την περιγραφή, οργάνωση, αποθήκευση και αναζήτηση μοτίβων ασφάλειας, καθώς και η ανάπτυξη μιας διαδικτυακής εφαρμογής μέσω της οποίας οι συμμετέχοντες σε ένα έργο λογισμικού μπορούν να εντοπίζουν και να διαχειρίζονται τα συγκεκριμένα μοτίβα.

Η εφαρμογή επιτρέπει στους χρήστες να αναζητούν με σημασιολογικό τρόπο μοτίβα ασφάλειας, καθώς και να δημιουργούν νέα, με σκοπό την υποστήριξη της ενσωμάτωσης πρακτικών ασφάλειας στα διαφορετικά στάδια και στις δραστηριότητες της διαδικασίας ανάπτυξης λογισμικού. Ειδικότερα, περιλαμβάνει διεπαφή χρήσης βασισμένη σε φόρμες για την προβολή, δημιουργία, επεξεργασία και αναζήτηση των μοτίβων.

Στη ραχοκοκαλιά της εφαρμογής βρίσκεται μια σημασιολογική βάση γνώσης, η οποία επιτρέπει τη σημασιολογική περιγραφή, αποθήκευση και αναζήτηση των μοτίβων ασφάλειας. Η βάση γνώσης στηρίζεται σε ένα σημασιολογικό μοντέλο περιγραφής μοτίβων ασφάλειας, το οποίο αναπτύξαμε με τη γλώσσα οντολογιών OWL (Web Ontology Language).

Με βάση το μοντέλο αυτό, τα μοτίβα ασφάλειας δεν αποθηκεύονται αποκλειστικά ως απλές περιγραφές κειμένου, αλλά οργανώνονται σε σημασιολογικές κατηγορίες και συνδέονται με σχετικές έννοιες. Ενδεικτικές κατηγορίες αποτελούν τα μοτίβα αυθεντικοποίησης (authentication patterns), εξουσιοδότησης (authorization patterns), επικύρωσης εισόδου (input validation patterns), κρυπτογραφίας (cryptography patterns), διαχείρισης συνεδρίας (session management patterns),

καταγραφής και παρακολούθησης (logging and monitoring patterns), ασφαλούς διαμόρφωσης (secure configuration patterns) και ασφάλειας διεπαφών API (API security patterns).

Παράλληλα, κάθε μοτίβο μπορεί να συνδέεται με στοιχεία όπως η απειλή που αντιμετωπίζει (mitigated threat), το προστατευόμενο αγαθό (protected asset), η ιδιότητα ασφάλειας που υποστηρίζει (security property), το πλαίσιο εφαρμογής (context), το πρόβλημα (problem), η προτεινόμενη λύση (solution), οι συνέπειες εφαρμογής (consequences) και η φάση ανάπτυξης στην οποία μπορεί να εφαρμοστεί (development phase).

1.3 Στόχος και συνεισφορά της προτεινόμενης λύσης

Βασικός στόχος της παρούσας εργασίας είναι η δημιουργία ενός πρακτικού τρόπου αξιοποίησης μιας σημασιολογικής βάσης γνώσης μοτίβων ασφάλειας, χωρίς να απαιτείται από τον τελικό χρήστη άμεση γνώση ή επεξεργασία τεχνολογιών όπως OWL, RDF (Resource Description Framework) και SPARQL.

Μέσω της εφαρμογής, ο χρήστης μπορεί να αναζητά, να μελετά, να δημιουργεί, να επεξεργάζεται και να διαχειρίζεται μοτίβα ασφάλειας μέσω μιας συμβατικής διαδικτυακής διεπαφής. Με τον τρόπο αυτό, η πολυπλοκότητα της σημασιολογικής αναπαράστασης αποκρύπτεται από τον τελικό χρήστη, ενώ εξακολουθούν να αξιοποιούνται οι δυνατότητες που παρέχει η οντολογία για τη δομημένη οργάνωση και συσχέτιση της γνώσης.

Η προτεινόμενη λύση οργανώνει τα μοτίβα ασφάλειας σύμφωνα με μια ενιαία δομή, επιτρέπει τη σημασιολογική σύνδεσή τους με σχετικά προβλήματα και έννοιες ασφάλειας και διευκολύνει τον εντοπισμό σχετικής γνώσης. Κατά συνέπεια, μπορεί να χρησιμοποιηθεί ως εργαλείο υποστήριξης από φοιτητές, προγραμματιστές, αναλυτές και μηχανικούς λογισμικού που επιθυμούν να ενσωματώσουν πρακτικές και αρχές ασφάλειας στη διαδικασία ανάπτυξης λογισμικού.

1.4 Δομή της διπλωματικής

Το υπόλοιπο της παρούσας διπλωματικής εργασίας οργανώνεται σε πέντε κεφάλαια. Στο Κεφάλαιο 2 παρουσιάζεται το θεωρητικό υπόβαθρο, με έμφαση στις γλώσσες μοντελοποίησης και αναπαράστασης γνώσης, στις τεχνολογίες του Σημασιολογικού Ιστού, στα μοτίβα ασφάλειας και στις σχετικές ερευνητικές προσεγγίσεις

Στο Κεφάλαιο 3 αναλύεται η προτεινόμενη λύση, από τον προσδιορισμό των απαιτήσεων και την εννοιολογική σχεδίαση έως την ανάπτυξη της οντολογίας, την αρχιτεκτονική και την υλοποίηση της διαδικτυακής εφαρμογής.

Στο Κεφάλαιο 4 παρουσιάζονται οι προϋποθέσεις εγκατάστασης και εκτέλεσης της εφαρμογής, καθώς και ενδεικτικά σενάρια χρήσης που αναδεικνύουν τις δυνατότητες αναζήτησης, προβολής και διαχείρισης μοτίβων ασφάλειας και εκτέλεσης σημασιολογικών επερωτήσεων.

Στο Κεφάλαιο 5 πραγματοποιείται η αποτίμηση της προτεινόμενης λύσης ως προς την ικανοποίηση των απαιτήσεων, τη σχέση της με συναφείς ερευνητικές εργασίες και τα βασικά πλεονεκτήματα και τους περιορισμούς της.

Στο Κεφάλαιο 6 συνοψίζονται τα συμπεράσματα της εργασίας και παρουσιάζονται κατευθύνσεις για μελλοντική έρευνα και περαιτέρω επέκταση της εφαρμογής και του οντολογικού μοντέλου.

Τέλος, στο Παράρτημα Ι παρατίθενται επιλεγμένα αποσπάσματα κώδικα που συμπληρώνουν την περιγραφή της υλοποίησης.

2 Θεωρητικό Υπόβαθρο

2.1 Γλώσσες Προσδιορισμού Τομέα / Εννοιολογικά Μοντέλα

2.1.1 Ημιδομημένες Γλώσσες (Semi-Structural Languages)

Η XML (eXtensible Markup Language) αναπτύχθηκε από Ομάδα Εργασίας του World Wide Web Consortium (W3C), η οποία συγκροτήθηκε το 1996, ενώ η πρώτη έκδοση της XML 1.0 δημοσιεύθηκε ως W3C Recommendation το 1998. Βασικοί σχεδιαστικοί στόχοι της XML ήταν η άμεση αξιοποίησή της στο Διαδίκτυο, η υποστήριξη ευρείας ποικιλίας εφαρμογών, η συμβατότητα με την SGML και η ευκολία ανάπτυξης προγραμμάτων για την επεξεργασία XML εγγράφων (W3C, 1998). Η αξιοποίηση αυτή επιτυγχάνεται μέσω της δυνατότητας ορισμού προσαρμοσμένων στοιχείων και μιας ιεραρχικής δομής δεδομένων, η οποία μπορεί να μεταδοθεί ως κείμενο και να υποβληθεί σε επεξεργασία από διαφορετικές εφαρμογές και πλατφόρμες μέσω XML parsers.

Κάθε έγγραφο XML διαθέτει λογική και φυσική δομή. Η λογική δομή αφορά τον τρόπο οργάνωσης του περιεχομένου του εγγράφου σε στοιχεία (elements), ιδιότητες (attributes), δεδομένα χαρακτήρων, σχόλια και άλλες δομές σήμανσης. Τα στοιχεία οργανώνονται ιεραρχικά σχηματίζοντας δενδρική δομή, στην κορυφή της οποίας βρίσκεται ένα μοναδικό ριζικό στοιχείο (document element). Ένα στοιχείο μπορεί να περιλαμβάνει στοιχεία-παιδιά, ιδιότητες και κείμενο. Ο ορισμός του πραγματοποιείται μέσω μιας ετικέτας έναρξης και μιας αντίστοιχης ετικέτας λήξης, ενώ οι ιδιότητές του εκφράζονται ως ζεύγη ονόματος–τιμής. Η φυσική δομή, αντίθετα, αφορά τις μονάδες αποθήκευσης από τις οποίες αποτελείται το έγγραφο, οι οποίες στην XML ονομάζονται οντότητες (entities). Το ίδιο το έγγραφο αποτελεί την κύρια οντότητα (document entity), ενώ είναι δυνατό να χρησιμοποιούνται και πρόσθετες εσωτερικές ή εξωτερικές οντότητες, οι οποίες ενσωματώνονται ή αναφέρονται από το έγγραφο. Με αυτόν τον τρόπο διαχωρίζεται η εννοιολογική οργάνωση του περιεχομένου από τον τρόπο με τον οποίο αυτό αποθηκεύεται και συντίθεται φυσικά (W3C, 1998).

Η XML θεωρείται θεμελιώδες πρότυπο για τη δομημένη αναπαράσταση, διαχείριση και ανταλλαγή δεδομένων. Μέσω της χρήσης μιας κοινής και ανεξάρτητης από πλατφόρμα μορφής αναπαράστασης, διαφορετικά πληροφοριακά συστήματα μπορούν να ανταλλάσσουν και να επεξεργάζονται δεδομένα με συνεπή τρόπο, ενισχύοντας έτσι τη μεταξύ τους διαλειτουργικότητα. Η XML χρησιμοποιείται, μεταξύ άλλων, για την αναπαράσταση και αποθήκευση δομημένων δεδομένων, την ανταλλαγή δεδομένων μεταξύ εφαρμογών και βάσεων δεδομένων, την επεξεργασία

και το φιλτράρισμα δεδομένων, καθώς και για την ανταλλαγή μηνυμάτων σε υπηρεσίες ιστού (web services[1]).

2.1.2 Ημιτυπικές Γλώσσες (Semi-Formal)

Η UML (Unified Modeling Language) αποτελεί πρότυπη γραφική γλώσσα μοντελοποίησης για την οπτική αναπαράσταση, την προδιαγραφή, τη σχεδίαση και την τεκμηρίωση συστημάτων λογισμικού (OMG, 2017). Χρησιμοποιείται ευρέως για τη μοντελοποίηση αντικειμενοστρεφών συστημάτων και υποστηρίζει διαφορετικούς τύπους διαγραμμάτων, μεταξύ των οποίων και το διάγραμμα κλάσεων. Το διάγραμμα κλάσεων μπορεί να αξιοποιηθεί σε διαφορετικές φάσεις του κύκλου ζωής ενός συστήματος. Κατά την ανάλυση απαιτήσεων και τη σχεδίαση μπορεί να χρησιμοποιηθεί για τη μοντελοποίηση των βασικών εννοιών, των κλάσεων και των μεταξύ τους σχέσεων, ενώ κατά την υλοποίηση μπορεί να αποτυπώσει τη στατική δομή των κλάσεων που υλοποιούνται στον πηγαίο κώδικα.

Ένα διάγραμμα κλάσεων μπορεί να χρησιμοποιηθεί για την αποτύπωση του αφηρημένου συντακτικού (abstract syntax) μιας γλώσσας μοντελοποίησης τομέα (Domain-Specific Language – DSL), για παράδειγμα μιας γλώσσας που αφορά τον τομέα των μοτίβων ασφάλειας. Το αφηρημένο συντακτικό περιγράφει τις βασικές έννοιες της γλώσσας και τις μεταξύ τους σχέσεις, ανεξάρτητα από τον τρόπο με τον οποίο αυτές αναπαρίστανται στον χρήστη. Στη συνέχεια, απαιτείται ο προσδιορισμός του συγκεκριμένου συντακτικού (concrete syntax), δηλαδή της μορφής με την οποία οι έννοιες της γλώσσας εκφράζονται γραφικά ή κειμενικά. Στο οικοσύστημα του Eclipse, η τεχνολογία Xtext[2] μπορεί να χρησιμοποιηθεί για τον ορισμό του συγκεκριμένου συντακτικού μιας κειμενικής DSL μέσω γραμματικών κανόνων, καθώς και για τη σύνδεσή του με το αντίστοιχο σημασιολογικό μοντέλο. Οι γλώσσες που προσδιορίζονται με τέτοιες τεχνικές μπορούν να προσφέρουν αυστηρά καθορισμένη σύνταξη και δομή, χωρίς όμως να διαθέτουν κατ' ανάγκη τις τυπικές σημασιολογικές δυνατότητες που παρέχουν γλώσσες οντολογιών, όπως η OWL, για σημασιολογική συσχέτιση, εξαγωγή συμπερασμάτων και συλλογιστική.

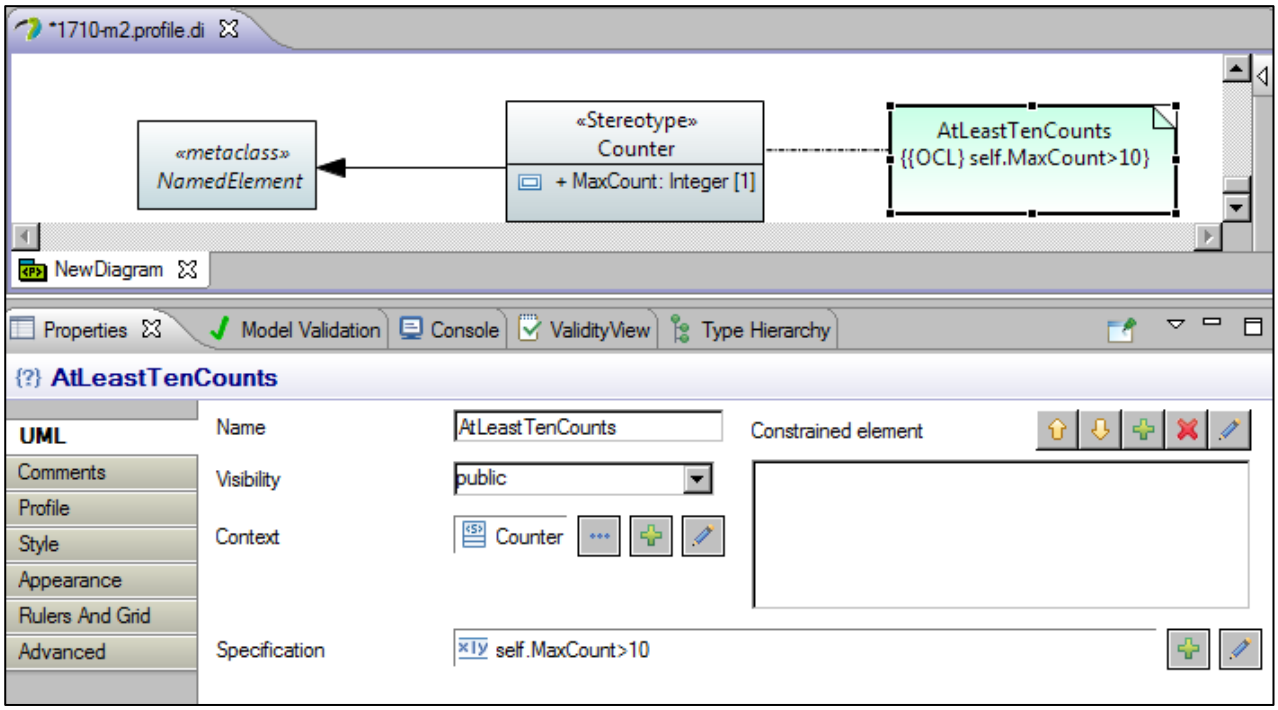
Επιπρόσθετα, οι ορισμοί ετικετών μπορούν να επισυναφθούν σε στοιχεία μοντέλου. Αυτές επιτρέπουν την εισαγωγή νέων ειδών ιδιοτήτων, οι οποίες μπορεί να έχουν τα στοιχεία του μοντέλου και είναι ανάλογες με τους ορισμούς των μετα-ιδιοτήτων. Κάθε ορισμός ετικέτας

[1] Υπηρεσία ιστού (web service) είναι ένα σύστημα λογισμικού σχεδιασμένο ώστε να υποστηρίζει διαλειτουργική επικοινωνία μεταξύ εφαρμογών μέσω δικτύου, χρησιμοποιώντας τυποποιημένες διεπαφές, πρωτόκολλα και μορφές ανταλλαγής μηνυμάτων.

[2] Το Eclipse Xtext είναι πλαίσιο ανάπτυξης γλωσσών προγραμματισμού και γλωσσών ειδικού τομέα (DSLs). Επιτρέπει τον ορισμό μιας γλώσσας μέσω γραμματικής και παρέχει υποδομή για parsing, linking, validation και υποστήριξη επεξεργαστή. Η γραμματική του Xtext περιγράφει το concrete syntax και τη χαρτογράφησή του στο αντίστοιχο semantic model. Επίσημη τεκμηρίωση: https://eclipse.dev/Xtext/documentation/

καθορίζει τις πραγματικές τιμές που έχουν οι ιδιότητες των μεμονωμένων στοιχείων του μοντέλου, οι οποίες ονομάζονται επισυναπτόμενες τιμές (tagged values).

Από την άλλη πλευρά, η χρήση περιορισμών επιτρέπει τον ακριβέστερο προσδιορισμό των κανόνων που διέπουν τα στοιχεία ενός μοντέλου. Στην περίπτωση των στερεοτύπων UML, μπορούν να οριστούν πρόσθετοι περιορισμοί μέσω της Object Constraint Language (OCL)[3], ώστε να εκφράζονται με τυπικό τρόπο συνθήκες εγκυρότητας και κανόνες που δεν μπορούν να αποτυπωθούν αποκλειστικά με τη γραφική σημειογραφία της UML (βλ. παράδειγμα στην Εικόνα 1). Ένα συνεκτικό σύνολο τέτοιων επεκτάσεων των βασικών στοιχείων της UML, το οποίο ορίζεται για συγκεκριμένο σκοπό ή τομέα εφαρμογής, συνιστά ένα UML Profile.



Εικόνα 1: Παράδειγμα εφαρμογής περιορισμού OCL σε στερεότυπο UML Profile, όπου το στερεότυπο «Counter» επεκτείνει τη μετακλάση NamedElement και υπόκειται σε τυπικό περιορισμό OCL (Eclipse OCL Documentation)

Επί του παρόντος, έχουν προταθεί αρκετές διαφορετικές προσεγγίσεις για την αναπαράσταση οντολογίας σε UML. Δύο εκ των βασικότερων είναι οι εξής:

- Επέκταση της UML με νέες δομές, για την υποστήριξη συγκεκριμένων εννοιών οντολογίας (Baclawski, 2001).

[3] Η Object Constraint Language (OCL) είναι τυπική γλώσσα προδιαγραφής του Object Management Group (OMG), η οποία χρησιμοποιείται για τη διατύπωση περιορισμών και εκφράσεων πάνω σε UML και άλλα μοντέλα που βασίζονται σε μεταμοντέλα του OMG. Επίσημη προδιαγραφή: https://www.omg.org/spec/OCL/2.4/

- Χρησιμοποιώντας μία τυπική UML και ορίζοντας ένα λειτουργικό πρότυπο UML για αναπαράσταση οντολογίας (Baclawski, 2001).

Στις περισσότερες περιπτώσεις και λόγω του ότι η αναπαράσταση οντολογίας στην UML μπορεί να επιτευχθεί χωρίς άτυπες επεκτάσεις UML, προτιμάται η δεύτερη από τις προαναφερθέντες προσεγγίσεις. Υπό αυτό το πλαίσιο, ένα λειτουργικό πρότυπο της UML για την αναπαράσταση οντολογιών ονομάζεται Ontology UML Profile (OUP).

Κατά την ανάπτυξη αυτού του προτύπου και για την εισαγωγή νέων δυνατοτήτων, συνδυάζονται συνήθως οι εμπειρίες διαφόρων σχεδιαστών π.χ. (Jürjens, 2003). Ορισμένες από τις δυνατότητες του OUP, είναι οι ακόλουθες:

- Προσφέρει στερεότυπα και ετικέτες για όλα τα επαναλαμβανόμενα στοιχεία σχεδίασης σε μια οντολογία, όπως κλάσεις, άτομα, ιδιότητες, συμπληρώματα, ενώσεις κ.α.
- Διευκολύνει την αναπαράσταση συγκεκριμένων στοιχείων οντολογίας και σχεδιασμού σε διαγράμματα UML, τα οποία παράγονται από τυπικά εργαλεία CASE (Computer-Aided Software Engineering), προσδιορίζοντας με αυτόν τον τρόπο τις οντολογικές πληροφορίες στα μοντέλα UML.
- Ενεργοποιεί την ενθυλάκωση της οντολογικής γνώσης σε μια ευανάγνωστη μορφή και την προσφέρει στους μηχανικούς του λογισμικού.
- Καθιστά δυνατή την αξιολόγηση των διαγραμμάτων οντολογίας UML, καθώς και την ένδειξη πιθανών ασυνεπειών.
- Υποστηρίζει την προδιαγραφή Ontology Definition Metamodel (ODM)[4] , ένα πρότυπο του Object Management Group (OMG) που ορίζει μεταμοντέλα, προφίλ και αντιστοιχίσεις για την αναπαράσταση και ανταλλαγή οντολογιών και σημασιολογικών μοντέλων, επιτρέποντας μεταξύ άλλων τη σύνδεση τεχνολογιών μοντελοποίησης όπως η UML με γλώσσες αναπαράστασης γνώσης.

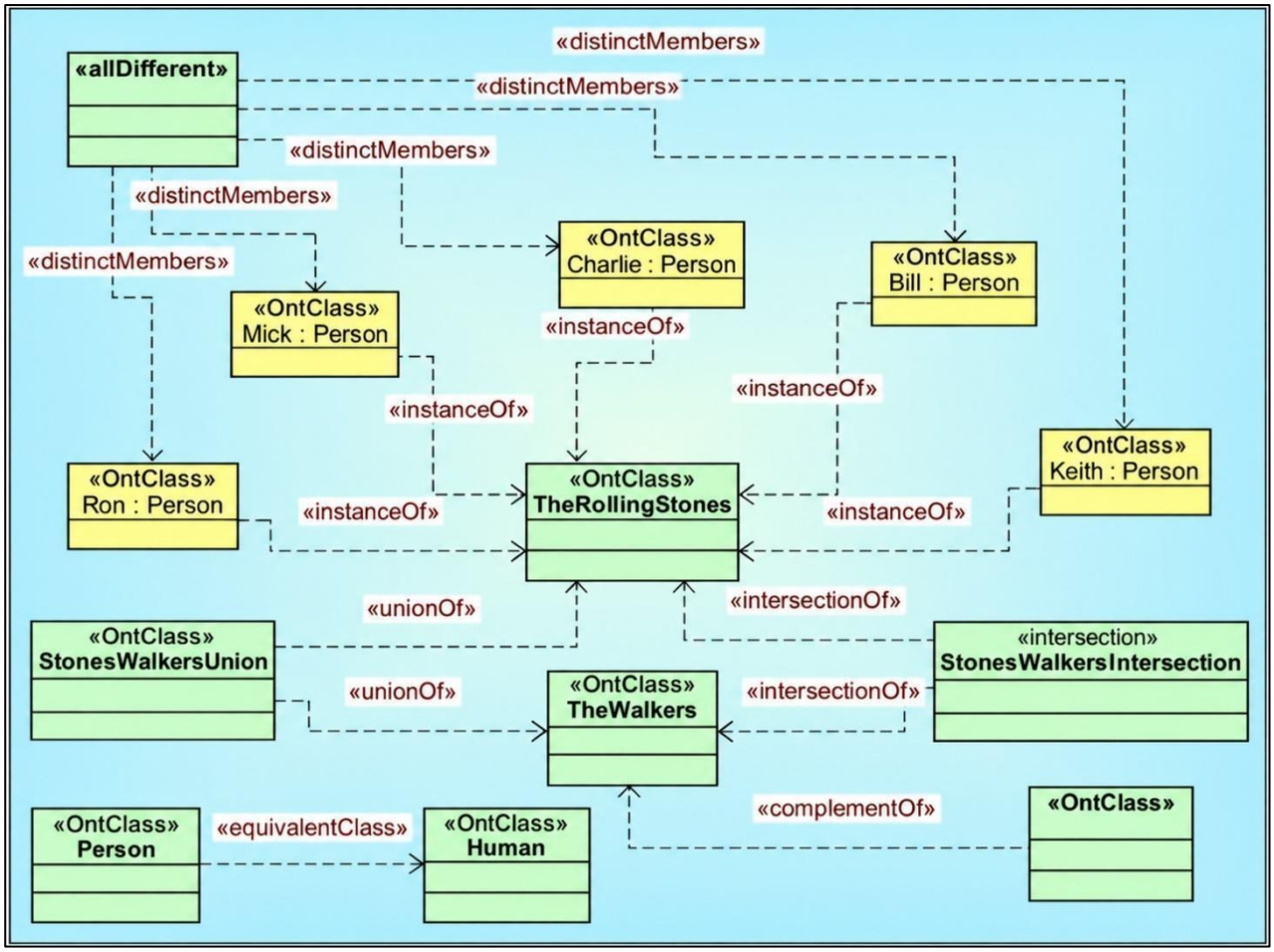
Στο OUP, σχολιάζονται συγκεκριμένες έννοιες οντολογίας χρησιμοποιώντας τους τυπικούς μηχανισμούς επέκτασης UML που περιγράφονται παραπάνω. Άρα, τα μοντέλα που δημιουργούνται μέσω ενός τέτοιου λειτουργικού προτύπου UML, υποστηρίζονται από τυπικά εργαλεία UML, διότι δεν προσθέτουν άτυπες έννοιες στην UML. Ωστόσο, σε περιπτώσεις όπου η UML χρησιμοποιείται για την υποστήριξη του ODM (δείτε παρακάτω ορισμό του) και όχι ως αυτόνομο εργαλείο για την μοντελοποίηση της οντολογίας, το OUP δεν καλύπτει όλες τις βασικές έννοιες του ODM. Επομένως, το OUP ορίζει δομές μόνο για συγκεκριμένες έννοιες, όπως ObjectProperty, Class ή Individual, αφήνοντας το ODM να ασχολείται με αφηρημένες δομές,

[4] Το Ontology Definition Metamodel (ODM) είναι προδιαγραφή του Object Management Group (OMG) για την αναπαράσταση, ανταλλαγή και αντιστοίχιση εννοιολογικών και οντολογικών μοντέλων. Η τρέχουσα επίσημη έκδοση είναι η ODM 1.1. Επίσημη προδιαγραφή: https://www.omg.org/spec/ODM/1.1/

όπως Resource, Instance, Classifier, κ.λπ., οι οποίες δεν χρησιμοποιούνται στην ανάπτυξη πραγματικών οντολογιών (μοντέλων) και δεν σχετίζονται με αντικείμενα του πραγματικού κόσμου (Real world objects). Αντιθέτως, εισάγονται στο ODM μόνο για τη δημιουργία μιας συνεκτικής ιεραρχίας.

Ο ορισμός του λειτουργικού προτύπου UML, στο πλαίσιο της αρχιτεκτονικής μεταμοντελοποίησης τεσσάρων επιπέδων MDA, εκλαμβάνεται ως επέκταση του UML στο επίπεδο του αντίστοιχου μεταμοντέλου. Επίσης, αυτές οι επεκτάσεις μπορούν να κατανοηθούν από τους χρήστες είτε ως μία νέα γλώσσα είτε ως μία οικογένεια γλωσσών. Αξίζει να σημειωθεί ότι κάθε μία από αυτές τις γλώσσες χρησιμοποιεί σημειογραφία UML, μαζί με τους τέσσερις μηχανισμούς επέκτασης που αναφέρθηκαν στην προηγούμενη υποενότητα. Η σημειογραφία που χρησιμοποιείται για τη δημιουργία στερεοτύπων του OUP (στερεότυπο «OntClass») ενσωματώνει την Κλάση (Class) του UML («μετακλάση»), ενώ η ύπαρξη αυτής της γραφικής σημειογραφίας για τον μηχανισμό επέκτασης της UML, μπορεί να είναι χρήσιμη για την εξήγηση ορισμένων σχέσεων μεταξύ των κατασκευών UML, των νέων στερεοτύπων, αλλά και μεταξύ των ίδιων των στερεοτύπων.

Στις ακόλουθες υπο-ενότητες περιγράφονται λεπτομερώς, οι πιο σημαντικές έννοιες του OUP. Γενικότερα, η κλάση είναι μια από τις πιο θεμελιώδεις έννοιες τόσο στο ODM όσο και στο OUP, ωστόσο, υπάρχουν ορισμένες διαφορές μεταξύ της παραδοσιακής κλάσης UML και της κλάσης της οντολογίας, όπως ορίζεται στην OWL (Web Ontology Language). Στο ODM, η έννοια της κλάσης αντιπροσωπεύεται ως ένα παράδειγμα της κλάσης MOF (Meta-Object Facility) και έχει πολλά συγκεκριμένα είδη (ανάλογα με την περιγραφή της κλάσης): Κλάση, Αριθμός, Ένωση, Τομή, Συμπλήρωμα, Περιορισμός και AllDifferent. Σε ένα OUP, όλες αυτές οι δομές κληρονομούνται από την έννοια της UML, η οποία είναι παρόμοια με αυτές, δηλαδή την κλάση UML. Ένα παράδειγμα διαμορφωμένων κλάσεων σε OUP, φαίνεται στην Εικόνα 2 παρακάτω.



Εικόνα 2: Διάγραμμα κλάσης που απεικονίζει τις σχέσεις μεταξύ των κλάσεων οντολογίας και των ατόμων, σε ένα OUP. (Djurić, 2004)

Η κλάση ODM, η οποία προσδιορίζεται από ένα αναγνωριστικό κλάσης, αποτελείται από τα στερεότυπα «OntClass», AllDifferent - «AllDifferent» και περιορισμού - «Restriction». Στο ODM, η αρίθμηση, το λογικό γινόμενο, η ένωση και το συμπλήρωμα είναι απόγονοι της κλάσης ODM ενώ στο OUP, οι παραπάνω έννοιες θεωρούνται ως στερεότυπα που βοηθούν στην επέκταση του στερεοτύπου «OntClass». Επιπροσθέτως, ορίζονται από επισυναπτόμενες τιμές τύπου Boole και μπορούν να προστεθούν στο «OntClass» με τον περιορισμό ότι μόνο μία από αυτές μπορεί να είναι αληθής. Ένα παράδειγμα ενός τέτοιου είδους OUP, είναι το Schema XML που διερευνήθηκε από τον Carlson (2001) και αποτελείται από τα στερεότυπα: any, choice και sequence.

Στο ODM, «Individual» ονομάζεται μία στιγμιαία κατάσταση (μιας βάσης δεδομένων) σε μία AbstractClass. Από την άλλη πλευρά, στην UML, μια αντίστοιχη στιγμιαία κατάσταση ορίζεται ως ένα αντικείμενο (Object). Αν και τα ODM Individual και UML Object έχουν κάποιες διαφορές, έχουν πολλά κοινά στοιχεία μεταξύ τους. Επομένως, στο OUP και όπως φαίνεται και στην Εικόνα 2, η «Individual» διαμορφώνεται ως UML Object. Επίσης, το στερεότυπο για ένα αντικείμενο πρέπει να ταιριάζει με το στερεότυπο της αντίστοιχης κλάσης («OntClass» σε αυτήν την περίπτωση). Η δήλωση μιας Individual γίνεται με τρεις τρόπους:

- χρησιμοποιώντας ένα υπογραμμισμένο όνομα ενός ατόμου, ακολουθούμενο από ":" και το όνομά του "ontClass" (για παράδειγμα, Mick: Person είναι ένα άτομο του οποίου ο τύπος είναι Person). Αυτή είναι η συνήθης μέθοδος UML για τη δήλωση του τύπου ενός Object.
- χρησιμοποιώντας το στερεότυπο «instanceOf» μιας UML Dependency (εξάρτησης) μεταξύ ενός ατόμου και της «ontClass» του. Αυτή η μέθοδος επιτρέπεται επίσης σε μια τυπική UML. Για παράδειγμα, ο Mick είναι ένα παράδειγμα του The Rolling Stones.
- Έμμεσα, μέσω λογικών τελεστών στο «OntClass». Εάν κάποια «OntClass» είναι ένωση, διασταύρωση ή συμπλήρωμα, είναι μια κατηγορία Ατόμων που δεν ορίζονται ρητά ως παρουσίες της. Για παράδειγμα, αν και ο Mick δεν ορίζεται ρητά ως μέλος του StonesWailersUnion, αποτελεί μέλος του καθώς είναι μέλος του TheRollingStones, το οποίο συνδέεται με το StonesWailersUnion μέσω μιας σύνδεσης «unionOf».

Αν και υπάρχουν ορισμένα εργαλεία UML (Together, Visio) τα οποία επιτρέπουν τις σχέσεις μεταξύ μιας κλάσης UML και ενός αντικειμένου UML σε ένα ανάλογο διάγραμμα, πολλά δημοφιλή εργαλεία UML (π.χ. Rational Rose, Poseidon) δεν παρέχουν αυτή την υποστήριξη. Έχει θεωρηθεί ότι αυτό σχετίζεται στενά με την κατανόηση της UML (ως γραφικού συμβολισμού), για τη μοντελοποίηση και τη χρήση της με αντικειμενοστρεφείς γλώσσες προγραμματισμού. Ένα άλλο πολύ σημαντικό ζήτημα σχετίζεται με την αρχιτεκτονική μεταμοντελοποίησης MDA (Model Driven Architecture). Συνήθως, θεωρείται ότι οι κλάσεις UML ανήκουν στο επίπεδο μοντέλου. Αυτό βέβαια δεν είναι απόλυτα σωστό, διότι τόσο η κλάση UML όσο και το αντικείμενο, ορίζονται στο ίδιο επίπεδο MDA. Σύμφωνα με την έρευνα των Atkinson & Kühne (2002), ένα αντικείμενο UML έχει την ικανότητα να διαμορφώνει ένα αντικείμενο από τον πραγματικό κόσμο, ωστόσο, τα αντικείμενα που διαμορφώνουν αποκλειστικά και μόνο, πράγματα του πραγματικού κόσμου, δεν θεωρούνται πραγματικά πράγματα (π.χ. στην Εικόνα 2 το αντικείμενο Mick διαμορφώνει μόνο μια παρουσία του Human). Επομένως, προκύπτει το εξής ερώτημα: πώς μπορεί να γίνει μία σωστή διάκριση μεταξύ της σχέσης των αντικειμένων (αλλά και των κλάσεων) και της κλάσης UML (μετακλάση) και κάποιας συγκεκριμένης κλάσης; Οι συγγραφείς Atkinson & Kuhne (2003) κατάφεραν να δώσουν τη λύση σε αυτό το πρόβλημα εισάγοντας δύο είδη για περιπτώσεις σχέσεων: τη γλωσσική και την οντολογική. Το γλωσσικό παράδειγμα σχέσης είναι το παράδειγμα μεταξύ εννοιών από διαφορετικά επίπεδα (ορισμός κατηγορίας UML και κάποια συγκεκριμένη κατηγορία, για παράδειγμα The Wailers). Από την άλλη πλευρά, η οντολογική περίπτωση είναι η περίπτωση της σχέσης μεταξύ εννοιών οι οποίες αν και βρίσκονται στο ίδιο γλωσσικό επίπεδο, εντοπίζονται σε διαφορετικά οντολογικά επίπεδα.

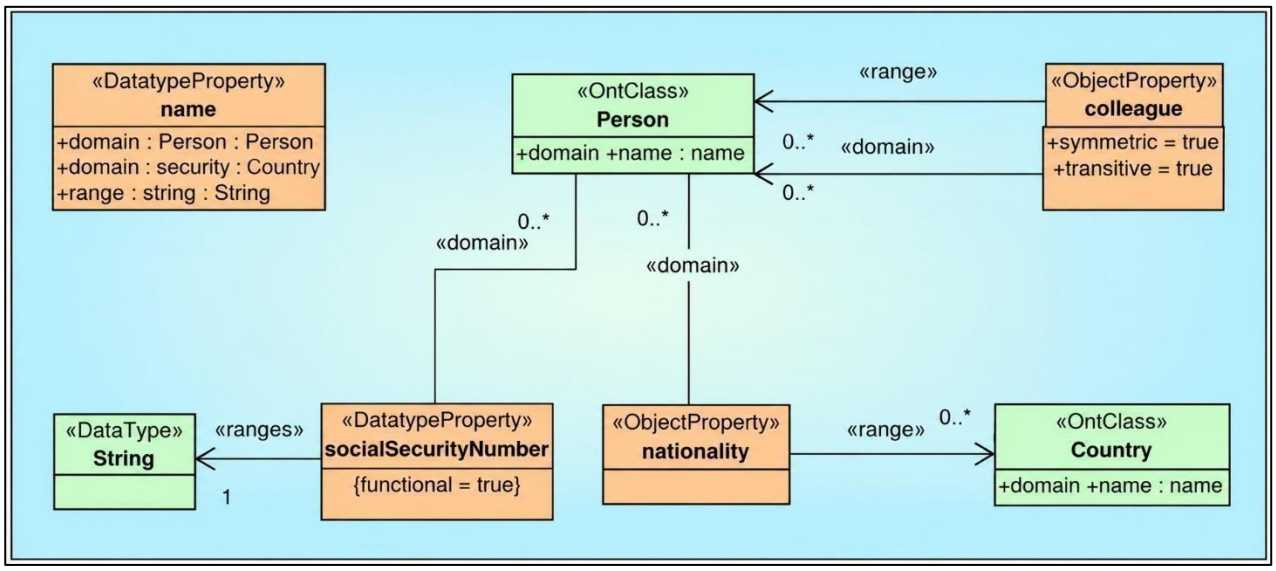
Η έννοια της ιδιότητας (Property) στις γλώσσες οντολογιών αποτελεί μία από τις δυσκολότερες έννοιες όσον αφορά τη μοντελοποίησή της με αντικειμενοστρεφείς γλώσσες και την UML. Το βασικό πρόβλημα προκύπτει από τη διαφορά μεταξύ της οντολογικής έννοιας «Property» και των συγγενών εννοιών της UML, δηλαδή των «Association» και «Attribute». Σε αντίθεση με αυτές, μια οντολογική «Property» αποτελεί ανεξάρτητη οντότητα και μπορεί να οριστεί χωρίς να εξαρτάται

από συγκεκριμένη κλάση. Οι Baclawski et al. (2001) πρότειναν την επέκταση της UML με νέες δομές για την υποστήριξη αυτής της αυτόνομης έννοιας, εισάγοντας παράλληλα έννοιες από τον προσανατολισμένο στις πτυχές προγραμματισμό. Ωστόσο, η προσέγγιση αυτή συνεπάγεται μη τυπικές επεκτάσεις της UML. Μία διαφορετική προσέγγιση, η οποία υιοθετείται στο Ontology UML Profile (OUP), προτείνει τη μοντελοποίηση της «Property» μέσω μιας UML κλάσης με το στερεότυπο «Property» (Djurić, 2004). Επειδή όμως μια ιδιότητα πρέπει να μπορεί να αναπαριστά σχέσεις μεταξύ πόρων, όπως κλάσεις και τύποι δεδομένων, η UML κλάση από μόνη της δεν επαρκεί. Για τον λόγο αυτό, σύμφωνα με το ODM και το OUP, οι σχέσεις αυτές αναπαρίστανται με τα στερεότυπα «domain» και «range» πάνω σε UML Associations ή Attributes. Για τη βελτίωση της αναγνωσιμότητας των διαγραμμάτων, η συσχέτιση «range» ορίζεται ως μονής κατεύθυνσης, από την ιδιότητα προς την αντίστοιχη κλάση (Djurić, 2004).

Ειδικότερα, το ODM ορίζει δύο τύπους (υποκατηγορίες) της Property: (α) την ObjectProperty και (β) την DatatypeProperty. Η ObjectProperty, η οποία μπορεί να έχει μόνο Individuals στο εύρος και τον τομέα της, αντιπροσωπεύεται στο OUP ως το στερεότυπο της κλάσης «ObjectProperty». Αντίστοιχα, η DatatypeProperty διαμορφώνεται με το στερεότυπο της κλάσης «DatatypeProperty». Ένα παράδειγμα διαγράμματος μιας τέτοιας κλάσης, το οποίο απεικονίζει τις ιδιότητες οντολογίας που έχουν διαμορφωθεί σε UML, φαίνεται στην Εικόνα 3. Αυτό περιέχει τέσσερις ιδιότητες: δύο κλάσεις UML με «DatatypeProperty» (όνομα/name και αριθμό Κοινωνικής Ασφάλισης/social Security Number) και δύο με «ObjectProperty» (εθνικότητα/nationality και συνεργάτης/colleague). Αυτές, σε συνεργασία με τις συσχετίσεις UML «domain» και «range», ή με τις ιδιότητες UML «domain» και «range», χρησιμοποιούνται για τη μοντελοποίηση των σχέσεων μεταξύ των κλάσεων UML «OntClass». Τέλος και σύμφωνα με τον Gamma (1995), οι επισυναπτόμενες τιμές περιγράφουν ορισμένα πρόσθετα χαρακτηριστικά, τα οποία προστίθενται στο ODM, εφαρμόζοντας το μοτίβο σχεδίασης.

Υπάρχει όμως ένα σημαντικό ζήτημα που πρέπει να διευκρινιστεί όσον αφορά το διάγραμμα που απεικονίζεται στην Εικόνα 3. Στην UML, οι σχέσεις αντιπροσωπεύονται είτε από συσχετίσεις (που απεικονίζονται γραφικά ως γραμμές) είτε από χαρακτηριστικά. Από την άλλη πλευρά, τα διαγράμματα ενός OUP μπορεί να φαίνονται υπερβολικά γεμάτα, διότι για να επιτευχθεί μια σωστή αναπαράσταση, κάθε σχέση απαιτεί ένα πλαίσιο και δύο γραμμές. Μία ενδεδειγμένη λύση είναι η χρήση τυπικών γραφικών συμβόλων, ωστόσο, η UML επιτρέπει μόνο προσαρμοσμένα γραφικά σύμβολα για ένα λειτουργικό πρότυπο UML. Για παράδειγμα, ένα προσαρμοσμένο γραφικό σύμβολο για την «Property», θα μπορούσε να είναι ένας μικροσκοπικός κύκλος με γραμμές (μειώνει τον χώρο στα διαγράμματα). Επίσης, μπορούν να χρησιμοποιηθούν πρόσθετες προσαρμοσμένες ρυθμίσεις, όπως διαφορετικά χρώματα για «OntClass» (πράσινο), «ObjectProperty» (πορτοκαλί) ή «DatatypeProperty» (πορτοκαλί), προκειμένου να αυξηθεί η αναγνωσιμότητα του διαγράμματος. Επομένως, για λόγους αναγνωσιμότητας, το συγκεκριμένο λειτουργικό πρότυπο επιτρέπει δύο είδη αναπαράστασης «DatatypeProperty» (για range και

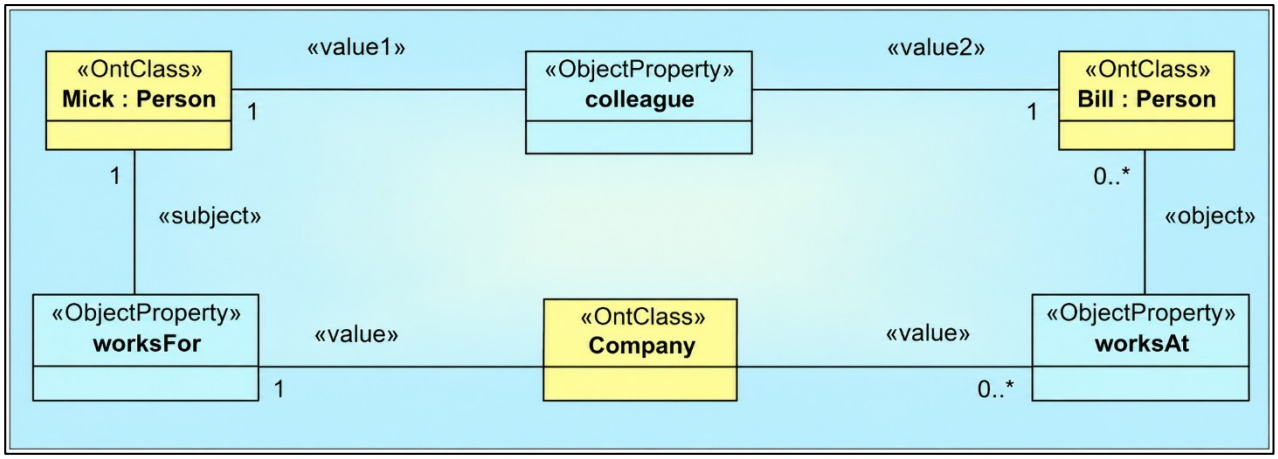
domain). Ένα παράδειγμα του πρώτου είδους (δηλαδή μια κλάση UML με δύο συσχετίσεις UML) είναι ο αριθμός Κοινωνικής Ασφάλισης, ενώ ένα παράδειγμα του δεύτερου (μια κλάση με χαρακτηριστικά range και domain) είναι το όνομα. Πρέπει ωστόσο να σημειωθεί, ότι το δεύτερο είδος επιτρέπεται μόνο για το «DatatypeProperty» του οποίου η πολλαπλότητα εύρους είναι ίση ή μικρότερη από 1.



Εικόνα 3: Διάγραμμα κλάσης UML που απεικονίζει ιδιότητες οντολογίας. (Djurić, 2004).

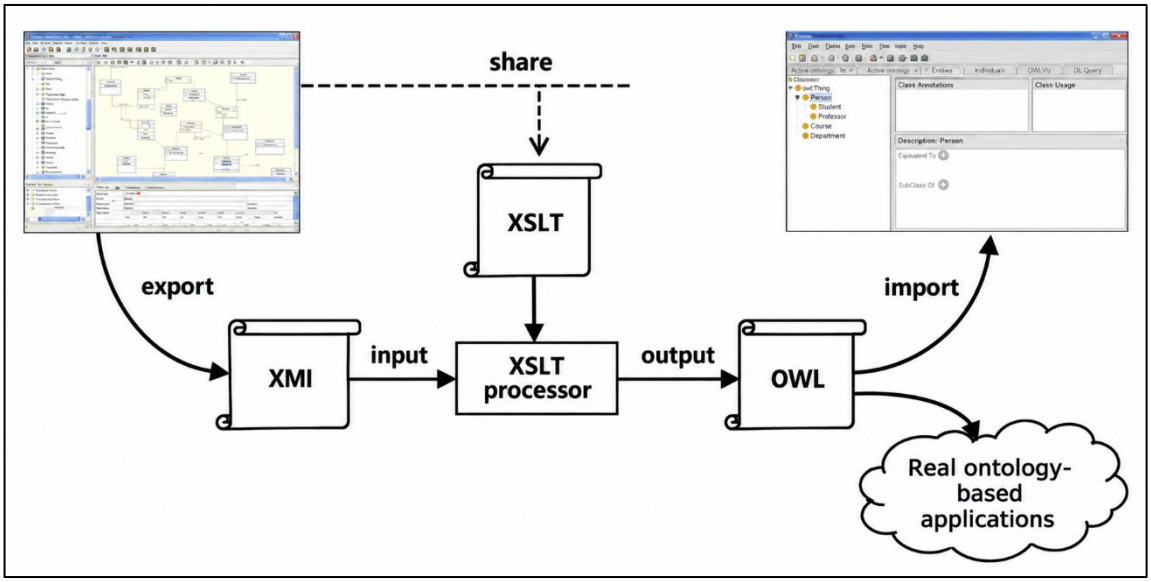
Στο πλαίσιο της παρούσας εργασίας, η UML εξετάζεται κυρίως ως γλώσσα γενικής μοντελοποίησης, η οποία επιτρέπει την οπτική αναπαράσταση της δομής ενός συστήματος μέσω εννοιών όπως οι κλάσεις, τα χαρακτηριστικά και οι συσχετίσεις. Η σημασία της για την παρούσα έρευνα έγκειται κυρίως στη δυνατότητα δομημένης και κατανοητής αποτύπωσης ενός εννοιολογικού μοντέλου. Ωστόσο, η UML δεν έχει σχεδιαστεί πρωτίστως ως γλώσσα αναπαράστασης γνώσης και αυτοματοποιημένης συλλογιστικής· για τον λόγο αυτό, η περαιτέρω ανάλυση μηχανισμών γεφύρωσής της με οντολογικά μοντέλα, όπως το OUP και το ODM, δεν αποτελεί βασικό αντικείμενο της παρούσας εργασίας.

Εν συντομία, στο OUP, η «Statement» αναπαρίσταται ως αντικείμενο με δύο συνδέσμους (subject Link και object Link,) οι οποίοι φαίνονται στην Εικόνα 4 παρακάτω. Τα πρόσωπα που αντιπροσωπεύονται (δηλαδή στο συγκεκριμένο παράδειγμα, ο Mick και ο Keith) είναι συνεργάτες (colleagues) και έχουν και οι δύο βρετανική υπηκοότητα. Όπως και με τις ιδιότητες οντολογίας, η αναγνωσιμότητα του διαγράμματος μπορεί να αυξηθεί περαιτέρω χρησιμοποιώντας διαφορετικά χρώματα και προσαρμοσμένα γραφικά σύμβολα. Επομένως, μπορεί να χρησιμοποιηθεί ένας μικροσκοπικός κύκλος (αντί για το τυπικό πλαίσιο) για την αναπαράσταση της «Statement», προκειμένου να μειωθεί η ασταθής συλλογή γραφικών οντοτήτων σε ένα διάγραμμα.



Εικόνα 4: Απεικόνιση Individuals και Statements σε ένα διάγραμμα αντικειμένων UML. (Djurić, 2004).

Στη βιβλιογραφία έχουν προταθεί τεχνικές για τη μετατροπή μοντέλων UML σε οντολογίες OWL (Gašević et al., 2004). Ωστόσο, η τεχνική υλοποίηση αυτών των μετασχηματισμών δεν αποτελεί αντικείμενο της παρούσας ενότητας, καθώς ο στόχος της είναι η παρουσίαση και σύγκριση των δύο γλωσσών ως προς τον σκοπό, τη μορφή αναπαράστασης και τις σημασιολογικές δυνατότητές τους. Στο πλαίσιο της παρούσας εργασίας, η UML εξετάζεται κυρίως ως γλώσσα οπτικής και δομικής μοντελοποίησης, ενώ η OWL ως γλώσσα τυπικής αναπαράστασης γνώσης, η οποία μπορεί να υποστηρίξει αυτοματοποιημένη συλλογιστική μέσω κατάλληλων μηχανισμών συμπερασμού.



Εικόνα 5: Χρησιμοποιημένη αρχή XSLT: επεκτάσεις εργαλείων UML για την ανάπτυξη της οντολογίας. Πηγή: (Djurić, 2004).

Τα OUP και ODM που αναφέρθηκαν προηγουμένως συνδέονται με τις προσπάθειες του Object Management Group (OMG) για την τυποποιημένη μοντελοποίηση οντολογιών. Η συμβατότητά τους με το Meta-Object Facility (MOF) επιτρέπει την ενσωμάτωση οντολογικών μοντέλων σε

ευρύτερα περιβάλλοντα μοντελοποίησης και την ανταλλαγή τους μέσω τυποποιημένων μορφών, όπως το XMI. Η προσέγγιση αυτή ανέδειξε τη δυνατότητα σύνδεσης της UML με οντολογικές αναπαραστάσεις, χωρίς όμως η περαιτέρω τεχνική ανάλυση του OUP, του ODM ή των σχετικών μηχανισμών μεταμοντελοποίησης να αποτελεί αντικείμενο της παρούσας εργασίας.

Η UML εντάσσεται στο οικοσύστημα μεταμοντελοποίησης του Object Management Group (OMG) και συνδέεται με το Meta Object Facility (MOF), πάνω στο οποίο βασίζεται ο ορισμός μεταμοντέλων όπως η UML. Για την ακριβέστερη διατύπωση περιορισμών στα UML μοντέλα χρησιμοποιείται συμπληρωματικά η Object Constraint Language (OCL), η οποία ορίζεται σε ξεχωριστή προδιαγραφή του OMG. Στην παρούσα εργασία η UML εξετάζεται κυρίως ως ημιτυπική γλώσσα για την οπτική και δομική μοντελοποίηση συστημάτων.

2.1.3 Τυπικές (Formal)

Οι τυπικές γλώσσες διαφοροποιούνται από τις ημιδομημένες και τις ημιτυπικές γλώσσες επειδή διαθέτουν αυστηρά καθορισμένη σύνταξη και, κυρίως, τυπικά ορισμένη σημασιολογία. Η σημασία των εκφράσεών τους δεν εξαρτάται αποκλειστικά από την ερμηνεία του αναγνώστη, αλλά προσδιορίζεται μέσω καθορισμένων κανόνων. Το χαρακτηριστικό αυτό καθιστά δυνατή την αυτοματοποιημένη επεξεργασία της γνώσης, τον έλεγχο της λογικής συνέπειας ενός μοντέλου και την εξαγωγή συμπερασμάτων που προκύπτουν από τις ήδη δηλωμένες πληροφορίες.

Στον χώρο της αναπαράστασης γνώσης, χαρακτηριστική εφαρμογή των τυπικών γλωσσών αποτελούν οι οντολογίες. Σύμφωνα με τον Gruber (1995), μια οντολογία αποτελεί ρητή προδιαγραφή μιας εννοιοποίησης. Στην πράξη, μια οντολογία προσδιορίζει τις βασικές έννοιες ενός πεδίου, τις μεταξύ τους σχέσεις και τους περιορισμούς που διέπουν τη χρήση τους. Με τον τρόπο αυτό δεν περιορίζεται στην απλή αποθήκευση δεδομένων, αλλά παρέχει ένα κοινό σημασιολογικό μοντέλο μέσω του οποίου διαφορετικές πληροφορίες μπορούν να ερμηνεύονται με συνεπή τρόπο.

Μια οντολογία περιλαμβάνει συνήθως κλάσεις (classes), οι οποίες αναπαριστούν κατηγορίες εννοιών, άτομα (individuals), τα οποία αναπαριστούν συγκεκριμένες οντότητες, και ιδιότητες (properties), οι οποίες εκφράζουν σχέσεις μεταξύ οντοτήτων ή συνδέουν μία οντότητα με μία τιμή δεδομένων. Επιπλέον, μπορούν να οριστούν αξιώματα και περιορισμοί που προσδιορίζουν περαιτέρω τη σημασιολογία των παραπάνω στοιχείων. Η δυνατότητα αυτή είναι ιδιαίτερα σημαντική, καθώς επιτρέπει την αξιοποίηση όχι μόνο των αποθηκευμένων τιμών, αλλά και των σχέσεων και των κανόνων που συνδέουν τις έννοιες μεταξύ τους.

2.1.3.1 RDF και RDFS

Το Resource Description Framework (RDF) αποτελεί βασικό μοντέλο αναπαράστασης δεδομένων στον Σημασιολογικό Ιστό. Η πληροφορία στο RDF οργανώνεται σε τριπλέτες της μορφής υποκείμενο–κατηγόρημα–αντικείμενο (subject–predicate–object), οι οποίες συνολικά σχηματίζουν έναν γράφο. Με αυτόν τον τρόπο, οι οντότητες μπορούν να συνδέονται μεταξύ τους μέσω ρητά

καθορισμένων σχέσεων και να αναγνωρίζονται μέσω μοναδικών αναγνωριστικών (Beckett & McBride, 2004).

Το RDF Schema (RDFS) επεκτείνει το RDF παρέχοντας μηχανισμούς για την περιγραφή κλάσεων, ιδιοτήτων και ιεραρχικών σχέσεων. Για παράδειγμα, μέσω της σχέσης rdfs:subClassOf μπορεί να δηλωθεί ότι μία κλάση αποτελεί εξειδίκευση μιας άλλης. Η δυνατότητα αυτή σχετίζεται άμεσα με την παρούσα εργασία, καθώς τα διαφορετικά είδη μοτίβων ασφάλειας μπορούν να οργανωθούν σε ιεραρχία κάτω από τη γενική έννοια SecurityPattern.

Παρότι το RDF και το RDFS παρέχουν τη βάση για τη δομημένη και σημασιολογικά συσχετισμένη αναπαράσταση δεδομένων, η εκφραστικότητά τους είναι περιορισμένη όταν απαιτείται η περιγραφή περισσότερο σύνθετων περιορισμών και λογικών σχέσεων. Για τον σκοπό αυτό χρησιμοποιούνται γλώσσες οντολογιών με ισχυρότερη τυπική σημασιολογία, όπως η Web Ontology Language (OWL).

2.1.3.2 Web Ontology Language – OWL

Η Web Ontology Language (OWL) είναι γλώσσα αναπαράστασης οντολογιών του World Wide Web Consortium (W3C), σχεδιασμένη για την τυπική περιγραφή γνώσης. Η OWL 2 επιτρέπει τον ορισμό κλάσεων, individuals, ιδιοτήτων και αξιωμάτων που περιγράφουν τις μεταξύ τους σημασιολογικές σχέσεις (W3C OWL Working Group, 2012).

Σε αντίθεση με μια απλή ιεραρχική ταξινόμηση, η OWL επιτρέπει την έκφραση πλουσιότερων σχέσεων και περιορισμών. Είναι δυνατό, για παράδειγμα, να δηλωθεί ότι μία κλάση αποτελεί υποκλάση μιας άλλης, ότι δύο κλάσεις είναι ισοδύναμες ή ασύμβατες, καθώς και ότι μία ιδιότητα συνδέει οντότητες συγκεκριμένων κατηγοριών. Η ύπαρξη τυπικά καθορισμένης σημασιολογίας επιτρέπει την επεξεργασία μιας OWL οντολογίας από κατάλληλους μηχανισμούς συλλογιστικής.

Η OWL διαφέρει ως προς τον σκοπό της από τις γλώσσες που παρουσιάστηκαν προηγουμένως. Η XML παρέχει κυρίως έναν τυποποιημένο τρόπο δομημένης αναπαράστασης και ανταλλαγής δεδομένων, ενώ η UML χρησιμοποιείται κυρίως για τη μοντελοποίηση της δομής και της συμπεριφοράς συστημάτων. Η OWL έχει σχεδιαστεί ειδικά για την αναπαράσταση γνώσης με τυπικά καθορισμένη σημασιολογία. Η ιδιότητα αυτή είναι καθοριστική σε εφαρμογές στις οποίες δεν αρκεί η αποθήκευση πληροφοριών, αλλά απαιτείται και η αξιοποίηση των σημασιολογικών σχέσεων που αναπτύσσονται μεταξύ τους.

2.1.3.3 Συλλογιστική και έλεγχος συνέπειας

Η τυπική σημασιολογία της OWL καθιστά δυνατή τη χρήση μηχανισμών συλλογιστικής (reasoners). Ένας reasoner επεξεργάζεται τα αξιώματα μιας οντολογίας και μπορεί να εξαγάγει συμπεράσματα τα οποία δεν έχουν δηλωθεί ρητά αλλά προκύπτουν λογικά από το μοντέλο.

Για παράδειγμα, εάν η κλάση AuthenticationPattern έχει οριστεί ως υποκλάση της SecurityPattern και ένα individual ανήκει στην AuthenticationPattern, τότε μπορεί να συναχθεί ότι το ίδιο individual αποτελεί επίσης SecurityPattern. Δεν απαιτείται επομένως η ανεξάρτητη δήλωση και των δύο σχέσεων (ότι το individual δηλαδή ανήκει και στις δυο κλάσεις).

Οι reasoners μπορούν επίσης να χρησιμοποιηθούν για τον έλεγχο της λογικής συνέπειας μιας οντολογίας. Ο HermiT αποτελεί reasoner για OWL 2 και υποστηρίζει, μεταξύ άλλων, έλεγχο συνέπειας και ταξινόμηση οντολογιών (Glimm et al., 2014). Στην παρούσα εργασία χρησιμοποιήθηκε μέσω του Protégé για τον έλεγχο της οντολογίας πριν από την αξιοποίησή της από την εφαρμογή.

Ο έλεγχος της λογικής συνέπειας δεν πρέπει να συγχέεται με την πληρότητα ή την πραγματική ορθότητα της γνώσης. Ένας reasoner μπορεί να εξετάσει εάν τα δηλωμένα αξιώματα είναι μεταξύ τους συνεπή και να εξαγάγει τις λογικές συνέπειές τους, αλλά δεν μπορεί από μόνος του να επιβεβαιώσει ότι όλες οι απαιτούμενες πληροφορίες έχουν καταγραφεί ή ότι οι καταγεγραμμένες πληροφορίες ανταποκρίνονται πλήρως στον πραγματικό κόσμο.

2.1.3.4 SPARQL και ανάκτηση σημασιολογικής πληροφορίας

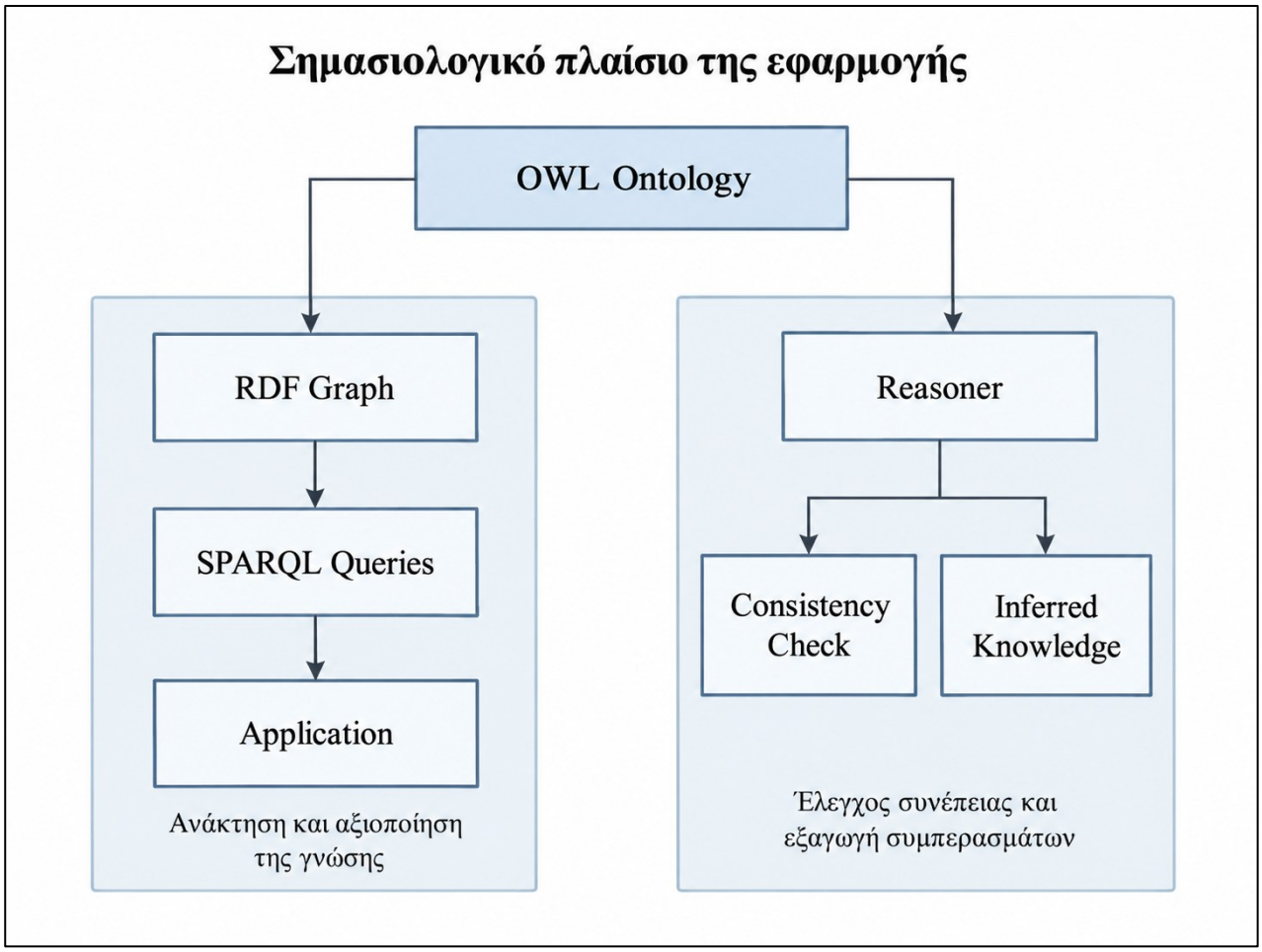
Για την ανάκτηση πληροφορίας από RDF γράφους χρησιμοποιείται η SPARQL. Η SPARQL αποτελεί γλώσσα επερωτήσεων για RDF και επιτρέπει την αναζήτηση πόρων και σχέσεων που ικανοποιούν συγκεκριμένα πρότυπα γράφου. Είναι σημαντικό να διακρίνεται από την OWL και από τους reasoners: η OWL χρησιμοποιείται για τη σημασιολογική περιγραφή της γνώσης, οι reasoners ενδέχεται να εμπλουτίζουν τη γνώση αυτή, ενώ η SPARQL χρησιμοποιείται για την αναζήτηση και ανάκτησή της.

Η διάκριση αυτή αποτυπώνεται και στην παρούσα εργασία. Η γνώση των μοτίβων ασφάλειας οργανώνεται μέσω μιας OWL οντολογίας, η οποία κατά τη λειτουργία της εφαρμογής φορτώνεται ως RDF γράφος. Η εφαρμογή χρησιμοποιεί SPARQL επερωτήσεις για την ανάκτηση των μοτίβων, των κατηγοριών και των σημασιολογικών τους σχέσεων. Αντίθετα, ο λογικός έλεγχος της οντολογίας πραγματοποιήθηκε ανεξάρτητα μέσω reasoner κατά τη διαδικασία ανάπτυξης και επικύρωσής της.

2.1.3.5 Σημασία των τυπικών γλωσσών για την παρούσα εργασία

Η χρήση τυπικής οντολογικής αναπαράστασης είναι ιδιαίτερα κατάλληλη για τον σκοπό της παρούσας εργασίας, επειδή τα μοτίβα ασφάλειας δεν αντιμετωπίζονται ως ανεξάρτητες περιγραφές κειμένου, αλλά ως συνδεδεμένες μονάδες γνώσης. Ένα μοτίβο μπορεί να ανήκει σε μία ή περισσότερες κατηγορίες, να μετριάζει συγκεκριμένες απειλές, να προστατεύει συγκεκριμένα αγαθά, να υποστηρίζει ιδιότητες ασφάλειας, να εφαρμόζεται σε συγκεκριμένες φάσεις ανάπτυξης και να σχετίζεται με άλλα μοτίβα ή βιβλιογραφικές πηγές.

Η αναπαράσταση των παραπάνω αποκλειστικά μέσω ελεύθερου κειμένου ή απλών πεδίων δεδομένων θα επέτρεπε την αποθήκευσή τους, αλλά δεν θα απέδιδε με τον ίδιο τρόπο τις μεταξύ τους σημασιολογικές σχέσεις. Για τον λόγο αυτό, η OWL επιλέχθηκε ως βάση του σημασιολογικού μοντέλου της εφαρμογής. Η επιλογή αυτή επιτρέπει την αναπαράσταση των security patterns ως οντοτήτων ενός ενιαίου δικτύου γνώσης και παρέχει τη βάση για ιεραρχική ταξινόμηση, σημασιολογικές επερωτήσεις και μελλοντική αξιοποίηση ισχυρότερων μηχανισμών reasoning και validation.



Εικόνα 6: Σχέση μεταξύ OWL, RDF, SPARQL και μηχανισμών συλλογιστικής στο σημασιολογικό πλαίσιο της παρούσας εργασίας.

2.1.3.6 Οντολογίες Ειδικού Τομέα στην Ασφάλεια

Στο πεδίο της αναπαράστασης γνώσης, μια οντολογία μπορεί να οριστεί ως ρητή προδιαγραφή μιας εννοιοποίησης (Gruber, 1995). Μέσω μιας οντολογίας προσδιορίζονται οι βασικές έννοιες ενός συγκεκριμένου πεδίου, οι μεταξύ τους σχέσεις και, όπου απαιτείται, οι περιορισμοί που διέπουν τη χρήση τους. Με τον τρόπο αυτό παρέχεται ένα κοινό σημασιολογικό πλαίσιο για την οργανωμένη αναπαράσταση, διασύνδεση και επαναχρησιμοποίηση της γνώσης ενός πεδίου.

Η ανάγκη για αντίστοιχη σημασιολογική οργάνωση έχει αναγνωριστεί ιδιαίτερα στον τομέα της ασφάλειας πληροφοριών, όπου διαφορετικές πηγές και οργανισμοί ενδέχεται να χρησιμοποιούν

διαφορετικούς όρους για συναφείς έννοιες. Ο Donner (2003) επισήμανε ότι η ασάφεια και η έλλειψη κοινά αποδεκτής ορολογίας δυσχεραίνουν την επικοινωνία μεταξύ των επαγγελματιών ασφάλειας και υποστήριξε την ανάπτυξη οντολογιών που θα περιγράφουν με σαφήνεια τις σημαντικότερες έννοιες του πεδίου και τις μεταξύ τους σχέσεις. Μια κοινή οντολογική αναπαράσταση μπορεί έτσι να διευκολύνει την ανταλλαγή πληροφοριών, τη συστηματική καταγραφή περιστατικών και τη συνεπέστερη επικοινωνία μεταξύ διαφορετικών οργανισμών και πληροφοριακών συστημάτων.

Οι Raskin et al. (2001) υποστηρίζουν επίσης ότι η οντολογική προσέγγιση μπορεί να συμβάλει στη συστηματοποίηση της γνώσης ασφάλειας, στην οργάνωσή της σε επαναχρησιμοποιήσιμες ενότητες και στην αναγνώριση νέων συσχετίσεων μεταξύ των εννοιών της. Η αξία μιας οντολογίας δεν περιορίζεται επομένως στη δημιουργία μιας ταξινομίας, αλλά επεκτείνεται στη ρητή αναπαράσταση ιδιοτήτων και σχέσεων, ώστε η διαθέσιμη γνώση να μπορεί να υποβληθεί σε μηχανική επεξεργασία και, όπου το οντολογικό μοντέλο το επιτρέπει, σε διαδικασίες συλλογιστικής.

Για παράδειγμα, σε μια οντολογία ασφάλειας μπορεί να δηλωθεί ότι μια επίθεση άρνησης υπηρεσίας (Denial of Service) αποτελεί τύπο απειλής που επηρεάζει τη Διαθεσιμότητα (Availability), καθώς και ότι συγκεκριμένα αντίμετρα αντιμετωπίζουν ή μετριάζουν τη συγκεκριμένη απειλή. Μέσω των ρητών αυτών σχέσεων, ένα σύστημα μπορεί να συσχετίζει διαφορετικές έννοιες του πεδίου και να ανακτά γνώση με βάση τη σημασία και τις μεταξύ τους σχέσεις και όχι αποκλειστικά μέσω λεκτικής αντιστοίχισης.

Στη βιβλιογραφία έχουν προταθεί τόσο γενικές όσο και περισσότερο εξειδικευμένες οντολογίες ασφάλειας. Οι Herzog et al. (2007) ανέπτυξαν μια OWL οντολογία γενικού σκοπού για την ασφάλεια πληροφοριών, η οποία αναπαριστά έννοιες όπως αγαθά (assets), απειλές (threats), ευπάθειες (vulnerabilities), αντίμετρα (countermeasures) και τις μεταξύ τους σχέσεις. Η οντολογία σχεδιάστηκε ώστε να λειτουργεί ως κοινό λεξιλόγιο και επεκτάσιμη εννοιολογική βάση για το πεδίο της ασφάλειας πληροφοριών, υποστηρίζοντας παράλληλα την εξαγωγή συμπερασμάτων σχετικά με τις σχέσεις μεταξύ των διαφορετικών εννοιών.

Παρόμοια κατεύθυνση ακολούθησαν οι Fenz και Ekelhart (2009), οι οποίοι πρότειναν ένα τυπικό οντολογικό μοντέλο γνώσης για την ασφάλεια πληροφοριών. Η προσέγγισή τους συνδυάζει γενική γνώση ασφάλειας, καθιερωμένες πρακτικές και πληροφορίες που αφορούν τον ίδιο τον οργανισμό, με στόχο την υποστήριξη διαδικασιών διαχείρισης και αξιολόγησης κινδύνου. Οι δύο αυτές προσεγγίσεις καταδεικνύουν ότι οι οντολογίες μπορούν να λειτουργήσουν ως κοινή σημασιολογική βάση για τη σύνδεση αγαθών, απειλών, ευπαθειών, απαιτήσεων προστασίας και αντίμετρων.

Οι Blanco et al. (2008) πραγματοποίησαν συστηματική ανασκόπηση και σύγκριση υφιστάμενων οντολογιών ασφάλειας. Η μελέτη τους ανέδειξε ότι οι σχετικές προσεγγίσεις βρίσκονταν ακόμη σε σχετικά πρώιμο στάδιο ανάπτυξης και παρουσίαζαν σημαντικές

διαφοροποιήσεις ως προς το πεδίο εφαρμογής και τις έννοιες που κάλυπταν. Το αποτέλεσμα αυτό υπογράμμισε την ανάγκη για περαιτέρω έρευνα και περισσότερο ολοκληρωμένες και συστηματικές οντολογικές προσεγγίσεις στον χώρο της ασφάλειας πληροφοριών.

Ειδικότερα ως προς τις απαιτήσεις ασφάλειας, οι Souag et al. (2012) πραγματοποίησαν βιβλιογραφική ανασκόπηση και ταξινόμησαν τις υφιστάμενες οντολογικές προσεγγίσεις σε οκτώ οικογένειες: αρχικές οντολογίες ασφάλειας, ταξινομίες ασφάλειας, γενικές οντολογίες ασφάλειας, εξειδικευμένες οντολογίες ασφάλειας, οντολογίες βασισμένες στον κίνδυνο, οντολογίες προσανατολισμένες στον Ιστό, οντολογίες για απαιτήσεις ασφάλειας και οντολογίες μοντελοποίησης. Η σύγκρισή τους έδειξε ότι οι υπάρχουσες οντολογίες διαφέρουν σημαντικά ως προς τον βαθμό στον οποίο καλύπτουν έννοιες όπως στόχοι και ιδιότητες ασφάλειας, αγαθά, ευπάθειες, απειλές, αντίμετρα, οργανωτικές έννοιες και απαιτήσεις ασφάλειας.

Η μοντελοποίηση των απαιτήσεων ασφάλειας παρουσιάζει ιδιαίτερο ενδιαφέρον και για την αξιοποίηση επαναχρησιμοποιήσιμης γνώσης ασφάλειας. Οι απαιτήσεις προσδιορίζουν τις ιδιότητες και τους στόχους ασφάλειας που πρέπει να ικανοποιούνται, χωρίς να καθορίζουν κατ' ανάγκη τη σχεδιαστική ή υλοποιητική λύση. Η σημασιολογική αναπαράστασή τους μπορεί επομένως να διευκολύνει τη μετέπειτα σύνδεσή τους με κατάλληλα αντίμετρα, τεχνικές ή επαναχρησιμοποιήσιμες σχεδιαστικές λύσεις, όπως τα μοτίβα ασφάλειας.

Οι Tsoumas και Gritzalis (2006) εξέτασαν την αξιοποίηση οντολογιών στο πλαίσιο της διαχείρισης ασφάλειας πληροφοριακών συστημάτων. Η προσέγγισή τους βασίζεται στην ύπαρξη επαναχρησιμοποιήσιμης γνώσης ασφάλειας και στη σημασιολογική συσχέτιση πληροφοριών που προέρχονται από διαφορετικές πηγές. Παράλληλα, διαχωρίζει τις απαιτήσεις ασφάλειας από τις συγκεκριμένες τεχνικές υλοποίησής τους, επιτρέποντας τη διαχείριση της γνώσης σε υψηλότερο επίπεδο αφαίρεσης.

Οι Dobson και Sawyer (2006) εξέτασαν τη χρήση οντολογιών στη μηχανική απαιτήσεων και ειδικότερα στο ευρύτερο πεδίο της αξιοπιστίας (dependability). Η έννοια της dependability δεν περιορίζεται αποκλειστικά στην ασφάλεια, αλλά περιλαμβάνει ιδιότητες όπως διαθεσιμότητα, αξιοπιστία, ασφάλεια έναντι ατυχημάτων (προστασία), ακεραιότητα, συντηρησιμότητα και εμπιστευτικότητα. Η προσέγγιση είναι σχετική με την ασφάλεια λογισμικού επειδή καταδεικνύει πώς διαφορετικές μη λειτουργικές απαιτήσεις και οι μεταξύ τους σχέσεις μπορούν να αποτυπωθούν σε ένα κοινό, μηχανικά επεξεργάσιμο σημασιολογικό μοντέλο.

Σε περισσότερο εξειδικευμένο πεδίο, οι Denker et al. (2005) μελέτησαν την εφαρμογή οντολογικών τεχνολογιών στην ασφάλεια του Σημασιολογικού Ιστού. Η προσέγγισή τους χρησιμοποιεί την OWL για την αναπαράσταση εννοιών σχετικών με την ασφάλεια και την εμπιστοσύνη, καθώς και πολιτικών που μπορούν να αξιοποιηθούν από μηχανισμούς συλλογιστικής. Ιδιαίτερο ενδιαφέρον παρουσιάζει η χρήση σημασιολογικών περιγραφών για πόρους Ιστού και υπηρεσίες Ιστού, ώστε αποφάσεις όπως ο έλεγχος πρόσβασης ή η επιλογή υπηρεσιών βάσει περιορισμών ασφάλειας να μπορούν να υποστηρίζονται από μηχανικά επεξεργάσιμη γνώση.

Η εξειδίκευση των οντολογιών ασφάλειας εμφανίζεται και σε νεότερα πεδία, όπως η ευφυΐα κυβερνοαπειλών (Cyber Threat Intelligence – CTI). Οι Mavroeidis και Bromander (2017) αξιολόγησαν υπάρχουσες ταξινομίες, πρότυπα ανταλλαγής και οντολογικές προσεγγίσεις για την αναπαράσταση πληροφοριών κυβερνοαπειλών. Η μελέτη τους ανέδειξε την έλλειψη μιας ολοκληρωμένης και διαλειτουργικής οντολογίας που να καλύπτει με επαρκή σημασιολογική ακρίβεια ολόκληρο το φάσμα της σχετικής γνώσης, όπως τους εμπλεκόμενους φορείς, τα κίνητρα, τις τεχνικές, τους δείκτες και τις ενέργειες αντιμετώπισης.

Οι παραπάνω προσεγγίσεις δείχνουν ότι δεν υπάρχει μία ενιαία οντολογία κατάλληλη για κάθε πρόβλημα ασφάλειας. Αντίθετα, το οντολογικό μοντέλο διαμορφώνεται ανάλογα με τον σκοπό και το πεδίο εφαρμογής: γενική αναπαράσταση γνώσης ασφάλειας, αξιολόγηση κινδύνου, απαιτήσεις ασφάλειας, dependability, ασφάλεια υπηρεσιών Ιστού ή ευφυΐα κυβερνοαπειλών. Παρά τις διαφορές τους, οι προσεγγίσεις αυτές συγκλίνουν στη χρήση ρητά ορισμένων εννοιών και σημασιολογικών σχέσεων για τη συστηματική οργάνωση και επαναχρησιμοποίηση γνώσης ασφάλειας.

Οι γενικές αυτές οντολογίες παρέχουν επίσης σημαντικές έννοιες που μπορούν να αξιοποιηθούν σε περισσότερο εξειδικευμένα μοντέλα. Έννοιες όπως απειλές, ευπάθειες, προστατευόμενα αγαθά, ιδιότητες και απαιτήσεις ασφάλειας, αντίμετρα και σχέσεις μεταξύ αυτών είναι άμεσα σχετικές και με τη σημασιολογική περιγραφή των μοτίβων ασφάλειας. Οι οντολογικές προσεγγίσεις που επικεντρώνονται ειδικά στη μοντελοποίηση, οργάνωση και αναζήτηση μοτίβων ασφάλειας εξετάζονται στη συνέχεια στις σχετικές ερευνητικές εργασίες.

2.1.4 Σύγκριση γλωσσών (UML και OWL)

Έχοντας παρουσιάσει προηγουμένως την UML ως ημι-τυπική γλώσσα μοντελοποίησης και την OWL ως τυπική γλώσσα αναπαράστασης γνώσης, στην παρούσα ενότητα εξετάζονται οι βασικές διαφορές τους ως προς τον σκοπό, τη μορφή αναπαράστασης και τη σημασιολογία τους. Παρότι και οι δύο γλώσσες μπορούν να χρησιμοποιηθούν για την περιγραφή εννοιών και σχέσεων ενός πεδίου, έχουν αναπτυχθεί για διαφορετικούς σκοπούς και παρέχουν διαφορετικές δυνατότητες. Η σύγκριση είναι σημαντική για την παρούσα εργασία, καθώς επιτρέπει να αιτιολογηθεί η επιλογή της OWL ως βασικής γλώσσας αναπαράστασης της γνώσης των μοτίβων ασφάλειας.

2.1.5 Σύγκριση Γλωσσικής Αναπαράστασης και Σκοπού

2.1.5.1 Η Συγκεκριμένη Σύνταξη (Concrete Syntax)

Το συγκεκριμένο συντακτικό (concrete syntax) καθορίζει τον τρόπο με τον οποίο εκφράζονται και παρουσιάζονται οι κατασκευές μιας γλώσσας. Η UML χρησιμοποιεί κυρίως διαγραμματική σημειογραφία, όπως διαγράμματα κλάσεων, αντικειμένων και αλληλεπιδράσεων, ενώ τα UML μοντέλα μπορούν επίσης να ανταλλάσσονται σε μορφή XML μέσω του προτύπου XMI. Αντίθετα,

οι οντολογίες OWL μπορούν να σειριοποιηθούν σε διαφορετικές μορφές, όπως RDF/XML και άλλες συντακτικές αναπαραστάσεις του RDF (Beckett & McBride, 2004).

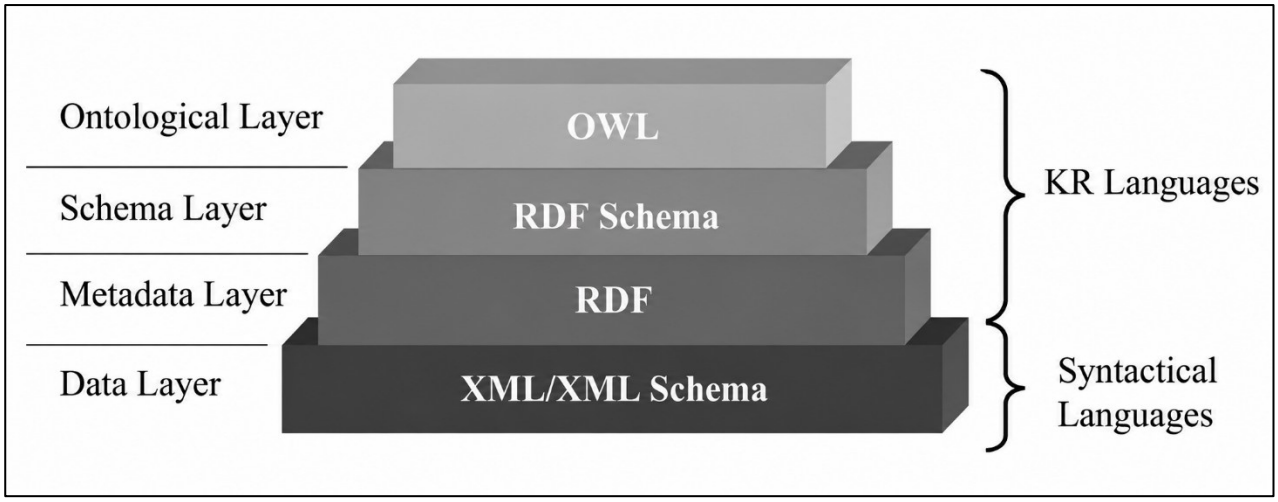
Η διαφορετική συντακτική μορφή δεν συνεπάγεται από μόνη της διαφορά στην εκφραστικότητα των δύο γλωσσών. Ένα μοντέλο μπορεί να διαθέτει διαφορετικές συντακτικές αναπαραστάσεις χωρίς να μεταβάλλεται η σημασία του. Για τον λόγο αυτό, η ουσιαστικότερη διαφορά μεταξύ UML και OWL δεν εντοπίζεται στον τρόπο με τον οποίο παρουσιάζονται οι πληροφορίες, αλλά στη σημασιολογία που αποδίδεται στις αντίστοιχες γλωσσικές κατασκευές.

Η UML δίνει ιδιαίτερη έμφαση στην οπτική αναπαράσταση. Τα διαγράμματά της είναι κατάλληλα για την επικοινωνία της δομής και της συμπεριφοράς ενός συστήματος μεταξύ αναλυτών, σχεδιαστών και προγραμματιστών. Η OWL, αντίθετα, δεν έχει σχεδιαστεί πρωτίστως ως οπτική γλώσσα. Στόχος της είναι η αναπαράσταση γνώσης σε μορφή που μπορεί να υποβληθεί σε τυπική επεξεργασία από λογισμικό.

2.1.5.2 Ο Σημασιολογικός Τομέας (The Semantic Domain)

Οι δύο γλώσσες έχουν διαφορετικό βασικό προσανατολισμό. Η UML αναπτύχθηκε κυρίως για τη μοντελοποίηση συστημάτων λογισμικού και χρησιμοποιείται για την αποτύπωση της δομής, της συμπεριφοράς και των αλληλεπιδράσεών τους. Η OWL, αντίθετα, έχει σχεδιαστεί για την τυπική αναπαράσταση γνώσης σχετικά με έναν τομέα και τις σχέσεις που αναπτύσσονται μεταξύ των εννοιών του.

Παρά τον διαφορετικό τους σκοπό, υπάρχει σημαντική εννοιολογική επικάλυψη. Και στις δύο περιπτώσεις μπορούν να περιγραφούν κατηγορίες οντοτήτων, συγκεκριμένες παρουσίες τους και σχέσεις μεταξύ αυτών. Για παράδειγμα, μία UML κλάση και μία OWL κλάση μπορούν σε υψηλό επίπεδο να χρησιμοποιηθούν για την αναπαράσταση μιας κατηγορίας αντικειμένων, ενώ οι συσχετίσεις της UML και οι ιδιότητες της OWL μπορούν να χρησιμοποιηθούν για την αναπαράσταση σχέσεων. Η ομοιότητα αυτή έχει οδηγήσει σε αρκετές μελέτες που εξετάζουν τη σχέση μεταξύ UML και οντολογικών γλωσσών (Calì et al., 2002· Kiko & Atkinson, 2008).



Εικόνα 7: Πλαίσιο τεσσάρων επιπέδων των τεχνολογιών XML, RDF, RDFS και OWL. Προσαρμογή από Kiko & Atkinson (2008, Fig. 2, p. 7)

Ωστόσο, οι αντίστοιχες έννοιες δεν είναι σημασιολογικά ισοδύναμες. Στην UML, μια κλάση αποτελεί κυρίως στοιχείο μοντελοποίησης ενός συστήματος. Στην OWL, μια κλάση ερμηνεύεται ως σύνολο individuals μέσα σε ένα τυπικά καθορισμένο σημασιολογικό μοντέλο. Αντίστοιχα, οι ιδιότητες της OWL αποτελούν ανεξάρτητες σημασιολογικές σχέσεις και μπορούν να συμμετέχουν σε αξιώματα και περιορισμούς που αξιοποιούνται από μηχανισμούς συλλογιστικής.

Και οι δύο γλώσσες μπορούν να περιγράψουν τόσο γενικές έννοιες όσο και συγκεκριμένες παρουσίες τους. Ως ορολογική ή εντατική γνώση (terminological/intensional knowledge) νοούνται οι γενικές έννοιες, οι κατηγορίες, οι ιδιότητες και οι περιορισμοί που ισχύουν για αυτές. Αντίθετα, ως ισχυριστική ή εκτατική γνώση (assertional/extensional knowledge) νοούνται οι δηλώσεις που αφορούν συγκεκριμένα αντικείμενα ή individuals και τις μεταξύ τους σχέσεις.

Στην OWL οι δύο μορφές γνώσης μπορούν να συνυπάρχουν στην ίδια οντολογία. Για παράδειγμα, μπορεί να ορίζεται ότι η AuthenticationPattern είναι υποκλάση της SecurityPattern και, παράλληλα, ότι ένα συγκεκριμένο pattern αποτελεί individual της AuthenticationPattern. Η ίδια γενική διάκριση μπορεί να αποτυπωθεί στην UML μέσω κλάσεων και αντικειμένων. Ωστόσο, η OWL αποδίδει στις δηλώσεις αυτές τυπική λογική σημασιολογία, γεγονός που επιτρέπει την αξιοποίησή τους από μηχανισμούς αυτοματοποιημένης συλλογιστικής.

2.1.5.3 Σύγκριση Ερμηνευτικών Υποθέσεων

Οι κοινότητες αναπαράστασης γνώσης και μηχανικής λογισμικού προσεγγίζουν τη μοντελοποίηση με διαφορετικούς στόχους και, κατά συνέπεια, με διαφορετικές ερμηνευτικές παραδοχές. Στην UML, η ερμηνεία μιας συγκεκριμένης γλωσσικής κατασκευής μπορεί να εξαρτάται από το πλαίσιο μοντελοποίησης και την οπτική που υιοθετείται. Για τις ανάγκες της παρούσας σύγκρισης εξετάζεται κυρίως η αναπαραστασιοκεντρική προσέγγιση (representation-oriented interpretation), στην οποία η UML χρησιμοποιείται για τη μοντελοποίηση εννοιών του πεδίου προβλήματος και

συνεπώς παρουσιάζει τη μεγαλύτερη εννοιολογική επικάλυψη με τη χρήση της OWL για την αναπαράσταση γνώσης (Kiko & Atkinson, 2008).

Η βασικότερη διαφορά είναι ότι η OWL διαθέτει τυπικά καθορισμένη λογική σημασιολογία. Η σημασία των αξιωμάτων της δεν εξαρτάται από την πρόθεση του σχεδιαστή ή από τον τρόπο με τον οποίο ένα διάγραμμα θα ερμηνευθεί από έναν αναγνώστη. Οι κανόνες ερμηνείας της γλώσσας είναι καθορισμένοι και επιτρέπουν σε έναν reasoner να εξετάζει τις λογικές συνέπειες των δηλώσεων μιας οντολογίας.

Αντίθετα, η UML έχει σχεδιαστεί κυρίως ως γλώσσα μοντελοποίησης και επικοινωνίας. Μπορεί να εκφράσει με μεγάλη σαφήνεια τη δομή ενός συστήματος και να ενισχυθεί με περιορισμούς OCL, χωρίς όμως να παρέχει από μόνη της το ίδιο μοντέλο αυτοματοποιημένης λογικής συλλογιστικής που χαρακτηρίζει την OWL.

Μια ακόμη σημαντική διαφορά αφορά την παραδοχή ανοικτού κόσμου (Open World Assumption – OWA), η οποία χαρακτηρίζει την OWL. Η απουσία μιας πληροφορίας από μια οντολογία δεν συνεπάγεται ότι η πληροφορία είναι ψευδής. Αν, για παράδειγμα, δεν έχει δηλωθεί ότι ένα security pattern προστατεύει ένα συγκεκριμένο αγαθό, δεν μπορεί να συναχθεί αυτομάτως ότι δεν το προστατεύει. Ενδέχεται απλώς η συγκεκριμένη πληροφορία να μην έχει καταγραφεί. Η προσέγγιση αυτή είναι κατάλληλη για κατανεμημένα και εξελισσόμενα σύνολα γνώσης, στα οποία δεν θεωρείται ότι όλες οι πληροφορίες είναι εξαρχής γνωστές.

2.1.5.4 Ταυτότητα, Ισοδυναμία και Ονόματα

Μία ακόμη διαφορά μεταξύ OWL και UML αφορά τον τρόπο με τον οποίο αντιμετωπίζεται η ταυτότητα των οντοτήτων. Η OWL δεν υιοθετεί την υπόθεση μοναδικού ονόματος (Unique Name Assumption – UNA). Κατά συνέπεια, δύο διαφορετικά αναγνωριστικά δεν θεωρείται αυτομάτως ότι αναφέρονται σε δύο διαφορετικές πραγματικές οντότητες. Η γλώσσα παρέχει ρητούς μηχανισμούς για τη δήλωση ταυτότητας ή διαφοράς μεταξύ individuals, καθώς και ισοδυναμίας μεταξύ κλάσεων και ιδιοτήτων.

Η δυνατότητα αυτή είναι σημαντική σε περιβάλλοντα στα οποία συνδυάζεται γνώση από διαφορετικές πηγές. Δύο πηγές μπορεί, για παράδειγμα, να χρησιμοποιούν διαφορετικά αναγνωριστικά ή ονομασίες για την ίδια έννοια. Μέσω των σημασιολογικών μηχανισμών της OWL μπορεί να δηλωθεί ρητά ότι οι αντίστοιχες οντότητες είναι ισοδύναμες και η πληροφορία αυτή να αξιοποιηθεί από έναν reasoner.

Στην UML, αντίθετα, η ταυτότητα των στοιχείων αντιμετωπίζεται κυρίως στο πλαίσιο του ίδιου του μοντέλου και των namespaces του. Η γλώσσα δεν παρέχει τον ίδιο λογικό μηχανισμό σημασιολογικής ταύτισης διαφορετικών αναγνωριστικών και εξαγωγής συμπερασμάτων πάνω σε αυτά.

2.1.5.5 Πλεονεκτήματα, Περιορισμοί και Καταλληλότητα

Από τη σύγκριση προκύπτει ότι η UML και η OWL δεν είναι ισοδύναμες γλώσσες ούτε μπορεί να θεωρηθεί ότι η μία υπερέχει καθολικά της άλλης. Η καταλληλότητά τους εξαρτάται από τον στόχο για τον οποίο χρησιμοποιούνται.

Η UML διαθέτει σημαντικά πλεονεκτήματα όταν ζητούμενο είναι η οπτική μοντελοποίηση ενός συστήματος. Η γραφική της σημειογραφία διευκολύνει την επικοινωνία μεταξύ των συμμετεχόντων σε ένα έργο λογισμικού και επιτρέπει την αναπαράσταση τόσο της στατικής δομής όσο και διαφορετικών πλευρών της συμπεριφοράς ενός συστήματος. Επιπλέον, αποτελεί ευρέως χρησιμοποιούμενη γλώσσα στη μηχανική λογισμικού και υποστηρίζεται από μεγάλο αριθμό εργαλείων.

Η OWL είναι λιγότερο προσανατολισμένη στην οπτική παρουσίαση, αλλά παρέχει ισχυρότερη τυπική σημασιολογία για την αναπαράσταση γνώσης. Επιτρέπει τη δήλωση ιεραρχιών, ισοδυναμιών, περιορισμών και σημασιολογικών σχέσεων και παρέχει τη βάση για αυτοματοποιημένη συλλογιστική. Για τον λόγο αυτό είναι ιδιαίτερα κατάλληλη όταν απαιτείται η οργάνωση ενός πεδίου ως συνδεδεμένου δικτύου γνώσης και όχι απλώς ως μοντέλου της δομής ενός συστήματος.

Η επιλογή μεταξύ των δύο γλωσσών εξαρτάται επομένως από το πρόβλημα που επιχειρείται να επιλυθεί. Για τον σκοπό της παρούσας εργασίας, τα security patterns πρέπει να αναπαρίστανται όχι μόνο ως περιγραφές ή κατηγορίες, αλλά ως σημασιολογικά συσχετισμένες οντότητες. Ένα pattern μπορεί να ανήκει σε περισσότερες κατηγορίες, να μετριάζει συγκεκριμένες απειλές, να προστατεύει αγαθά, να υποστηρίζει ιδιότητες ασφάλειας, να εφαρμόζεται σε διαφορετικές φάσεις ανάπτυξης και να σχετίζεται με άλλα patterns και βιβλιογραφικές πηγές.

Για τις απαιτήσεις αυτές η OWL αποτελεί καταλληλότερη βασική γλώσσα αναπαράστασης, καθώς επιτρέπει οι παραπάνω συσχετίσεις να αποτελούν ρητές σημασιολογικές σχέσεις του μοντέλου. Η UML παραμένει χρήσιμη ως εργαλείο οπτικής και εννοιολογικής μοντελοποίησης, αλλά δεν χρησιμοποιείται ως ο σημασιολογικός πυρήνας της εφαρμογής. Η επιλογή αυτή συνδέεται άμεσα με την τελική υλοποίηση, όπου η γνώση αποθηκεύεται σε OWL, αναπαρίσταται λειτουργικά ως RDF γράφος και ανακτάται μέσω SPARQL επερωτήσεων.

2.2 Μοτίβα Ασφάλειας

2.2.1 Ορισμός

Ένα μοτίβο ασφάλειας αποτελεί επαναχρησιμοποιήσιμη λύση σε ένα επαναλαμβανόμενο πρόβλημα ασφάλειας που εμφανίζεται σε συγκεκριμένο πλαίσιο. Παρότι αρκετά μοτίβα ασφάλειας έχουν τη μορφή σχεδιαστικών μοτίβων (design patterns), η έννοια είναι ευρύτερη. Τα μοτίβα ασφάλειας μπορεί να αφορούν διαφορετικά επίπεδα και στάδια ανάπτυξης ενός συστήματος, όπως την ανάλυση απαιτήσεων, την αρχιτεκτονική, τον σχεδιασμό, την υλοποίηση ή ακόμη και τη

διαδικασία ανάπτυξης και διαχείρισης της ασφάλειας. Για τον λόγο αυτό στη βιβλιογραφία συναντώνται, μεταξύ άλλων, αρχιτεκτονικά, σχεδιαστικά, υλοποιητικά και διαδικαστικά μοτίβα ασφάλειας.

Λόγω της δημοτικότητας των μοτίβων σχεδίασης στην κοινότητα των μηχανικών λογισμικού, η φυσική τάση είναι να θεωρηθεί ότι οτιδήποτε φέρει το όνομα "μοτίβο ασφαλείας" πρέπει να περιγράφεται χρησιμοποιώντας ένα διάγραμμα UML και να περιλαμβάνει δείγμα πηγαίου κώδικα. Ενώ είναι αλήθεια ότι πολλά ενδιαφέροντα μοτίβα ασφαλείας μπορούν να παρουσιαστούν με αυτόν τον τρόπο, υπάρχουν πολλά άλλα σημαντικά μοτίβα ασφάλειας που δεν ταιριάζουν σε αυτούς τους περιορισμούς.

Αν και δεν είναι σκόπιμο να γίνει κάποια προσπάθεια επίσημης κατηγοριοποίησης διαφορετικών κατηγοριών μοτίβων, ωστόσο μπορούν να προσδιοριστούν δύο μεγάλες κατηγορίες:

1. ***Δομικά μοτίβα***: Πρόκειται για μοτίβα που εφαρμόζονται σε ένα προϊόν λογισμικού. Περιλαμβάνουν σχεδιαστικά μοτίβα και συχνά συνοδεύονται από διαγράμματα δομής (structural diagrams) και περιγραφές αλληλεπίδρασης (interaction descriptions/specifications).
2. ***Διαδικαστικά μοτίβα***: Πρόκειται για μοτίβα που μπορούν να χρησιμοποιηθούν για τη βελτίωση της διαδικασίας ανάπτυξης λογισμικού κρίσιμου ως προς την ασφάλεια (security-critical software). Συχνά επηρεάζουν την οργάνωση, τις διαδικασίες ή τη διαχείριση ενός έργου ανάπτυξης λογισμικού.

Τα μοτίβα ασφαλείας ενσωματώνουν αποδεδειγμένες λύσεις τεχνογνωσίας σε επαναλαμβανόμενα προβλήματα ασφάλειας. Συνήθως, οι απαιτήσεις ασφάλειας αντιμετωπίζονται με την αφαίρεση των προβλημάτων ασφάλειας που προκύπτουν σε ένα συγκεκριμένο πλαίσιο και την παροχή μιας καλά αποδεδειγμένης λύσης σε αυτά. Η ικανότητα των προτύπων ασφαλείας να μετριάζουν και να σταματούν τις απειλές ασφαλείας μπορεί να βρεθεί στην εργασία των Steel & Nagappan (2006) όπου τα μοτίβα ασφαλείας που ενσωματώνονται σε ένα σύστημα θα μπορούσαν να συμβάλλουν στη βελτίωση του επιπέδου ασφάλειας του συστήματος αυτού.

Θα πρέπει να σημειωθεί ότι τα μοτίβα ασφαλείας μπορούν να σχεδιαστούν και να αναπτυχθούν από ειδικούς σε θέματα ασφάλειας για διαφορετικά είδη επίλυσης προβλημάτων και να εφαρμοστούν σε διαφορετικά περιβάλλοντα. Για παράδειγμα, μπορεί να είναι αφηρημένα αρχιτεκτονικά μοτίβα υψηλότερου επιπέδου που καθορίζουν τον τρόπο επίλυσης ενός προβλήματος ασφάλειας αρχιτεκτονικά ή μπορεί να είναι ακόμη πιο αφηρημένα μοτίβα που απεικονίζουν τη διαδικασία για την ασφαλή ανάπτυξη λογισμικού.

2.2.1.1 Ταξινόμηση και Οργάνωση Μοτίβων Ασφαλείας

Η τεκμηρίωση των προτύπων ασφαλείας δημιουργήθηκε αρχικά από τους Yoder & Barcalow (1997). Η δομή ενός μοτίβου αποτελείται από πολλά στοιχεία, τα οποία είναι τα ακόλουθα:

- ***Πρόθεση***: περιγραφή του στόχου και ζητήματα που αντιμετωπίζει το μοτίβο

- ***Πλαίσιο***: περιγραφή καταστάσεων ή περιβάλλοντος στο οποίο χρησιμοποιείται το μοτίβο.
- ***Πρόβλημα***: περιγραφή του προβλήματος (ασφάλειας) που λύνει αυτό το μοτίβο
- ***Περιγραφή***: περιγραφή των σεναρίων που απεικονίζουν το πρόβλημα σχεδιασμού.
- ***Λύση***: περιγραφή της λύσης του προβλήματος.
- ***Συνέπειες***: περιγραφή των αντισταθμίσεων και των αποτελεσμάτων όταν χρησιμοποιείται αυτό το πρότυπο.
- ***Δυνάμεις***: περιγραφή των περιορισμών που πρέπει να λαμβάνονται υπόψη όταν εφαρμόζεται το πρότυπο.
- ***Γνωστές χρήσεις***: περιγραφή της χρήσης του μοτίβου σε πραγματικά συστήματα.
- ***Σχετικά μοτίβα***: περιγραφή των σχετικών μοτίβων που χρησιμοποιούν αυτό το μοτίβο ως αναφορά.

Κατά την σύνταξη ενός μοτίβου, οι συντάκτες μοτίβων μπορούν να περιγράψουν όλες τις ενότητες που θεωρούν σημαντικές. Η δομή των προτύπων ασφαλείας που υιοθετήθηκαν στην εργασία των Guan et al. (2016) βασίστηκε στα παραδοσιακά μοτίβα σχεδίασης, τα οποία χαρακτηρίζονται από ένα εκφραστικό όνομα, ένα πλαίσιο εφαρμογής, ένα πρόβλημα προς επίλυση και μια λύση για αυτό το πρόβλημα. Επομένως, το μοτίβο ασφαλείας αναπαρίσταται ως μια τριπλέτα <Context, Problem, Solution> όπου:

- ***Το πλαίσιο (context)*** καθορίζει τις συνθήκες και την κατάσταση στην οποία εφαρμόζεται το μοτίβο: Ο χρόνος και η τοποθεσία συνήθως θεωρούνται ως σημαντικά χαρακτηριστικά του πλαισίου στον τομέα της ασφάλειας. Ο χρόνος σχετίζεται με το πότε παρουσιάζεται ένα πρόβλημα ασφαλείας και η τοποθεσία καθορίζει το σε ποιο επίπεδο της υποδομής του συστήματος παρουσιάζεται το πρόβλημα αυτό. Όσον αφορά τον τομέα λογισμικού, τυπικό παράδειγμα του χρόνου μέσα σε ένα πλαίσιο είναι οι φάσεις του κύκλου ζωής του λογισμικού, δηλαδή η ανάλυση, ο σχεδιασμός, η υλοποίηση και η εν γένει λειτουργία του λογισμικού.
- ***Το πρόβλημα (problem)*** καθορίζει την ευάλωτη πτυχή ενός αγαθού: Το πεδίο προβλημάτων ενός προτύπου ασφαλείας είναι σημαντικό για τους προγραμματιστές λογισμικού προκειμένου να προσδιορίσουν εάν ένα μοτίβο ασφαλείας είναι κατάλληλο για την κατάστασή τους. Αυτό το πεδίο ορίζει τα προβλήματα ασφαλείας που παρουσιάζονται σε συγκεκριμένα περιβάλλοντα και τα οποία μπορούν να επιλυθούν από το μοτίβο ασφαλείας. Πρόβλημα ασφαλείας παρουσιάζεται κάθε φορά που ένα σύστημα είναι απροστάτευτο ή δεν προστατεύεται επαρκώς από κατάχρηση ή κακή χρήση. Σε γενικές γραμμές, το πρόβλημα ασφάλειας μπορεί να είναι κάποια είδη απειλών που προκαλούν πιθανό κίνδυνο ή ζημιά όταν κάποιος ή κάτι παραβιάζει τις πολιτικές ασφαλείας.
- ***Η λύση (solution)*** ορίζει το σχήμα που επιλύει το πρόβλημα ασφάλειας, το οποίο παρουσιάστηκε στο πλαίσιο ασφάλειας: Η λύση ασφαλείας αποτελεί μια ομάδα αντιμέτρων

που πρέπει να εφαρμοστούν στο σύστημα προκειμένου να μετριαστεί ο κίνδυνος ασφάλειας. Είναι σημαντικό να εφαρμόζεται τουλάχιστον ένα αντίμετρο ασφαλείας για να διατηρείται το σύστημα άτρωτο για κάθε απειλή.

Ένας σημαντικός αριθμός μοτίβων ασφάλειας έχει προταθεί από το 1997, όταν οι Yoder & Barcalow (1997) παρουσίασαν μία από τις πρώτες σχετικές προσεγγίσεις. Ένα μοτίβο ασφάλειας μπορεί να αφορά περισσότερες από μία ιδιότητες ασφάλειας. Για παράδειγμα, ένα μοτίβο ελέγχου ταυτότητας μπορεί να συμβάλλει στην προστασία τόσο της εμπιστευτικότητας όσο και της ακεραιότητας. Αντίστοιχα, για την υποστήριξη μιας συγκεκριμένης ιδιότητας ασφάλειας μπορεί να υπάρχουν περισσότερα από ένα κατάλληλα μοτίβα. Επομένως, μεταξύ μοτίβων και ιδιοτήτων ασφάλειας μπορεί να αναπτύσσεται σχέση πολλά-προς-πολλά. Επιπλέον, τα μοτίβα ασφάλειας μπορούν να οργανωθούν με βάση διαφορετικά κριτήρια και σε διαφορετικά επίπεδα αφαίρεσης. Ως εκ τούτου, η επιλογή των κατάλληλων μοτίβων για την αντιμετώπιση ενός συγκεκριμένου προβλήματος ασφάλειας καθίσταται δυσκολότερη χωρίς ένα κατάλληλο σχήμα ταξινόμησης. Ένα τέτοιο σύστημα ταξινόμησης διευκολύνει τόσο την οργανωμένη αποθήκευση και ανάκτηση των σχετικών πληροφοριών όσο και την αναζήτηση και επιλογή των κατάλληλων μοτίβων.

Ακολουθεί η περιγραφή ενός αποτελεσματικού πλαισίου ταξινόμησης για μοτίβα ασφαλείας ώστε να διευκολυνθεί η εύρεση των κατάλληλων εκείνων μοτίβων ασφαλείας που είναι σύμφωνα με τις παρεχόμενες απαιτήσεις ασφαλείας. Καθώς η απαίτηση ασφαλείας βασίζεται σε μοντελοποίηση απειλών και ανάλυση αγαθών, οι ιδιότητες της απειλής και του αγαθού θα θεωρηθούν ως παράγοντες για την επιλογή μοτίβων ασφαλείας.

Οι Guan et al. (2016) προτείνουν ένα πολυδιάστατο πλαίσιο ταξινόμησης των μοτίβων ασφάλειας, με στόχο να διευκολύνεται η επιλογή κατάλληλων μοτίβων με βάση τις εκάστοτε απαιτήσεις ασφάλειας. Η προσέγγιση λαμβάνει υπόψη διαφορετικά χαρακτηριστικά του προβλήματος ασφάλειας και του περιβάλλοντος εφαρμογής, όπως οι απειλές, τα προστατευόμενα αγαθά και άλλες σχετικές παράμετροι. Το σύστημα ταξινόμησης που προτείνουν βασίζεται σε πολλαπλές διαστάσεις, οι οποίες παρουσιάζονται στη συνέχεια.

- ***Στάδιο κύκλου ζωής***. Παρότι πολλά μοτίβα ασφάλειας προέρχονται από την παράδοση των σχεδιαστικών μοτίβων, δεν εφαρμόζονται αποκλειστικά κατά τη φάση της σχεδίασης. Για τον λόγο αυτό, οι Guan et al. (2016) χρησιμοποιούν ως διάσταση ταξινόμησης το στάδιο του κύκλου ζωής στο οποίο εφαρμόζεται ένα μοτίβο. Στο συγκεκριμένο σχήμα η Αρχιτεκτονική αντιμετωπίζεται ως διακριτή κατηγορία, παρότι σε άλλες προσεγγίσεις μπορεί να θεωρείται μέρος της ευρύτερης φάσης σχεδίασης. Οι κατηγορίες που χρησιμοποιούνται είναι: Ανάλυση (Analysis), Αρχιτεκτονική (Architecture), Σχεδιασμός (Design), Υλοποίηση (Implementation) και Διάταξη (Deployment).
- ***Αρχιτεκτονικό Επίπεδο***. Το επίπεδο αυτό παρέχει μια άλλη χρήσιμη διάσταση, καθώς τα προβλήματα και οι λύσεις τους σε διαφορετικά επίπεδα της αρχιτεκτονικής διαφέρουν, αλλά όλα είναι σημαντικά. Χονδρικά, η αρχιτεκτονική έχει χωριστεί με μια ακολουθία από το

χαμηλό στο υψηλό επίπεδο αφαίρεσης. Ως επίπεδα αρχιτεκτονικής χρησιμοποιούνται οι ακόλουθες διακρίσεις: Δεδομένα, Εφαρμογή, Σύστημα και Δίκτυο.

- ***Πλαίσιο εφαρμογής***. Η δομή του συστήματος συνήθως λαμβάνεται υπόψη ως ένας άλλος παράγοντας ταξινόμησης για την κατανομή των μοτίβων ασφαλείας ανάλογα με το τμήμα του συστήματος που προσπαθούν να προστατεύσουν. Η ασφάλεια ενός συστήματος αναλύεται σε τρία επίπεδα: ασφάλεια πυρήνα, ασφάλεια περιμέτρου και εξωτερική ασφάλεια. Η ασφάλεια πυρήνα ασχολείται με την εφαρμογή της ασφάλειας εντός του συστήματος, ενώ η περιμετρική ασφάλεια εστιάζει σε ζητήματα που σχετίζονται με την ασφάλεια στα σημεία εισόδου του συστήματος, όπως η εξουσιοδότηση, ο έλεγχος ταυτότητας και η ασφάλεια. Η εξωτερική ασφάλεια εξετάζει την προστασία των δεδομένων κατά τη μετάδοση τους και την ασφάλεια των πρωτοκόλλων επικοινωνίας.
- ***Τομέας Εφαρμογής***. Ο τομέας εφαρμογής μπορεί να παρέχει έναν σημαντικό παράγοντα διαφοροποίησης ή ένα φίλτρο για να περιορίσει το πεδίο εφαρμογής της γνώσης. Ορισμένες λύσεις μοτίβων ασφαλείας είναι συγκεκριμένες για έναν συγκεκριμένο τομέα ή τύπο εφαρμογής. Αυτή η διάσταση αποτελεί εξαίρεση όσον αφορά το γεγονός ότι δεν απαιτεί διχοτόμηση ή διάταξη — ο χώρος ορίζεται ελεύθερα. Οι σχεδιαστές μοτίβων μπορούν να δημιουργήσουν μοτίβα για τον δικό τους τομέα με τη μορφή σύλληψης γνώσης. Μάλιστα, υπάρχουν διάφορα παραδείγματα τομέων όπου κάτι τέτοιο έχει συμβεί όπως: το Ubiquitous computing, η κατανεμημένη υπολογιστική (distributed computing), το Web και το J2EE, ενσωματωμένα συστήματα (embedded systems), λειτουργικά συστήματα, η αρχιτεκτονική προσανατολισμένη στην υπηρεσία (service-oriented architecture–SOA), η SCADA (εποπτικός έλεγχος και ανάκτηση δεδομένων σε βιομηχανικά συστήματα) κ.λ.π..
- ***Τύπος απειλής***. Το σχήμα ταξινόμησης που βασίζεται στη μοντελοποίηση απειλών είναι πιο διαισθητικό επειδή προδιαγράφουν τα προβλήματα ασφάλειας που επιλύουν τα μοτίβα. Οι αρχιτέκτονες ασφαλείας χρησιμοποιούν μοντελοποίηση απειλών για να προσδιορίσουν και να ιεραρχήσουν την απειλή ασφαλείας ενός συστήματος, το οποίο καθιστά δυνατή την προτεραιοποίηση της προσπάθειας μετριασμού. Το STRIDE είναι ένα από τα ευρέως χρησιμοποιούμενα μοντέλα για την ταξινόμηση των απειλών σύμφωνα με διαφορετικές πηγές. Είναι το αγγλικό ακρωνύμιο των παρακάτω έξι τύπων απειλών:
    - ***Spoofing (πλαστογράφηση):*** είναι κάποιος ή κάτι που εμφανίζεται υπό τη μάσκα του νόμιμου ή έγκυρου.
    - ***Tampering (Παραβίαση):*** Πρόκειται για δεδομένα που υποκλέπτονται, παρεμβάλλονται ή τροποποιούνται κατά την δικτυακή επικοινωνία.
    - ***Repudiation (Απάρνηση):*** είναι η κατάσταση κατά την οποία ο χρήστης αρνείται να επιβεβαιώσει μια συγκεκριμένη ενέργεια, συνήθως επειδή είναι παράνομη και επιβλαβής.

- ***Information disclosure (αποκάλυψη πληροφοριών):*** Όταν ένας μη εξουσιοδοτημένος χρήστης αποκτά πρόσβαση σε εμπιστευτικές πληροφορίες, στις οποίες υποτίθεται ότι δεν έχει πρόσβαση.
- ***Denial of service (Άρνηση υπηρεσίας):*** Όταν μια υπηρεσία καταρρίπτεται ή παρεμποδίζεται ηθελημένα ή ακούσια, με αποτέλεσμα τη διακοπή της παράδοσής της προς τους νόμιμους χρήστες.
- ***Elevation of privilege (ανύψωση του προνομίου):*** Όταν ένας χρήστης αποκτά πρόσβαση υψηλότερων προνομίων από αυτήν που υποτίθεται ότι είχε, κάτι που μπορεί να οδηγήσει σε μη εξουσιοδοτημένη πρόσβαση σε εμπιστευτικές πληροφορίες ή εφαρμογή επικίνδυνων εργασιών.

- ***Στόχοι και Ιδιότητες Ασφάλειας***. Μία ακόμη διάσταση του συστήματος ταξινόμησης των Guan et al. (2016) αφορά τους στόχους ή τις ιδιότητες ασφάλειας που επιδιώκει να προστατεύσει ένα μοτίβο. Στις βασικές κατηγορίες περιλαμβάνονται ο Έλεγχος Πρόσβασης, ο Έλεγχος Ταυτότητας, η Εμπιστευτικότητα, η Ακεραιότητα, η Διαθεσιμότητα, η Λογοδοσία και η Μη Απάρνηση (Non-repudiation). Η διάσταση αυτή παρουσιάζει αναμενόμενη επικάλυψη με την προηγούμενη ταξινόμηση βάσει τύπου απειλής. Για παράδειγμα, η απειλή Denial of Service σχετίζεται άμεσα με τη Διαθεσιμότητα, ενώ η Repudiation σχετίζεται με τη Μη Απάρνηση. Ωστόσο, οι δύο διαστάσεις εξετάζουν το πρόβλημα από διαφορετική οπτική: ο τύπος απειλής περιγράφει το ανεπιθύμητο γεγονός ή την ενέργεια που πρέπει να αντιμετωπιστεί, ενώ ο στόχος ασφάλειας εκφράζει την ιδιότητα ή απαίτηση που πρέπει να προστατευθεί. Για τον λόγο αυτό οι δύο κατηγοριοποιήσεις είναι συσχετισμένες αλλά όχι ταυτόσημες.

2.2.1.2 Οντολογική Μοντελοποίηση Μοτίβων Ασφάλειας

Η χρήση οντολογιών για τη μοντελοποίηση μοτίβων ασφάλειας παρέχει σημαντικά πλεονεκτήματα σε σχέση με την απλή αποθήκευσή τους ως ανεξάρτητες περιγραφές κειμένου. Οι έννοιες που χαρακτηρίζουν ένα μοτίβο, όπως οι απειλές που αντιμετωπίζει, τα αγαθά που προστατεύει, οι ιδιότητες ασφάλειας που υποστηρίζει και το στάδιο ανάπτυξης στο οποίο εφαρμόζεται, μπορούν να αναπαρασταθούν ως διακριτές αλλά σημασιολογικά συσχετισμένες οντότητες. Με τον τρόπο αυτό διευκολύνονται η συνεπής ταξινόμηση και επαναχρησιμοποίηση της γνώσης, η σημασιολογική αναζήτηση και η συσχέτιση μοτίβων με συγκεκριμένες απαιτήσεις ή προβλήματα ασφάλειας. Επιπλέον, όταν το χρησιμοποιούμενο οντολογικό μοντέλο το επιτρέπει, η τυπική σημασιολογία μπορεί να αξιοποιηθεί από κατάλληλους μηχανισμούς συλλογιστικής για την εξαγωγή πρόσθετων συμπερασμάτων από τις ήδη καταγεγραμμένες σχέσεις.

Για παράδειγμα, σε μια οντολογία ασφάλειας μπορεί να δηλωθεί ότι μια επίθεση άρνησης υπηρεσίας (Denial of Service) αποτελεί τύπο απειλής που επηρεάζει τη Διαθεσιμότητα (Availability), ενώ ένα συγκεκριμένο μοτίβο ασφάλειας μπορεί να συνδεθεί με την ίδια απειλή

μέσω μιας σχέσης αντιμετώπισης ή μετριασμού. Μέσω των ρητών αυτών σχέσεων είναι δυνατό να συσχετιστεί το μοτίβο όχι μόνο με την απειλή που αντιμετωπίζει, αλλά και με την αντίστοιχη ιδιότητα ασφάλειας. Με τον τρόπο αυτό, η πληροφορία οργανώνεται ως διασυνδεδεμένη γνώση και όχι ως σύνολο ανεξάρτητων όρων ή καταχωρίσεων.

2.2.2 Συσχέτιση Μοτίβων Ασφάλειας με τη Διαδικασία Ανάπτυξης Λογισμικού

Παρά την ανάπτυξη βέλτιστων πρακτικών για τη διαχείριση των απαιτήσεων ασφάλειας, τη χρήση μοτίβων απαιτήσεων ασφάλειας (security requirements patterns) και την εξέλιξη προσεγγίσεων μοντελοποίησης και διαχείρισης απαιτήσεων ασφάλειας, εξακολουθούν να εμφανίζονται παραβιάσεις και αστοχίες ασφάλειας σε συστήματα λογισμικού. Τα μοτίβα απαιτήσεων ασφάλειας αποτελούν ειδικότερη κατηγορία μοτίβων που χρησιμοποιείται κυρίως κατά τη μηχανική απαιτήσεων, με στόχο την επαναχρησιμοποίηση και συστηματική διατύπωση επαναλαμβανόμενων απαιτήσεων ασφάλειας, σε αντίθεση με τα γενικότερα μοτίβα ασφάλειας, τα οποία μπορούν να εφαρμοστούν και σε άλλα στάδια του κύκλου ζωής ανάπτυξης.

Ενδεικτικά, το Information Security Breaches Survey 2015 κατέγραψε ότι το 90% των μεγάλων οργανισμών που συμμετείχαν στην έρευνα είχαν υποστεί κάποια μορφή παραβίασης ασφάλειας, έναντι 81% το προηγούμενο έτος PwC (2015). Τα στοιχεία αυτά υπογραμμίζουν ότι η ασφάλεια δεν πρέπει να αντιμετωπίζεται ως μεμονωμένη δραστηριότητα σε ένα μόνο στάδιο της ανάπτυξης, αλλά να λαμβάνεται υπόψη σε ολόκληρο τον κύκλο ζωής του λογισμικού. Η παράλειψη ζητημάτων ασφάλειας σε πρώιμες ή μεταγενέστερες φάσεις μπορεί να οδηγήσει σε ευπάθειες που είναι δυσκολότερο και δαπανηρότερο να αντιμετωπιστούν στη συνέχεια.

Η ασφαλής ανάπτυξη λογισμικού απαιτεί την ενσωμάτωση ζητημάτων ασφάλειας σε κάθε διαφορετική φάση του κύκλου ζωής ανάπτυξης του λογισμικού (π.χ. απαιτήσεις, σχεδιασμός, υλοποίηση και δοκιμή). Είναι ζωτικής σημασίας να ενσωματωθεί από την αρχή η ασφάλεια, ειδικά στη φάση της ανάλυσης απαιτήσεων, ώστε να αποφευχθεί η επανεπεξεργασία και οι αστοχίες λογισμικού ως προς την ασφάλεια. Μέσω των πρώιμων σταδίων ανάπτυξης ασφαλούς λογισμικού, η συμμετοχή των Μηχανικών Απαιτήσεων (Requirements Engineers) διαπιστώθηκε ότι είναι κρίσιμη, καθώς αυτοί διαχειρίζονται όλα τα προαπαιτούμενα, συμπεριλαμβανομένων των ίδιων των απαιτήσεων ασφαλείας. Ωστόσο, οι ερευνητές ανακάλυψαν ότι τα κύρια εμπόδια στην ανάπτυξη ασφαλούς λογισμικού είναι η έλλειψη γνώσεων και τεχνογνωσίας για την ασφάλεια μεταξύ των απλών προγραμματιστών λογισμικού, συμπεριλαμβανομένων των Μηχανικών Απαιτήσεων Elahi (2009).

Μεταξύ των προσπαθειών που έχουν πραγματοποιηθεί από ερευνητές για τη διαχείριση των απαιτήσεων ασφαλείας είναι ο σχεδιασμός μοτίβων ασφαλείας. Διαπιστώθηκε ότι είναι πολλά τα μοτίβα ασφαλείας που έχουν τεκμηριωθεί από διάφορους συγγραφείς. Αυτά τα μοτίβα έχουν τους δικούς τους σκοπούς και στόχους καθώς και επιλύουν διάφορα προβλήματα ασφάλειας. Πολλοί ερευνητές ενσωματώνουν διάφορα υπάρχοντα μοτίβα ασφαλείας με ίδιους στόχους, που

επικεντρώνονται στην ανάπτυξη ασφαλούς λογισμικού. Ο στόχος της μελέτης των Guan et al. (2016) είναι να αποσαφηνιστεί ο ορισμός των μοτίβων απαιτήσεων ασφαλείας και επίσης να βρεθεί η γνώση των στόχων ασφαλείας που καλύπτονται μεταξύ των υφιστάμενων προτύπων απαιτήσεων ασφαλείας.

Η διατήρηση της ασφάλειας σε ένα σύστημα λογισμικού αποτελεί ένα δυναμικό αγώνα. Πρόκειται για μια μάχη για την εξασφάλιση νόμιμων δικαιωμάτων πρόσβασης έναντι παράνομης χρήσης σε ένα σύστημα λογισμικού. Τα μοτίβα ασφαλείας είναι πάντα παρόντα σε οποιαδήποτε εφαρμογή ασφαλείας για πολλές δεκαετίες. Αυτά τα μοτίβα είναι επαναλαμβανόμενα για την παροχή συγκεκριμένου επιπέδου ασφάλειας, όπως τα χαρακτηριστικά ασφαλείας του «Παραθύρου σύνδεσης»/ «Login Windows», το οποίο είναι ένα παράδειγμα μοτίβου ασφαλείας ενός σημείου πρόσβασης (SAP). Οι ερευνητές ταξινόμησαν τα μοτίβα ασφαλείας στους τυπικούς καταλόγους προτύπων ασφαλείας. Αυτοί οι κατάλογοι παρέχουν λεπτομέρειες εφαρμογής μοτίβων ασφαλείας χρησιμοποιώντας τυπικά μοτίβα προτύπων ασφαλείας (Fernandez, 2013). Τα μοτίβα ασφάλειας λογισμικού αποτελούν ένα από τα δομικά στοιχεία της ασφάλειας λογισμικού αφού μπορούν να βοηθήσουν στην παρεμπόδιση των εισβολέων από το να χρησιμοποιούν παράνομα ένα σύστημα λογισμικού. Άρα, μπορούν να αποτελούν και είναι τη λύση σε προβλήματα σχεδιασμού ασφάλειας.

Στην ανάπτυξη ασφαλούς λογισμικού, τα μοτίβα ασφάλειας μπορούν να χρησιμοποιηθούν ως επαναχρησιμοποιήσιμα αρχιτεκτονικά και σχεδιαστικά στοιχεία για την αντιμετώπιση επαναλαμβανόμενων προβλημάτων ασφάλειας. Η διασφάλιση της ορθής εφαρμογής των μοτίβων ασφάλειας σε ένα σύστημα λογισμικού αποτελεί σημαντικό μέρος των διαδικασιών επαλήθευσης και διασφάλισης ποιότητας. Ως ορθή εφαρμογή νοείται η υλοποίηση ενός μοτίβου σύμφωνα με τη δομή, τις σχέσεις και τους περιορισμούς που περιγράφονται στην τεκμηρίωσή του. Οι κατάλογοι μοτίβων επιτρέπουν την εξέταση και αξιολόγηση των ίδιων των περιγραφών των μοτίβων, όμως για να αξιολογηθεί κατά πόσο ένα συγκεκριμένο μοτίβο έχει πράγματι εφαρμοστεί ορθά σε ένα υπάρχον σύστημα απαιτείται πρώτα ο εντοπισμός της υλοποίησής του μέσα σε αυτό.

Η ασφάλεια αποτελεί ένα από τα χαρακτηριστικά ποιότητας ενός συστήματος λογισμικού και περιλαμβάνει επιμέρους ιδιότητες, όπως η εμπιστευτικότητα (confidentiality), η ακεραιότητα (integrity), η διαθεσιμότητα (availability), η αυθεντικότητα (authenticity), η λογοδοσία (accountability) και η μη απάρνηση (non-repudiation). Παράλληλα, ένα σύστημα λογισμικού χαρακτηρίζεται και από άλλα χαρακτηριστικά ποιότητας, όπως η απόδοση και η δυνατότητα τροποποίησης. Τα διαφορετικά αυτά χαρακτηριστικά δεν είναι πάντοτε ανεξάρτητα μεταξύ τους, καθώς η ενίσχυση ενός από αυτά μπορεί να επηρεάσει κάποιο άλλο. Για παράδειγμα, η προσθήκη επιπλέον μηχανισμών ασφάλειας μπορεί να επιβαρύνει την απόδοση ενός συστήματος, ενώ συχνές τροποποιήσεις του λογισμικού μπορούν, εάν δεν συνοδεύονται από κατάλληλους ελέγχους, να επηρεάσουν αρνητικά την ασφάλειά του.

Κατά την ανάπτυξη ασφαλούς λογισμικού, ακόμη και μετά την υλοποίηση των απαιτήσεων ασφάλειας, ένα μοτίβο ασφάλειας μπορεί να έχει εφαρμοστεί ελλιπώς, εσφαλμένα ή με τρόπο που

αποκλίνει από την προβλεπόμενη χρήση του. Οι δοκιμές και οι διαδικασίες διασφάλισης ποιότητας μπορούν να χρησιμοποιηθούν για τον εντοπισμό αδυναμιών στην υλοποίηση και ενσωμάτωση των μοτίβων και για την επαλήθευση της ορθής εφαρμογής τους.

Η αξιολόγηση δεν θα πρέπει, ωστόσο, να περιορίζεται στην εξέταση κάθε μοτίβου μεμονωμένα. Όταν περισσότερα μοτίβα ασφάλειας χρησιμοποιούνται συνδυαστικά, είναι απαραίτητο να εξετάζεται και η μεταξύ τους αλληλεπίδραση, καθώς ένας συνδυασμός που δεν έχει σχεδιαστεί ή υλοποιηθεί σωστά μπορεί να δημιουργήσει κενά ασφάλειας, ασυμβατότητες ή ανεπιθύμητες αλληλεπιδράσεις. Επομένως, οι διαδικασίες δοκιμών και διασφάλισης ποιότητας πρέπει να εξετάζουν τόσο την ορθή εφαρμογή κάθε μοτίβου όσο και τη συνολική επίδραση του συνδυασμού τους στους στόχους ασφάλειας του συστήματος.

Για παράδειγμα, περισσότερα από ένα μοτίβα ασφάλειας μπορούν να χρησιμοποιηθούν για την ενίσχυση της αρχιτεκτονικής ελέγχου πρόσβασης ενός συστήματος λογισμικού. Τα μοτίβα Single Access Point, Check Point και Security Session μπορούν να εφαρμοστούν συνδυαστικά με στόχο τον έλεγχο της πρόσβασης των χρηστών στους προστατευόμενους πόρους του συστήματος Schumacher (2013). Στην περίπτωση αυτή, η αξιολόγηση πρέπει να εξετάζει όχι μόνο εάν κάθε μοτίβο έχει εφαρμοστεί σωστά, αλλά και εάν οι μεταξύ τους σχέσεις διατηρούν συνολικά τις επιδιωκόμενες ιδιότητες ασφάλειας.

2.3 Σχετικές Ερευνητικές Εργασίες

2.3.1 Ταξινόμηση Μοτίβων Ασφάλειας

Ο αυξανόμενος αριθμός μοτίβων ασφάλειας που εμφανίζονται στη βιβλιογραφία, καθώς και οι διαφορετικές ονομασίες και ιδιότητες με τις οποίες περιγράφονται, έχουν οδηγήσει στην ανάπτυξη διαφορετικών προσεγγίσεων ταξινόμησής τους. Μια ταξινόμηση οργανώνει μοτίβα σε ομάδες μοτίβων που μοιράζονται μία ή πολλές ιδιότητες, όπως είναι ο τομέας εφαρμογής ή ένας συγκεκριμένος σκοπός. Μία από τις θεμελιώδεις προσεγγίσεις ταξινόμησης σχεδιαστικών μοτίβων προτάθηκε από τους Gamma et al. (1995) στον κατάλογο των λεγόμενων Gang of Four (GoF) patterns. Οι συγγραφείς ταξινόμησαν γενικά αντικειμενοστραφή σχεδιαστικά μοτίβα σε δημιουργικά (Creational), δομικά (Structural) και συμπεριφορικά (Behavioral) μοτίβα και όχι ειδικά σε μοτίβα ασφάλειας. Η προσέγγιση αφορά επομένως τη φάση σχεδιασμού του λογισμικού και αποτέλεσε γενικότερη βάση για μεταγενέστερες προσεγγίσεις οργάνωσης και περιγραφής εξειδικευμένων μοτίβων.

Οι Heyman et al. (2007) ταξινόμησαν 220 μοτίβα ασφαλείας σε τρεις κατηγορίες: κατευθυντήριες γραμμές και αρχές (guidelines and principles), δραστηριότητες διαδικασίας (process activities) και βασικά μοτίβα (core patterns). Ως βασικά μοτίβα θεωρούνται εκείνα που ικανοποιούν τον ορισμό ενός μοτίβου, δηλαδή περιγράφουν ένα σαφώς οριοθετημένο πρόβλημα και μια επαναχρησιμοποιήσιμη, κατασκευάσιμη και δοκιμασμένη λύση. Αντίθετα, οι

κατευθυντήριες γραμμές και οι αρχές παρέχουν γενικές συστάσεις χωρίς συγκεκριμένη κατασκευαστική λύση, ενώ οι δραστηριότητες διαδικασίας περιγράφουν ενέργειες ή πρακτικές που πρέπει να εκτελεστούν στο πλαίσιο της διαδικασίας ανάπτυξης. Στην ανάλυση των 220 καταχωρίσεων, οι Heyman et al. (2007) ταξινόμησαν το 55% ως βασικά μοτίβα, το 35% ως κατευθυντήριες γραμμές και αρχές και το 10% ως δραστηριότητες διαδικασίας. Οι κατευθυντήριες γραμμές σχεδιασμού που περιγράφηκαν από τους Viega & McGraw (2001) χρησιμοποιήθηκαν για τη σύγκριση 8 προτύπων ασφαλείας από τους Konrad et al. (2003). Επέκτειναν την ταξινόμησή τους με βάση τους τύπους πρόσβασης των μοτίβων ασφαλείας και κατά συνέπεια ταξινόμησαν τα μοτίβα όσον αφορά τα επίπεδα εφαρμογής:σε επίπεδο δικτύου,

- σε επίπεδο κεντρικού υπολογιστή και
- σε επίπεδο εφαρμογής.

Οι Kienzle et al. (2002) ταξινόμησαν τα μοτίβα ασφάλειας σε δύο μεγάλες κατηγορίες: τα δομικά (structural patterns) και τα διαδικαστικά (procedural patterns). Τα δομικά μοτίβα αφορούν λύσεις που μπορούν να ενσωματωθούν στη δομή ενός συστήματος λογισμικού, σε αρχιτεκτονικό, σχεδιαστικό ή υλοποιητικό επίπεδο. Αντίθετα, τα διαδικαστικά μοτίβα αφορούν πρακτικές και δραστηριότητες που αποσκοπούν στη βελτίωση της διαδικασίας ανάπτυξης λογισμικού κρίσιμου ως προς την ασφάλεια.

Μια άλλη ταξινόμηση προτάθηκε από τους Blakley & Heath (2004), οι οποίοι διακρίνουν τα μοτίβα σε δύο κατηγορίες: μοτίβα διαθέσιμων συστημάτων (Available System Patterns) και μοτίβα προστατευμένων συστημάτων (Protected System Patterns). Τα Available System Patterns αποσκοπούν στην κατασκευή συστημάτων που παρέχουν προβλέψιμη και αδιάλειπτη πρόσβαση στις υπηρεσίες και στους πόρους τους, αντιμετωπίζοντας ζητήματα όπως αστοχίες και ανοχή σφαλμάτων. Αντίθετα, τα Protected System Patterns αποσκοπούν στην προστασία πολύτιμων πόρων από μη εξουσιοδοτημένη χρήση, αποκάλυψη ή τροποποίηση. Επομένως, οι δύο κατηγορίες αφορούν διαφορετικούς στόχους προστασίας ενός συστήματος και όχι τη «διαθεσιμότητα» των ίδιων των μοτίβων.

2.3.2 Περιγραφή Μοτίβων Ασφάλειας

Τα μοτίβα ασφάλειας περιγράφουν επαναλαμβανόμενα προβλήματα και αντίστοιχες λύσεις που σχετίζονται με την ανάπτυξη και λειτουργία ασφαλών συστημάτων λογισμικού. Οι Washizaki et al. (2021) αναφέρουν ότι από τα τέλη της δεκαετίας του 1990 έχουν προταθεί περίπου 500 μοτίβα ασφάλειας. Παρά τη σημαντική έρευνα γύρω από τις τεχνικές πτυχές τους, ζητήματα όπως η συνολική κατεύθυνση της έρευνας και τα εμπόδια στην πρακτική εφαρμογή τους δεν είχαν μελετηθεί στον ίδιο βαθμό. Για τον λόγο αυτό, οι Washizaki et al. (2021) πραγματοποίησαν συστηματική βιβλιογραφική ανασκόπηση (Systematic Literature Review – SLR) 240 δημοσιεύσεων και πρότειναν μια ταξινόμηση της σχετικής έρευνας. Στόχος της ταξινόμησης ήταν, μεταξύ άλλων, η διευκόλυνση της επικοινωνίας μεταξύ ερευνητών και επαγγελματιών, η

συστηματοποίηση της σχετικής ορολογίας και η αποτελεσματικότερη αξιοποίηση των μοτίβων ασφάλειας.

Είναι σαφές ότι μόνο η εξέταση της τριάδας των περιορισμένων χαρακτηριστικών ασφαλείας (CIA: Εμπιστευτικότητα (Confidentiality), Ακεραιότητα (Integrity) και Διαθεσιμότητα (Availability)) δεν αρκεί για να επιτευχθεί σωστά η πολυπλοκότητα των ασφαλών συστημάτων λογισμικού. Επομένως, είναι σημαντικό να υπάρχει ένας πόρος για τον χαρακτηρισμό της διαθέσιμης έρευνας μοτίβων ασφαλείας σχετικά με διάφορα χαρακτηριστικά ασφαλείας, όχι μόνο τα CIA αλλά και άλλα, όπως κατάλληλες μεθοδολογίες και φάσεις ανάπτυξης, για την υποστήριξη των επαγγελματιών που επιλέγουν υπάρχουσες μεθόδους και εργαλεία μοτίβων ασφαλείας και για τη βοήθεια στην ασφαλή ανάπτυξη συστημάτων και λογισμικού. Για το λόγο αυτό, η κοινότητα ασφαλείας διεξάγει περαιτέρω έρευνες σε μεθόδους και εργαλεία μοτίβων ασφαλείας.

Η ταξινόμηση των Washizaki et al. (2021) περιλαμβάνει 13 επιμέρους διαστάσεις (facets), οργανωμένες σε πέντε ευρύτερες κατηγορίες: (α) Σκοπός (Purpose), που περιλαμβάνει το ερευνητικό θέμα (Topic) και τη φάση του κύκλου ζωής (Phase), (β) Ερευνητική Υλοποίηση (Research Implementation), που περιλαμβάνει την πλατφόρμα (Platform), τα εργαλεία και τον βαθμό αυτοματοποίησης (Tool and Automation) και τη μέθοδο αξιολόγησης (Evaluation Method), (γ) Ποιότητα (Quality), που περιλαμβάνει ευπάθειες και απειλές (Vulnerability and Threat), χαρακτηριστικά ασφάλειας (Security Characteristic) και μετρήσεις ασφάλειας (Security Measurement), (δ) Μοτίβα (Pattern), που διακρίνει τα μοτίβα επίθεσης (Attack Pattern) και τα μοτίβα ασφάλειας (Security Pattern), και (ε) Μέθοδος (Method), που περιλαμβάνει τη μεθοδολογία (Methodology), τη σημειογραφία μοντελοποίησης των μοτίβων (Pattern Modeling) και τις σχέσεις μεταξύ μοτίβων (Relationship between Patterns).

Οι παραπάνω διαστάσεις ταξινομούν πρωτίστως ερευνητικές εργασίες γύρω από τα μοτίβα ασφάλειας και, επομένως, δεν αποτελούν όλες απαραίτητα χαρακτηριστικά που πρέπει να περιλαμβάνονται στην περιγραφή ενός μεμονωμένου μοτίβου. Για την παρούσα εργασία ιδιαίτερο ενδιαφέρον παρουσιάζουν οι διαστάσεις που μπορούν να αποδοθούν ως σημασιολογικά χαρακτηριστικά των ίδιων των μοτίβων, όπως η φάση ανάπτυξης, οι απειλές, οι ιδιότητες ασφάλειας και οι σχέσεις μεταξύ μοτίβων. Οι πτυχές αυτές λαμβάνονται υπόψη στο σημασιολογικό μοντέλο της εφαρμογής, ενώ χαρακτηριστικά που αφορούν κυρίως την ερευνητική υλοποίηση μιας δημοσίευσης, όπως η μέθοδος αξιολόγησης ή ο βαθμός αυτοματοποίησης ενός ερευνητικού εργαλείου, βρίσκονται εκτός του σκοπού της οντολογικής περιγραφής των μοτίβων.

Έχουν διεξαχθεί αρκετές έρευνες σχετικά με συγκεκριμένα πρότυπα ασφάλειας και σχετικά μοτίβα για την ταξινόμηση και ανάλυσή τους. Οι Yoshioka et al. (2008) διεξήγαγαν μια έρευνα για τα πρότυπα ασφαλείας γενικά. Ωστόσο, ο αριθμός των προτύπων που μελετήθηκαν ήταν περιορισμένος καθώς η έρευνα ήταν πρώιμη και πραγματοποιήθηκε το 2008. Ο Uzunov (2016) παρουσίασε μια εκτενή επισκόπηση λύσεων ασφάλειας για κατανεμημένα συστήματα δημοσίευσης/συνδρομής (publish-subscribe distributed systems), εξετάζοντας 13 συγκεκριμένες

αρχιτεκτονικές λύσεις και 8 αφηρημένες λύσεις. Η μελέτη ανέλυσε επίσης τα επιμέρους μοτίβα ασφάλειας και τις απειλές που αντιμετωπίζονται από τις εξεταζόμενες λύσεις, αναδεικνύοντας υπάρχουσες καλύψεις και ερευνητικά κενά. Οι Washizaki et al. (2020) πραγματοποίησαν συστηματική βιβλιογραφική ανασκόπηση των αρχιτεκτονικών και σχεδιαστικών μοτίβων για συστήματα IoT, από την οποία εξήγαγαν 143 μοτίβα από 32 δημοσιεύσεις. Μεταξύ των αποτελεσμάτων διαπιστώθηκε ότι το 57% των μοτίβων δεν ήταν ειδικά για IoT, ενώ η ασφάλεια συγκαταλεγόταν, μαζί με τη συμβατότητα και τη συντηρησιμότητα, στα χαρακτηριστικά ποιότητας που εμφανίζονταν συχνά στα εξεταζόμενα μοτίβα. Οι Laverdière et al. (2006) διεξήγαγαν άλλη μία ολοκληρωμένη έρευνα προτύπων ασφαλείας χρήσιμων στη φάση του σχεδιασμού. Οι παραπάνω έρευνες έχουν διαφορετικό και περιορισμένο ως προς τον σκοπό τους πεδίο: οι Yoshioka et al. (2008) εξετάζουν προσεγγίσεις χρήσης μοτίβων ασφάλειας κατά την ανάπτυξη λογισμικού, ο Uzunov (2016) εστιάζει ειδικά σε κατανεμημένα συστήματα publish/subscribe, οι Washizaki et al. (2020) σε αρχιτεκτονικά και σχεδιαστικά μοτίβα για IoT και οι Laverdière et al. (2006) κυρίως σε μοτίβα ασφάλειας που εφαρμόζονται κατά τις φάσεις αρχιτεκτονικής και σχεδιασμού. Επομένως, καμία από τις παραπάνω μελέτες δεν καλύπτει πλήρως το σύνολο της έρευνας γύρω από τα μοτίβα ασφάλειας σε όλες τις φάσεις του κύκλου ζωής, καθώς καθεμία εστιάζει σε συγκεκριμένο υποσύνολο μοτίβων, τεχνολογικό πεδίο ή φάση ανάπτυξης.

Εκτός από τις έρευνες συγκεκριμένων ή γενικών προτύπων ασφάλειας, υπάρχουν κάποιες υπάρχουσες εργασίες για τη μελέτη διαφορετικών ερευνητικών τεχνικών και προσεγγίσεων για την ταξινόμηση, μοντελοποίηση, εφαρμογή και αξιολόγηση των μοτίβων ασφάλειας. Οι Alvi & Zulkernine (2012) διεξήγαγαν μελέτη για τη σύγκριση διαφόρων σχημάτων ταξινόμησης προτύπων ασφαλείας. Η συγκεκριμένη μελέτη περιορίζεται στη σύγκριση σχημάτων ταξινόμησης ως προς τους στόχους, τα χαρακτηριστικά, τις διαστάσεις και τα κριτήρια ποιότητάς τους. Ζητήματα όπως η εφαρμογή και η τυποποίηση των μοτίβων βρίσκονται εκτός της εμβέλειας της συγκεκριμένης έρευνας των Alvi & Zulkernine (2012) και δεν αξιολογούνται από τους συγγραφείς.

Οι Rajmohan et al. (2020) πραγματοποίησαν συστηματική χαρτογράφηση περισσότερων από χίλιες αρχικά εντοπισμένων δημοσιεύσεων και, μετά τη διαδικασία επιλογής, ανέλυσαν 24 μελέτες σχετικές με μοτίβα και αρχιτεκτονικές για την ασφάλεια και το απόρρητο του IoT. Τα αποτελέσματα έδειξαν αυξητική τάση των σχετικών δημοσιεύσεων κατά τα τελευταία έτη της περιόδου που εξετάστηκε, αλλά και την ύπαρξη ερευνητικών κενών που περιορίζουν την ευρύτερη αξιοποίηση μοτίβων για την ασφάλεια και την ιδιωτικότητα των συστημάτων IoT. Οι Ito et al. (2015) διεξήγαγαν συστηματική μελέτη χαρτογράφησης με στόχο την έρευνα προτύπων ασφάλειας. Ωστόσο, ο αριθμός των ερευνητικών εργασιών που ερευνήθηκαν περιορίστηκε σε 30 μόνο και αποτελεί μια σύντομη περίληψη της έρευνας προτύπων χωρίς εις βάθος ανάλυση.

Οι απαιτήσεις ασφάλειας προσδιορίζουν τι πρέπει να προστατεύεται ή ποιες ιδιότητες ασφάλειας πρέπει να ικανοποιούνται, χωρίς να καθορίζουν από μόνες τους τη σχεδιαστική ή υλοποιητική λύση. Τα μοτίβα ασφάλειας μπορούν να χρησιμοποιηθούν στη συνέχεια ως

επαναχρησιμοποιήσιμες λύσεις που συμβάλλουν στην ικανοποίηση των αντίστοιχων απαιτήσεων. Οι Guan et al. (2016) επισημαίνουν ότι η έλλειψη μιας τυπικά καθορισμένης σημασιολογικής αναπαράστασης των μοτίβων δυσχεραίνει την υποβολή σύνθετων ερωτημάτων και την ανάκτηση σημασιολογικά κατάλληλων απαντήσεων από ένα σύστημα αναζήτησης. Για την αντιμετώπιση του προβλήματος προτείνουν μια οντολογική προσέγγιση που συνδέει τις απαιτήσεις ασφάλειας με τα κατάλληλα μοτίβα ασφάλειας.

Η προτεινόμενη οντολογία ασφάλειας υλοποιήθηκε σε OWL-DL και αποτελείται από δύο υπο-οντολογίες: την υπο-οντολογία απαιτήσεων ασφάλειας (security requirement subontology), η οποία περιλαμβάνει έννοιες όπως Asset, Threat, Vulnerability, Attribute και Priority, και την υπο-οντολογία μοτίβων ασφάλειας (security pattern subontology), η οποία περιλαμβάνει τις βασικές έννοιες Security Context, Security Problem και Security Solution. Για την ανάπτυξη και τον έλεγχο της οντολογίας χρησιμοποιήθηκαν το Protégé και ο reasoner FaCT++.

Η οντολογία ενσωματώθηκε σε μια μηχανή αναζήτησης μοτίβων ασφάλειας, η οποία χρησιμοποιεί το OWL API και έναν αλγόριθμο αντιστοίχισης μεταξύ απαιτήσεων και μοτίβων. Η μηχανή διαθέτει λειτουργίες εισόδου, συμπερασμού, αναζήτησης και εξόδου και αξιοποιεί τις σημασιολογικές σχέσεις της οντολογίας για να εντοπίζει μοτίβα ασφαλείας που αντιστοιχούν σε χαρακτηριστικά μιας απαίτησης, όπως το προστατευόμενο αγαθό, η απειλή, η ιδιότητα ασφάλειας και ο τομέας εφαρμογής. Οι συγγραφείς αξιολόγησαν τη δυνατότητα αυτή και μέσω ερωτημάτων Description Logic (DL queries), όπως η αναζήτηση των μοτίβων που μπορούν να προστατεύσουν ευαίσθητα δεδομένα από συγκεκριμένη απειλή. Με τον τρόπο αυτό, η προσέγγιση υποστηρίζει σημασιολογική αναζήτηση και αυτοματοποιημένη επιλογή υποψήφιων μοτίβων ασφάλειας με βάση τις δηλωμένες απαιτήσεις.

Η ανάπτυξη ενός συστήματος που ικανοποιεί τις απαιτήσεις ασφάλειας απαιτεί συχνά εξειδικευμένη τεχνογνωσία, η οποία δεν είναι πάντοτε διαθέσιμη σε όλους τους σχεδιαστές και προγραμματιστές λογισμικού. Η δυσκολία αυτή αυξάνεται καθώς τα συστήματα γίνονται μεγαλύτερα και περισσότερο σύνθετα. Στο πλαίσιο αυτό, η αξιοποίηση καταγεγραμμένης και επαναχρησιμοποιήσιμης γνώσης ασφάλειας μπορεί να υποστηρίξει τους λιγότερο έμπειρους επαγγελματίες κατά τη λήψη σχεδιαστικών αποφάσεων. Τα μοτίβα ασφάλειας συμβάλλουν προς αυτή την κατεύθυνση, καθώς επιτρέπουν την αξιοποίηση καθιερωμένων πρακτικών και τεκμηριωμένων λύσεων κατά την ανάπτυξη ασφαλών συστημάτων λογισμικού Schumacher (2003).

2.3.3 Ανίχνευση Μοτίβων Ασφάλειας

Οι Alvi και Zulkernine (2021) πρότειναν ένα πλαίσιο ανίχνευσης μοτίβων ασφάλειας (Security Pattern Detection – SPD), με στόχο τον εντοπισμό ήδη υλοποιημένων μοτίβων ασφάλειας μέσα σε υπάρχοντα συστήματα λογισμικού. Η προσέγγιση βασίζεται στην εξαγωγή των δομικών σχέσεων μεταξύ των κλάσεων του υπό εξέταση συστήματος και στη σύγκρισή τους με τις αντίστοιχες σχέσεις γνωστών μοτίβων ασφάλειας.

Το πλαίσιο χρησιμοποιεί δύο βασικές τεχνικές αντιστοίχισης, την Ordered Matrix Matching (OMM) και τη Non-uniform Distributed Matrix Matching (NDMM). Η OMM επιχειρεί να εντοπίσει τη δομή ενός μοτίβου ασφάλειας μέσα στη δομική αναπαράσταση του εξεταζόμενου συστήματος, ενώ η NDMM επιτρέπει περισσότερο ευέλικτη αντιστοίχιση των σχέσεων μεταξύ των κλάσεων ενός μοτίβου και των κλάσεων του συστήματος. Επιπλέον, χρησιμοποιείται σημασιολογική ανάλυση με σκοπό τη μείωση των ψευδώς θετικών αποτελεσμάτων.

Η προσέγγιση δεν αποσκοπεί στην επιλογή του καταλληλότερου μοτίβου ασφάλειας για ένα νέο πρόβλημα, αλλά στην αναγνώριση μοτίβων που έχουν ήδη υλοποιηθεί σε ένα υπάρχον σύστημα. Ειδικότερα, μπορεί να προσδιορίσει τις κλάσεις και τις μεταξύ τους σχέσεις που αντιστοιχούν στη δομή ενός συγκεκριμένου μοτίβου ασφάλειας.

Η εργασία αυτή σχετίζεται με την παρούσα έρευνα επειδή αξιοποιεί τη δομημένη περιγραφή των μοτίβων ασφάλειας ώστε να διευκολύνει την αυτοματοποιημένη επεξεργασία τους. Ωστόσο, διαφοροποιείται ως προς τον κύριο στόχο. Η προσέγγιση των Alvi και Zulkernine επικεντρώνεται στην ανίχνευση ήδη υλοποιημένων μοτίβων μέσα σε συστήματα λογισμικού, ενώ η παρούσα εργασία επικεντρώνεται στη σημασιολογική περιγραφή, οργάνωση και αναζήτηση μοτίβων ασφάλειας μέσω οντολογίας, ώστε οι χρήστες να μπορούν να εντοπίζουν και να αξιοποιούν τη σχετική γνώση.

2.3.4 Αποθετήρια και Βάσεις Γνώσεις Μοτίβων Ασφάλειας

Το “Security Patterns Repository Version 1.0” των Kienzle et al. (2002) αποτελεί έναν τεχνικό κατάλογο μοτίβων ασφάλειας, ο οποίος επικεντρώνεται στην ασφάλεια εφαρμογών Ιστού και περιλαμβάνει 26 μοτίβα και 3 mini-patterns. Για την περιγραφή των μοτίβων χρησιμοποιείται κοινό δομημένο πρότυπο (template), το οποίο περιλαμβάνει τα πεδία Name, Abstract, Problem, Solution, Issues, Trade-Offs, Related Patterns και References, ενώ τα mini-patterns περιγράφονται συνοπτικότερα μέσω Problem και Solution. Η συγκεκριμένη εργασία δεν παρουσιάζει κάποιο αυτοματοποιημένο σύστημα ή σημασιολογική τεχνολογία για την αναζήτηση και ανακάλυψη των μοτίβων· τα μοτίβα οργανώνονται ως καταχωρίσεις του καταλόγου και διακρίνονται κυρίως σε δομικά και διαδικαστικά (Kienzle et al., 2002). Τα δομικά μοτίβα είναι μοτίβα που μπορούν να εφαρμοστούν σε μια εφαρμογή. Περιλαμβάνουν σχεδιαστικά μοτίβα, αλλά μπορούν επίσης να εφαρμοστούν σε αρχιτεκτονικό επίπεδο ή επίπεδο υλοποίησης. Τα διαδικαστικά μοτίβα είναι μοτίβα που μπορούν να χρησιμοποιηθούν για τη βελτίωση της διαδικασίας ανάπτυξης λογισμικού κρίσιμου ως προς την ασφάλεια/προστασία και συχνά επηρεάζουν την οργάνωση ή τη διαχείριση ενός έργου (ανάπτυξης λογισμικού).

Οι Pereira Vale και Fernández (2019) προτείνουν μια οντολογική προσέγγιση για τη σημασιολογική περιγραφή των μοτίβων ασφάλειας. Η προσέγγιση επεκτείνει την περιγραφή των μοτίβων με σημασιολογικά στοιχεία που αφορούν, μεταξύ άλλων, το πλαίσιο εφαρμογής, το πρόβλημα, τη λύση, τη φάση του κύκλου ζωής και το αρχιτεκτονικό επίπεδο. Οι συγγραφείς

ανέπτυξαν την αντίστοιχη οντολογία σε OWL και παρουσίασαν παραδείγματα ερωτημάτων για την αξιοποίησή της, με στόχο τη διευκόλυνση της ακριβέστερης επιλογής και αναζήτησης σχετικών μοτίβων ασφάλειας. Η δημιουργία ενός πλήρους καταλόγου, στον οποίο κάθε μοτίβο θα συνοδεύεται από οντολογική περιγραφή, παρουσιάζεται ως μελλοντικός στόχος της προσέγγισης και όχι ως αποτέλεσμα που υλοποιήθηκε στην συγκεκριμένη εργασία.

Η οντολογία προτύπων ασφαλείας θα επιτρέψει την εφαρμογή ενός σημασιολογικού wiki που μπορεί να χρησιμοποιηθεί τόσο από δημιουργούς προτύπων όσο και από προγραμματιστές λογισμικού. Με αυτόν τον τρόπο, είναι δυνατή η διαχείριση των αναφερόμενων μοτίβων ασφαλείας. Έτσι, οι αναζητήσεις από τους προγραμματιστές, για το ποιο μοτίβο ασφαλείας είναι κατάλληλο να εφαρμοστεί σε ορισμένες περιπτώσεις κατά τη διάρκεια της μεθοδολογίας ανάπτυξης ασφαλών συστημάτων θα είναι αυτοματοποιημένες, εύκολες και γρήγορες. Η χρήση τεχνολογιών σημασιολογικού Ιστού (Semantic Web technologies) επιτρέπει την προσθήκη σημασιολογικού περιεχομένου στα δεδομένα που σχετίζονται με τα μοτίβα ασφαλείας, επιτρέποντας την οργάνωση των προσφερόμενων πληροφοριών, οι οποίες είναι κατάλληλες για αναζητήσεις σχετικά με αυτά σημασιολογικά.

Όπως αναφέρθηκε και στο κεφάλαιο που αφορά την OWL, η OWL χρησιμοποιήθηκε από τους Pereira Vale και Fernández (2019) για την υλοποίηση της οντολογίας που προτείνουν, λόγω της εκφραστικότητάς της που επιτρέπει την εφαρμογή των λογιστών (reasoners) FaCT++, Racer ή Pellet, οι οποίοι μπορούν να χρησιμοποιηθούν για τον έλεγχο της λογικής συνέπειας μιας οντολογίας, την ταξινόμηση των κλάσεών της και την εξαγωγή συμπερασμάτων από τα δηλωμένα αξιώματα.

Οι Khoury et al. (2008) ανέφεραν μια προσέγγιση που βασίζεται στη χρήση οντολογιών ώστε να βρεθεί ένα κατάλληλο σύνολο μοτίβων ασφαλείας που ζητούνται από τους προγραμματιστές λογισμικού. Η εργασία τους βασίστηκε στην επίσημη εννοιολογική περιγραφή των ιδιοτήτων ασφαλείας που παρουσιάστηκε στο Serenity (Maña & Pujol, 2008), αλλά δεν ανέφεραν τον τρόπο με τον οποίο θα ορίζονταν και θα χρησιμοποιούνταν τα μοτίβα ασφαλείας. Μάλιστα, η οντολογία που προτάθηκε βασίστηκε περισσότερο σε έννοιες που σχετίζονται με απαιτήσεις ασφάλειας, μοντέλα απειλών και σφάλματα ασφαλείας.

3 *Προτεινόμενη λύση*

Η προτεινόμενη λύση της παρούσας εργασίας αφορά την ανάπτυξη ενός ολοκληρωμένου σημασιολογικού πλαισίου ιστού (semantic web framework) για την περιγραφή, οργάνωση, τεκμηρίωση, αναζήτηση και διαχείριση μοτίβων ασφάλειας λογισμικού. Η βασική σχεδιαστική επιλογή είναι ότι ένα μοτίβο ασφάλειας δεν αντιμετωπίζεται αποκλειστικά ως ανεξάρτητο κείμενο ή ως εγγραφή ενός συμβατικού αποθετηρίου, αλλά ως οντότητα γνώσης η οποία διαθέτει συγκεκριμένη σημασιολογική ταυτότητα και συνδέεται με άλλες έννοιες του πεδίου της ασφάλειας. Η προσέγγιση αυτή ακολουθεί τη γενικότερη κατεύθυνση της σχετικής έρευνας, σύμφωνα με την οποία η τυπική περιγραφή και ταξινόμηση των μοτίβων διευκολύνει την οργάνωση και επαναχρησιμοποίηση της γνώσης ασφάλειας και την επιλογή κατάλληλων λύσεων για επαναλαμβανόμενα προβλήματα (Guan, Yang, & Wang, 2016; Jafari & Rasoolzadegan, 2018; Pereira Vale & Fernández, 2019). Η εργασία των Guan, Yang, & Wang, 2016 χρησιμοποιεί επίσης την OWL για τη σημασιολογική περιγραφή και ανάκτηση security patterns, ενώ οι Pereira Vale και Fernández (2019) προτείνουν ρητά οντολογική αναπαράσταση των στοιχείων που περιβάλλουν ένα security pattern.

Η ανάπτυξη της λύσης πραγματοποιήθηκε σε διαδοχικά επίπεδα. Αρχικά μελετήθηκε η βιβλιογραφία γύρω από τη δομή, την ταξινόμηση και τη χρήση των μοτίβων ασφάλειας, ώστε να προσδιοριστούν τα στοιχεία που είναι χρήσιμα για τη συστηματική περιγραφή τους. Η βιβλιογραφική ανάλυση έδειξε ότι ένα μοτίβο δεν αρκεί να χαρακτηρίζεται μόνο από ένα όνομα και μια λύση, αλλά είναι χρήσιμο να αποτυπώνονται στοιχεία όπως η πρόθεσή του, το πλαίσιο εφαρμογής, το πρόβλημα, οι δυνάμεις που επηρεάζουν τη λύση, οι συνέπειες, οι απειλές που αντιμετωπίζει, τα προστατευόμενα αγαθά, οι ιδιότητες ασφάλειας που υποστηρίζει και η θέση του στον κύκλο ζωής ανάπτυξης λογισμικού. Παράλληλα, η ύπαρξη μεγάλου αριθμού μοτίβων και διαφορετικών σχημάτων ταξινόμησης στη βιβλιογραφία καθιστά αναγκαία τη χρήση ενός συνεπούς σημασιολογικού μοντέλου για την οργάνωσή τους (Jafari & Rasoolzadegan, 2020).

Με βάση την ανάλυση αυτή σχεδιάστηκε η οντολογία της εργασίας σε OWL 2. Η OWL επιτρέπει την αναπαράσταση κλάσεων, ιδιοτήτων, ατόμων και δεδομένων με τυπικά καθορισμένη σημασιολογία και επομένως προσφέρει την απαιτούμενη υποδομή για τη μετατροπή της περιγραφής των μοτίβων σε μηχανικά επεξεργάσιμη γνώση (W3C OWL Working Group, 2012). Η ανάπτυξη και η επιθεώρηση του μοντέλου πραγματοποιήθηκαν στο περιβάλλον Protégé, ένα ευρέως χρησιμοποιούμενο εργαλείο ανάπτυξης και συντήρησης οντολογιών (Musen, 2015). Στο Protégé δημιουργήθηκε η ιεραρχία των κλάσεων, ορίστηκαν οι ιδιότητες της οντολογίας,

δημιουργήθηκαν τα απαραίτητα individuals και εξετάστηκε ο τρόπος με τον οποίο αυτά συνδέονται μεταξύ τους.

Κεντρική κλάση του μοντέλου είναι η SecurityPattern. Κάτω από αυτή οργανώθηκε ιεραρχία περισσότερο εξειδικευμένων κατηγοριών μοτίβων, έτσι ώστε ένα συγκεκριμένο pattern να μπορεί να ανήκει σε μία ή περισσότερες κατηγορίες μέσω του rdf:type, ενώ παράλληλα να αναγνωρίζεται ως security pattern μέσω της μεταβατικής ιεραρχίας rdfs:subClassOf. Η επιλογή αυτή είναι ουσιωδώς διαφορετική από την αποθήκευση μιας κατηγορίας ως απλού κειμένου: η κατηγορία αποτελεί μέρος του σημασιολογικού μοντέλου και μπορεί να χρησιμοποιηθεί από επερωτήσεις και μηχανισμούς συλλογισμού. Η εφαρμογή ανακτά την ιεραρχία αυτή από την οντολογία και δεν βασίζεται σε στατικά hard-coded ονόματα κατηγοριών για την παρουσίασή της (Queries.cs, SecurityPatternsService.cs).

Εκτός από την SecurityPattern, δημιουργήθηκαν αυτοτελείς σημασιολογικές κλάσεις για έννοιες που είναι χρήσιμο να διαθέτουν δική τους ταυτότητα μέσα στη βάση γνώσης. Σε αυτές περιλαμβάνονται οι Threat, ThreatCategory, ProtectedAsset, SecurityProperty, DevelopmentPhase, RiskLevel, ApplicationContext, PatternLevel, ValidationStatus και BibliographicSource. Έτσι, για παράδειγμα, μια απειλή δεν αποθηκεύεται απλώς ως το λεκτικό «Identity Spoofing», αλλά μπορεί να αποτελεί συγκεκριμένο individual της κλάσης Threat, το οποίο είναι δυνατό να συσχετιστεί με περισσότερα από ένα security patterns. Αντίστοιχα, μια βιβλιογραφική πηγή αποτελεί ξεχωριστή οντότητα με στοιχεία όπως τίτλος, συγγραφείς, έτος, τύπος, DOI και URL. Η αντίστοιχη δομή της τελικής εφαρμογής αποτυπώνεται και στους σταθερούς URI που χρησιμοποιούνται από το Data Access Layer (ConstantUris.cs).

Κατά τη μοντελοποίηση έγινε σαφής διάκριση μεταξύ Datatype Properties και Object Properties. Οι Datatype Properties χρησιμοποιούνται για πληροφορίες περιγραφικού χαρακτήρα, όπως hasName, hasAlias, hasDescription, hasIntent, hasContext, hasProblem, hasForces, hasSolution, hasConsequence και hasImplementationExample. Αντίθετα, πληροφορίες που αντιστοιχούν σε έννοιες με αυτοτελή σημασιολογική ταυτότητα μοντελοποιήθηκαν μέσω Object Properties. Ενδεικτικά, οι mitigatesThreat, protectsAsset, supportsSecurityProperty, appliesAtPhase, hasRiskLevel, hasApplicationContext, hasPatternLevel και hasValidationStatus συνδέουν ένα μοτίβο με αντίστοιχα individuals της οντολογίας. Επιπλέον, μοντελοποιήθηκαν σχέσεις μεταξύ των ίδιων των μοτίβων, όπως relatedPattern, usesPattern, refinesPattern και alternativeToPattern, επιτρέποντας στην οντολογία να αποτυπώνει μέρος της δομής μιας γλώσσας μοτίβων και όχι μόνο έναν επίπεδο κατάλογο εγγραφών. Για τις Object Properties ορίστηκαν κατάλληλα domain και range, όπως για τη mitigatesThreat, η οποία συνδέει μια SecurityPattern με μια Threat.

Ιδιαίτερη βαρύτητα δόθηκε επίσης στην προέλευση και τεκμηρίωση της γνώσης. Για τον σκοπό αυτό δημιουργήθηκε η κλάση BibliographicSource και η σχέση documentedInSource, με αντίστροφη σχέση documentsPattern. Παράλληλα, στο περιγραφικό μέρος ενός pattern μπορούν να

καταγράφονται στοιχεία όπως hasSourceCitation, hasOriginalSourceCitation, hasCatalogCitation, hasSourcePage, hasSourceUrl και hasValidationNote. Με τον τρόπο αυτό διαχωρίζεται το ίδιο το περιεχόμενο ενός μοτίβου από την πληροφορία σχετικά με την προέλευση και τον βαθμό τεκμηρίωσής του, ενώ παρέχεται δυνατότητα συσχέτισης πολλών patterns με κοινές βιβλιογραφικές πηγές. Η τελική εφαρμογή ανακτά τις βιβλιογραφικές πηγές ως ξεχωριστές σημασιολογικές οντότητες και τις ενσωματώνει στο μοντέλο SecurityPattern (SecurityPattern.cs, SecurityPatternsService.cs).

Αφού ορίστηκε το οντολογικό σχήμα, ακολούθησε ο εμπλουτισμός της βάσης γνώσης με πραγματικά security patterns. Τα μοτίβα δημιουργήθηκαν ως named individuals της οντολογίας και όχι ως απλές γραμμές δεδομένων. Για κάθε pattern καταγράφηκαν τα διαθέσιμα περιγραφικά στοιχεία και, όπου υπήρχαν αντίστοιχες σημασιολογικές έννοιες, δημιουργήθηκαν οι σχετικές Object Property συνδέσεις προς απειλές, κατηγορίες απειλών, αγαθά, ιδιότητες ασφάλειας, φάσεις ανάπτυξης, επίπεδα κινδύνου, πλαίσια εφαρμογής, επίπεδα μοτίβου, άλλα patterns και βιβλιογραφικές πηγές. Ένα pattern μπορεί επομένως να διαθέτει περισσότερες από μία ταξινομήσεις και περισσότερες από μία συσχετίσεις του ίδιου είδους. Η δομή αυτή μεταφέρεται αυτούσια και στο αντικειμενοστραφές μοντέλο της εφαρμογής, όπου οι σημασιολογικές σχέσεις αναπαρίστανται ως συλλογές OntologyReference και όχι ως μία συμβολοσειρά (SecurityPattern.cs, SemanticPatternInput.cs).

Το περιεχόμενο της βιβλιοθήκης των patterns συγκροτήθηκε με συνδυασμό της θεωρητικής ανάλυσης της διπλωματικής, της μελέτης πηγών και καταλόγων μοτίβων ασφάλειας και γνώσης ασφαλούς ανάπτυξης λογισμικού. Οι παραγόμενες περιγραφές προσαρμόστηκαν στη δομή της οντολογίας, συσχετίστηκαν με τις κατάλληλες σημασιολογικές έννοιες και ενσωματώθηκαν στη βάση γνώσης μετά από έλεγχο της συμβατότητάς τους με το μοντέλο. Η διαδικασία αυτή καταγράφεται και στην τεχνική τεκμηρίωση του repository, σύμφωνα με την οποία η βιβλιοθήκη προέκυψε από συνδυασμό της θεωρίας της εργασίας, γνώσης ασφαλούς σχεδίασης, OWL/RDF δομής, σύνταξης και χειροκίνητης αναθεώρησης.

Χαρακτηριστικό παράδειγμα της τελικής σημασιολογικής αναπαράστασης αποτελεί το individual AuthenticationEnforcer. Το συγκεκριμένο μοτίβο ταξινομείται σε περισσότερες από μία κλάσεις μοτίβων και συνδέεται, μέσω Object Properties, με συγκεκριμένες απειλές, προστατευόμενα αγαθά, ιδιότητες ασφάλειας, φάσεις ανάπτυξης, πλαίσια εφαρμογής, επίπεδα μοτίβου και βιβλιογραφικές πηγές. Ταυτόχρονα, οι λεκτικές πληροφορίες που αφορούν την πρόθεση, το πλαίσιο, το πρόβλημα, τις δυνάμεις, τη λύση και τις συνέπειες διατηρούνται ως Datatype Properties. Το παράδειγμα αυτό αποτυπώνει τον βασικό στόχο του μοντέλου: τη συνύπαρξη της αναγνώσιμης από άνθρωπο περιγραφής και της μηχανικά επεξεργάσιμης σημασιολογικής πληροφορίας στο ίδιο pattern. Η συγκεκριμένη αναπαράσταση έχει ήδη αποτυπωθεί και ελεγχθεί στο Protégé.

Μετά τον εμπλουτισμό της οντολογίας ακολούθησε ξεχωριστό στάδιο σημασιολογικής επικύρωσης, πριν από τη σύνδεσή της με την εφαρμογή. Η οντολογία φορτώθηκε στο Protégé και επιθεωρήθηκαν η asserted ιεραρχία των κλάσεων, η ιεραρχία των Object Properties, οι δηλώσεις domain και range και αντιπροσωπευτικά individuals. Για παράδειγμα, ελέγχθηκε ότι η mitigatesThreat έχει ως domain τη SecurityPattern και ως range τη Threat, καθώς και ότι η documentedInSource συνδέει SecurityPattern με BibliographicSource. Η οπτική επιθεώρηση της δομής πραγματοποιήθηκε επίσης μέσα από τα εργαλεία παρουσίασης της οντολογίας του Protégé.

Στη συνέχεια εκτελέστηκε ο HermiT reasoner, ώστε να ελεγχθεί η λογική συνέπεια της οντολογίας και να παραχθεί η inferred ιεραρχία των κλάσεων. Ο HermiT είναι reasoner για OWL 2 που υποστηρίζει, μεταξύ άλλων, έλεγχο συνέπειας και ταξινόμηση κλάσεων και ιδιοτήτων (Glimm et al., 2014). Με τον τρόπο αυτό ο έλεγχος της οντολογίας πραγματοποιήθηκε ως ανεξάρτητο στάδιο και δεν θεωρήθηκε ότι η επιτυχής λειτουργία της εφαρμογής αποτελεί από μόνη της απόδειξη σημασιολογικής ορθότητας. Η διάκριση μεταξύ ontology validation και application validation είναι ουσιώδης στη μεθοδολογία της εργασίας. Στο υπάρχον κείμενο έχει ήδη αποτυπωθεί η inferred ιεραρχία που προέκυψε μετά την ταξινόμηση με τον HermiT.

Αφού ολοκληρώθηκε ο έλεγχος του σημασιολογικού μοντέλου, η επικυρωμένη οντολογία ενσωματώθηκε στην εφαρμογή ως η κεντρική βάση γνώσης. Η εφαρμογή αναπτύχθηκε σε ASP.NET Core MVC πάνω σε .NET 8 και οργανώθηκε σε διακριτά επίπεδα διεπαφής χρήστη, επιχειρησιακής λογικής και πρόσβασης στα δεδομένα. Το Data Access Layer χρησιμοποιεί τη βιβλιοθήκη dotNetRDF για τη φόρτωση του security_patterns.owl σε RDF graph, τη δημιουργία in-memory dataset και την εκτέλεση SPARQL επερωτήσεων. Το αρχείο OWL αποτελεί την canonical πηγή δεδομένων της εφαρμογής και η θέση του παρέχεται μέσω της ρύθμισης OwlFilePath, η οποία επιλύεται κατά την εκκίνηση του UI project (Program.cs, appsettings.json, SparqlAdapter.cs).

Πάνω από τη σημασιολογική βάση αναπτύχθηκε διεπαφή ιστού που επιτρέπει στον τελικό χρήστη να αξιοποιεί την οντολογία χωρίς να χρειάζεται να επεξεργάζεται απευθείας OWL/RDF. Ο χρήστης μπορεί να εμφανίζει τη βιβλιοθήκη των patterns, να αναζητά εγγραφές, να εξετάζει την πλήρη σημασιολογική περιγραφή ενός μοτίβου και να δημιουργεί, επεξεργάζεται ή διαγράφει patterns. Κατά τη δημιουργία και την επεξεργασία, οι διαθέσιμες κατηγορίες, απειλές, επίπεδα κινδύνου, προστατευόμενα αγαθά, ιδιότητες ασφάλειας, φάσεις ανάπτυξης, πλαίσια εφαρμογής, επίπεδα μοτίβου, καταστάσεις επικύρωσης, σχετικά patterns και βιβλιογραφικές πηγές ανακτώνται από την ίδια την οντολογία. Έτσι, ο χρήστης δεν εισάγει αυθαίρετα κείμενα για τις σημασιολογικές σχέσεις, αλλά επιλέγει υπάρχουσες οντολογικές οντότητες. Η σχετική ροή υλοποιείται μέσω των SemanticPatternInput, SecurityPatternsController, SecurityPatternsService και SecurityPatternsRepository.

Η δημιουργία ενός νέου pattern από την εφαρμογή δεν αντιστοιχεί επομένως σε εισαγωγή μιας απλής εγγραφής. Δημιουργείται νέο named individual με δικό του IRI, προστίθενται οι

επιλεγμένες class assertions, καταγράφονται οι Datatype Properties και δημιουργούνται οι αντίστοιχες Object Property assertions προς τα ήδη υπάρχοντα individuals της οντολογίας. Αντίστοιχα, η επεξεργασία ενός pattern αντικαθιστά ελεγχόμενα τις σχετικές δηλώσεις της οντολογίας. Οι μεταβολές αποθηκεύονται πίσω στο ίδιο OWL αρχείο και ενημερώνεται παράλληλα το in-memory RDF graph που χρησιμοποιεί η εφαρμογή (SparqlAdapter.AddSemanticPattern, EditSemanticPattern).

Η διαγραφή σχεδιάστηκε ως λογική διαγραφή (soft delete) και όχι ως φυσική αφαίρεση όλων των RDF/OWL δηλώσεων του μοτίβου. Στην οντότητα προστίθεται η ιδιότητα isDeleted με τιμή true και διατηρείται σχετικό σχόλιο με βασικές προηγούμενες τιμές. Οι κανονικές SPARQL επερωτήσεις της εφαρμογής εξαιρούν τις διαγραμμένες οντότητες. Αντίστοιχα, κατά την επεξεργασία διατηρείται σχόλιο με προηγούμενες τιμές, παρέχοντας ένα βασικό ίχνος μεταβολών χωρίς να επιχειρείται η υλοποίηση πλήρους συστήματος version control.

Πέρα από τη γραφική διεπαφή, η τελική λύση παρέχει και ξεχωριστή διεπαφή εκτέλεσης SPARQL, ώστε ένας πιο εξειδικευμένος χρήστης να μπορεί να διερευνήσει απευθείας τη σημασιολογική βάση γνώσης. Η λειτουργία έχει σχεδιαστεί ως read-only: γίνονται δεκτές μόνο επερωτήσεις της οικογένειας SELECT, απαγορεύονται SERVICE, FROM και FROM NAMED, ενώ εφαρμόζεται ανώτατο όριο στον αριθμό των αποτελεσμάτων. Με αυτόν τον τρόπο παρέχεται δυνατότητα άμεσης σημασιολογικής διερεύνησης της οντολογίας χωρίς να επιτρέπεται η τροποποίησή της μέσω αυθαίρετων SPARQL Update εντολών (SparqlController.cs, SparqlAdapter.ExecuteSelectQuery).

Η συνολική προτεινόμενη λύση, επομένως, δεν αποτελεί απλώς μια εφαρμογή διαχείρισης καταλόγου μοτίβων. Αποτελεί μια πλήρη ροή από την απόκτηση και οργάνωση γνώσης, στη σημασιολογική μοντελοποίηση σε OWL, στην κατασκευή και τεκμηρίωση πραγματικών security pattern individuals, στην ανεξάρτητη επικύρωση της οντολογίας στο Protégé με τον HermiT, και τελικά στην αξιοποίηση της επικυρωμένης γνώσης από μια εφαρμογή λογισμικού. Η εφαρμογή λειτουργεί ως ενδιάμεσο επίπεδο μεταξύ του χρήστη και της οντολογίας: ένας μη εξειδικευμένος χρήστης μπορεί να εργαστεί μέσω συμβατικών φορμών και σελίδων, ενώ ένας πιο έμπειρος χρήστης μπορεί να αξιοποιήσει απευθείας SPARQL επερωτήσεις. Με αυτόν τον τρόπο επιτυγχάνεται ο βασικός στόχος της εργασίας, δηλαδή η μετατροπή των security patterns από απομονωμένες περιγραφές φυσικής γλώσσας σε μια οργανωμένη, επεκτάσιμη και μηχανικά επεξεργάσιμη βάση γνώσης.

3.1 Περίγραμμα

Η εργασία οργανώνεται γύρω από την ανάπτυξη και αξιολόγηση ενός σημασιολογικού πλαισίου για την περιγραφή και διαχείριση μοτίβων ασφάλειας. Η εφαρμογή αποτελεί το πρακτικό μέρος της λύσης και επιτρέπει την αξιοποίηση της γνώσης που αναπαρίσταται μέσω της οντολογίας, ώστε τα

μοτίβα ασφάλειας να μπορούν να περιγράφονται, να ταξινομούνται, να συσχετίζονται και να αναζητούνται με βάση τόσο τα περιγραφικά όσο και τα σημασιολογικά χαρακτηριστικά τους.

Η ανάπτυξη της εφαρμογής πραγματοποιήθηκε με βάση το μοντέλο του καταρράκτη. Στο μοντέλο αυτό, η ανάπτυξη πραγματοποιείται σειριακά με βάση ένα σύνολο από φάσεις. Κάθε φάση θα πρέπει να έχει ολοκληρωθεί πριν ξεκινήσει η επόμενη. Βάσει αυτού του μοντέλου, η υπόλοιπη δομή του κεφαλαίου είναι οργανωμένη γύρω από τις φάσεις ανάπτυξης και τα αποτελέσματά τους. Το κεφάλαιο ολοκληρώνεται με την αποτίμηση της προτεινόμενης λύσης.

Το πρώτο μέρος της τεχνικής ανάλυσης παρουσιάζει τις απαιτήσεις της εφαρμογής. Οι απαιτήσεις χωρίζονται σε λειτουργικές και μη λειτουργικές. Οι λειτουργικές απαιτήσεις περιγράφουν τις επιχειρησιακές δυνατότητες που παρέχονται στον χρήστη, όπως η προβολή και αναζήτηση μοτίβων, η εμφάνιση της σημασιολογικής τους περιγραφής, η δημιουργία, η επεξεργασία και η λογική διαγραφή τους, καθώς και η δυνατότητα εκτέλεσης επερωτήσεων SPARQL για την ανάκτηση γνώσης από την οντολογία. Οι μη λειτουργικές απαιτήσεις αφορούν χαρακτηριστικά ποιότητας του συστήματος, όπως η επεκτασιμότητα, η συντηρησιμότητα, η ευχρηστία και η συνέπεια των δεδομένων.

Στη συνέχεια παρουσιάζεται η εννοιολογική σχεδίαση. Στο σημείο αυτό περιγράφεται η θεωρητική δομή ενός μοτίβου ασφάλειας και τα στοιχεία που απαιτούνται για τη σημασιολογική αναπαράστασή του. Πέρα από τα βασικά περιγραφικά στοιχεία, όπως το όνομα, η περιγραφή, η πρόθεση, το πλαίσιο, το πρόβλημα, οι δυνάμεις, η λύση και οι συνέπειες, εξετάζονται οι σημασιολογικές σχέσεις του μοτίβου με έννοιες όπως οι κατηγορίες μοτίβων, οι απειλές, τα προστατευόμενα αγαθά, οι ιδιότητες ασφάλειας, οι φάσεις ανάπτυξης, τα επίπεδα κινδύνου, τα πλαίσια εφαρμογής και άλλα μοτίβα, καθώς και η σύνδεσή του με βιβλιογραφικές πηγές. Η δομή αυτή αντιστοιχεί στο εμπλουτισμένο μοντέλο που χρησιμοποιεί η τελική εφαρμογή.

Ακολουθεί η δημιουργία της οντολογίας. Παρουσιάζονται η ιεραρχία των κλάσεων, τα άτομα της οντολογίας, οι Datatype και Object Properties και οι σημασιολογικές σχέσεις μέσω των οποίων αναπαρίστανται τα μοτίβα και οι συναφείς έννοιες. Περιγράφεται επίσης η ανάπτυξη και επιθεώρηση της οντολογίας στο Protégé, καθώς και ο έλεγχος της λογικής συνέπειας και της inferred ταξινόμησής της με τη χρήση του HermiT reasoner. Η σημασιολογική βάση γνώσης της εφαρμογής καλύπτεται κυρίως από το περιεχόμενο του αρχείου OWL, το οποίο περιλαμβάνει τόσο το οντολογικό σχήμα όσο και τα instances και τις μεταξύ τους σχέσεις.

Στη συνέχεια της αρχιτεκτονικής παρουσιάζεται ο διαχωρισμός του συστήματος σε επίπεδα (layers). Το UI Layer είναι υπεύθυνο για την αλληλεπίδραση με τον χρήστη, το Business Logic Layer αναλαμβάνει την επεξεργασία και τη μετατροπή της σημασιολογικής πληροφορίας σε αντικείμενα που χρησιμοποιούνται από την εφαρμογή, υλοποιώντας την επιχειρηματική λογική, ενώ το Data Access Layer αποτελεί το επίπεδο πρόσβασης στη σημασιολογική βάση γνώσης. Ο διαχωρισμός αυτός περιορίζει την άμεση εξάρτηση της διεπαφής χρήστη από τον τρόπο αναπαράστασης και αποθήκευσης της οντολογικής γνώσης.

Στην ενότητα της υλοποίησης παρουσιάζονται οι βασικοί μηχανισμοί μέσω των οποίων η αρχιτεκτονική αυτή μεταφέρεται στον κώδικα της εφαρμογής. Περιγράφονται η φόρτωση της κεντρικής οντολογίας, η ανάκτηση κλάσεων, ατόμων και σημασιολογικών σχέσεων, η αντιστοίχισή τους στα μοντέλα της εφαρμογής και η λειτουργία της προβολής, δημιουργίας, επεξεργασίας και λογικής διαγραφής μοτίβων. Παρουσιάζεται επίσης η διεπαφή εκτέλεσης επερωτήσεων SPARQL σε λειτουργία μόνο ανάγνωσης και ο τρόπος με τον οποίο οι επιτρεπόμενες αλλαγές αποθηκεύονται πίσω στην οντολογία. Τα αποσπάσματα κώδικα χρησιμοποιούνται για την τεκμηρίωση συγκεκριμένων σχεδιαστικών και υλοποιητικών επιλογών και όχι ως ανεξάρτητα παραδείγματα. Η τελική υλοποίηση πράγματι διαχωρίζει semantic create/edit/delete από τις παλαιότερες απλές λειτουργίες και δημιουργεί Class, Data Property και Object Property assertions στο OWL.

Τέλος, παρουσιάζονται τα συμπεράσματα και η αποτίμηση της λύσης. Η αξιολόγηση εξετάζει τόσο το σημασιολογικό μοντέλο και τη συνέπεια της οντολογίας όσο και τη λειτουργία της εφαρμογής σε σχέση με τις απαιτήσεις που τέθηκαν. Ακολουθεί η σύγκριση με σχετικές εργασίες, ώστε να αποτιμηθούν τα χαρακτηριστικά της προτεινόμενης προσέγγισης, οι περιορισμοί της και οι δυνατότητες περαιτέρω επέκτασής της.

3.2 Ανάλυση Απαιτήσεων

Οι απαιτήσεις του συστήματος προκύπτουν από τον βασικό στόχο της εργασίας: την αξιοποίηση μιας σημασιολογικής αναπαράστασης των μοτίβων ασφάλειας μέσα από μια εφαρμογή ιστού, η οποία επιτρέπει τόσο την εύκολη πρόσβαση στη σχετική γνώση όσο και τη διατήρηση της σημασιολογικής δομής της. Κατά συνέπεια, οι απαιτήσεις δεν περιορίζονται στις λειτουργίες μιας συμβατικής εφαρμογής διαχείρισης δεδομένων, αλλά αφορούν επίσης τον τρόπο με τον οποίο τα μοτίβα ταξινομούνται, συσχετίζονται με άλλες έννοιες και αποθηκεύονται στη βάση γνώσης.

Βασική απαίτηση είναι ο χρήστης να μπορεί να αξιοποιεί τη βιβλιοθήκη των μοτίβων χωρίς να απαιτείται γνώση των τεχνολογιών OWL, RDF ή SPARQL. Οι συνηθέστερες ενέργειες πρέπει να πραγματοποιούνται μέσω της γραφικής διεπαφής της εφαρμογής, ενώ οι σημασιολογικές λεπτομέρειες της αναπαράστασης να παραμένουν διαφανείς για τον χρήστη. Παράλληλα, για περισσότερο εξειδικευμένη διερεύνηση της βάσης γνώσης πρέπει να παρέχεται ξεχωριστή δυνατότητα εκτέλεσης επερωτήσεων SPARQL μόνο για ανάγνωση.

Ιδιαίτερη σημασία έχει η διατήρηση της σημασιολογικής συνέπειας των δεδομένων. Οι κατηγορίες των μοτίβων, οι απειλές, τα προστατευόμενα αγαθά, οι ιδιότητες ασφάλειας, οι φάσεις ανάπτυξης, τα επίπεδα κινδύνου και οι υπόλοιπες έννοιες δεν πρέπει να αντιμετωπίζονται ως ανεξέλεγκτες τιμές κειμένου όταν αντιστοιχούν σε οντότητες της οντολογίας. Η εφαρμογή πρέπει επομένως, όπου είναι δυνατό, να επιτρέπει στον χρήστη να επιλέγει μεταξύ των υπαρχουσών σημασιολογικών οντοτήτων και να δημιουργεί τις αντίστοιχες σχέσεις μεταξύ τους. Η απαίτηση αυτή αντανακλάται στην τρέχουσα εφαρμογή, όπου τα διαθέσιμα semantic values των φορμών Create και Edit παρέχονται ως ontology references.

Επιπλέον, το σύστημα πρέπει να αποφεύγει την ακούσια απώλεια γνώσης. Για τον λόγο αυτό, η διαγραφή ενός μοτίβου δεν πρέπει να συνεπάγεται την οριστική αφαίρεση της οντότητάς του από τη βάση γνώσης, ενώ κατά την επεξεργασία πρέπει να διατηρείται ένα βασικό ίχνος της προηγούμενης κατάστασης. Η απαίτηση αυτή δεν ισοδυναμεί με πλήρες σύστημα versioning, αλλά παρέχει έναν στοιχειώδη μηχανισμό ιχνηλασιμότητας των μεταβολών.

Στην παρούσα υλοποίηση δεν έχει ενσωματωθεί μηχανισμός αυθεντικοποίησης ή διαφοροποίησης ρόλων χρηστών. Συνεπώς, θεωρείται ότι ο χρήστης που αποκτά πρόσβαση στην εφαρμογή μπορεί να χρησιμοποιήσει το σύνολο των διαθέσιμων λειτουργιών. Το στοιχείο αυτό αποτελεί περιορισμό της παρούσας έκδοσης και λαμβάνεται υπόψη κατά την αποτίμηση και τη μελλοντική επέκταση του συστήματος.

Οι απαιτήσεις διακρίνονται σε λειτουργικές απαιτήσεις, οι οποίες περιγράφουν τις ενέργειες και τις επιχειρησιακές δυνατότητες που πρέπει να παρέχει η εφαρμογή, και σε μη λειτουργικές απαιτήσεις, οι οποίες αφορούν χαρακτηριστικά ποιότητας και περιορισμούς της λύσης. Οι λεπτομέρειες σχετικά με τον τρόπο με τον οποίο οι απαιτήσεις αυτές υλοποιούνται στον κώδικα της εφαρμογής παρουσιάζονται στις επόμενες ενότητες της αρχιτεκτονικής και της υλοποίησης.

3.2.1 Λειτουργικές απαιτήσεις

Οι λειτουργικές απαιτήσεις της εφαρμογής προσδιορίζουν τις δυνατότητες που πρέπει να παρέχονται στον χρήστη κατά την αναζήτηση, τη μελέτη και τη διαχείριση των μοτίβων ασφάλειας.

1. ***Προβολή της βιβλιοθήκης μοτίβων ασφάλειας***: Ο χρήστης πρέπει να μπορεί να προβάλλει το σύνολο των ενεργών μοτίβων ασφάλειας που είναι διαθέσιμα στη βάση γνώσης. Για κάθε μοτίβο πρέπει να εμφανίζονται βασικές συνοπτικές πληροφορίες, ώστε να είναι δυνατή η αναγνώρισή του και η μετάβαση στην αναλυτική του παρουσίαση. Η λίστα πρέπει επίσης να υποστηρίζει τμηματική παρουσίαση των αποτελεσμάτων (σελιδοποίηση/paging), ώστε να παραμένει λειτουργική ακόμη και όταν αυξάνεται το πλήθος των μοτίβων. Για την κατάλληλη και πλήρη υποστήριξη του τελευταίου χαρακτηριστικού, η διεπαφή θα πρέπει να παρέχει επιλογή μεγέθους σελίδας και πλοήγηση μεταξύ των αποτελεσμάτων.
2. ***Αναζήτηση μοτίβων μέσω της βασικής διεπαφής:*** Ο χρήστης πρέπει να μπορεί να περιορίζει τη λίστα των εμφανιζόμενων μοτίβων με βάση όρους αναζήτησης. Η βασική αναζήτηση της γραφικής διεπαφής πρέπει να λειτουργεί άμεσα επάνω στις πληροφορίες που παρουσιάζονται στον πίνακα, κυρίως στο όνομα και στην περιγραφή του μοτίβου. Η λειτουργία αυτή αποσκοπεί στη γρήγορη εύρεση γνωστού ή σχετικού μοτίβου χωρίς να απαιτείται σύνταξη σημασιολογικού ερωτήματος.
3. ***Αναλυτική προβολή της σημασιολογικής περιγραφής ενός μοτίβου***: Ο χρήστης πρέπει να μπορεί να επιλέγει ένα μοτίβο και να προβάλλει την πλήρη πληροφορία που είναι διαθέσιμη για αυτό. Η παρουσίαση πρέπει να περιλαμβάνει τόσο τα περιγραφικά στοιχεία του μοτίβου, όπως όνομα, περιγραφή, πρόθεση, πλαίσιο, πρόβλημα, δυνάμεις, λύση, συνέπειες και

παράδειγμα υλοποίησης, όσο και τις σημασιολογικές του σχέσεις. Σε αυτές περιλαμβάνονται οι απειλές, οι κατηγορίες απειλών, τα προστατευόμενα αγαθά, οι ιδιότητες ασφάλειας, οι φάσεις ανάπτυξης, το επίπεδο κινδύνου, τα πλαίσια εφαρμογής, τα επίπεδα και οι καταστάσεις επικύρωσης του μοτίβου, καθώς και οι σχέσεις του με άλλα μοτίβα. Πρέπει ακόμη να είναι δυνατή η προβολή των βιβλιογραφικών πηγών και της διαδρομής ταξινόμησης του μοτίβου στην ιεραρχία της οντολογίας. Η τελική σελίδα Details υποστηρίζει ακριβώς αυτή την εμπλουτισμένη παρουσίαση.

4. ***Δημιουργία νέου μοτίβου ασφάλειας***: Ο χρήστης πρέπει να μπορεί να δημιουργεί νέο μοτίβο μέσω κατάλληλης φόρμας. Η δημιουργία πρέπει να υποστηρίζει τα περιγραφικά στοιχεία του μοτίβου, τις βιβλιογραφικές πληροφορίες και τις σημασιολογικές του σχέσεις με άλλες οντότητες της βάσης γνώσης. Για τις έννοιες που ήδη υπάρχουν στην οντολογία, όπως κατηγορίες, απειλές, επίπεδα κινδύνου, προστατευόμενα αγαθά, ιδιότητες ασφάλειας, φάσεις ανάπτυξης, πλαίσια εφαρμογής, επίπεδα μοτίβου, καταστάσεις επικύρωσης, άλλα μοτίβα και βιβλιογραφικές πηγές, ο χρήστης πρέπει να επιλέγει από τις διαθέσιμες οντολογικές τιμές. Το όνομα του μοτίβου είναι υποχρεωτικό, θα πρέπει να είναι μοναδικό ενώ κάθε νέο μοτίβο πρέπει να διαθέτει τουλάχιστον μία οντολογική ταξινόμηση.
5. ***Επεξεργασία υπάρχοντος μοτίβου***: Ο χρήστης πρέπει να μπορεί να τροποποιεί τόσο τα περιγραφικά στοιχεία όσο και τις σημασιολογικές σχέσεις ενός υπάρχοντος μοτίβου. Η φόρμα επεξεργασίας πρέπει να εμφανίζει τις ήδη αποθηκευμένες τιμές, καθώς και τις διαθέσιμες οντολογικές επιλογές, ώστε ο χρήστης να μπορεί να προσθέτει ή να αφαιρεί συσχετίσεις χωρίς να επεξεργάζεται απευθείας το OWL αρχείο. Μετά την επιτυχή επεξεργασία, οι νέες τιμές πρέπει να αποθηκεύονται στη βάση γνώσης, ενώ το σύστημα πρέπει να αποτρέπει τη μετονομασία ενός μοτίβου σε όνομα που χρησιμοποιείται ήδη από άλλο μοτίβο. Η επεξεργασία αφορά την υπάρχουσα οντότητα και όχι τη δημιουργία νέας ανεξάρτητης εγγραφής.
6. ***Λογική διαγραφή μοτίβου ασφάλειας***: Ο χρήστης πρέπει να μπορεί να ζητά τη διαγραφή ενός μοτίβου και να επιβεβαιώνει ρητά την ενέργεια πριν από την εκτέλεσή της. Η διαγραφή πρέπει να είναι λογική και όχι φυσική: το μοτίβο πρέπει να επισημαίνεται ως διαγραμμένο και να αποκρύπτεται από την κανονική λίστα και τις συνήθεις αναζητήσεις, χωρίς να αφαιρείται οριστικά η οντότητά του από την οντολογία. Με τον τρόπο αυτό διατηρείται η ήδη καταγεγραμμένη γνώση και ένα βασικό ίχνος της προηγούμενης κατάστασης του μοτίβου. Η επιβεβαίωση διαγραφής παρέχεται και από τη σελίδα Details της τελικής εφαρμογής.
7. ***Εκτέλεση σημασιολογικών επερωτήσεων SPARQL***: Πέρα από τη βασική αναζήτηση της γραφικής διεπαφής, ο χρήστης πρέπει να μπορεί να υποβάλει δικές του επερωτήσεις SPARQL SELECT επάνω στη βάση γνώσης και να βλέπει τα αποτελέσματα σε δομημένη μορφή. Η δυνατότητα αυτή επιτρέπει πιο σύνθετες ερωτήσεις που αξιοποιούν τις κλάσεις,

τις ιδιότητες και τις σχέσεις της οντολογίας και επομένως δεν περιορίζονται σε αντιστοίχιση λέξεων-κλειδιών. Η διεπαφή SPARQL πρέπει να είναι αποκλειστικά αναγνωστική και να μην επιτρέπει τροποποίηση της οντολογίας.

8. ***Διατήρηση βιβλιογραφικής προέλευσης και σημασιολογικών συσχετίσεων:*** Το σύστημα πρέπει να επιτρέπει σε ένα μοτίβο να συσχετίζεται με μία ή περισσότερες βιβλιογραφικές πηγές και να διατηρεί τα σχετικά στοιχεία τεκμηρίωσης. Παράλληλα, πρέπει να υποστηρίζονται πολλαπλές σημασιολογικές σχέσεις του ίδιου τύπου, όταν αυτό απαιτείται από τη γνώση που περιγράφεται. Για παράδειγμα, ένα μοτίβο μπορεί να σχετίζεται με περισσότερες από μία απειλές, ιδιότητες ασφάλειας, φάσεις ανάπτυξης ή άλλα μοτίβα. Η απαίτηση αυτή είναι απαραίτητη ώστε η αναπαράσταση να μη συμπιέζει σύνθετη σημασιολογική πληροφορία σε ένα μοναδικό πεδίο κειμένου.

3.2.2 Μη Λειτουργικές απαιτήσεις

Οι μη λειτουργικές απαιτήσεις αφορούν χαρακτηριστικά ποιότητας και περιορισμούς που πρέπει να ικανοποιεί το σύστημα ανεξάρτητα από τις επιμέρους λειτουργίες του.

1. ***Ευχρηστία***: Η εφαρμογή πρέπει να επιτρέπει την εκτέλεση των βασικών εργασιών χωρίς να απαιτεί από τον χρήστη γνώση OWL, RDF ή SPARQL. Οι λειτουργίες προβολής, αναζήτησης, δημιουργίας, επεξεργασίας και διαγραφής πρέπει να είναι διαθέσιμες μέσω συμβατικών στοιχείων μιας εφαρμογής ιστού, όπως πίνακες, φόρμες, πεδία επιλογής και σαφείς ενέργειες πλοήγησης. Η απευθείας χρήση SPARQL αποτελεί πρόσθετη δυνατότητα για εξειδικευμένους χρήστες και όχι προϋπόθεση για τη συνήθη χρήση της εφαρμογής.
2. ***Σημασιολογική συνέπεια και ακεραιότητα των δεδομένων***: Η εφαρμογή πρέπει να διατηρεί τη διάκριση μεταξύ απλών περιγραφικών τιμών και πραγματικών οντολογικών σχέσεων. Οι σημασιολογικές συσχετίσεις πρέπει να χρησιμοποιούν έγκυρα IRI υπαρχουσών οντοτήτων, ενώ κάθε νέο ή τροποποιημένο μοτίβο πρέπει να διαθέτει τουλάχιστον μία έγκυρη ταξινόμηση ως security pattern. Πρέπει επίσης να αποφεύγεται η δημιουργία διπλότυπων μοτίβων βάσει ονόματος. Οι περιορισμοί αυτοί εφαρμόζονται ήδη στο semantic Create/Edit workflow.
3. ***Συντηρησιμότητα***: Η δομή της εφαρμογής πρέπει να διαχωρίζει τη διεπαφή χρήστη, την επιχειρησιακή λογική και την πρόσβαση στα δεδομένα, ώστε μια αλλαγή στον τρόπο παρουσίασης των πληροφοριών να μην απαιτεί αντίστοιχη αλλαγή στον μηχανισμό αποθήκευσής τους και αντίστροφα. Ο διαχωρισμός αυτός πρέπει επίσης να περιορίζει την άμεση εξάρτηση των controllers και των views από τις λεπτομέρειες επεξεργασίας του OWL/RDF γράφου.
4. ***Επεκτασιμότητα***: Το σύστημα πρέπει να μπορεί να προσαρμοστεί σε επέκταση της οντολογίας και της βιβλιοθήκης μοτίβων χωρίς να απαιτείται σημαντικός ανασχεδιασμός της βασικής λειτουργίας του. Η ανάκτηση της ιεραρχίας κατηγοριών, των διαθέσιμων

individuals και των σημασιολογικών συσχετίσεων πρέπει να βασίζεται όσο είναι δυνατό στη δομή της ίδιας της οντολογίας. Η τελική υλοποίηση, για παράδειγμα, ανακατασκευάζει τις διαδρομές ταξινόμησης από τις σχέσεις rdfs:subClassOf και όχι από έναν στατικό κατάλογο πιθανών διαδρομών.

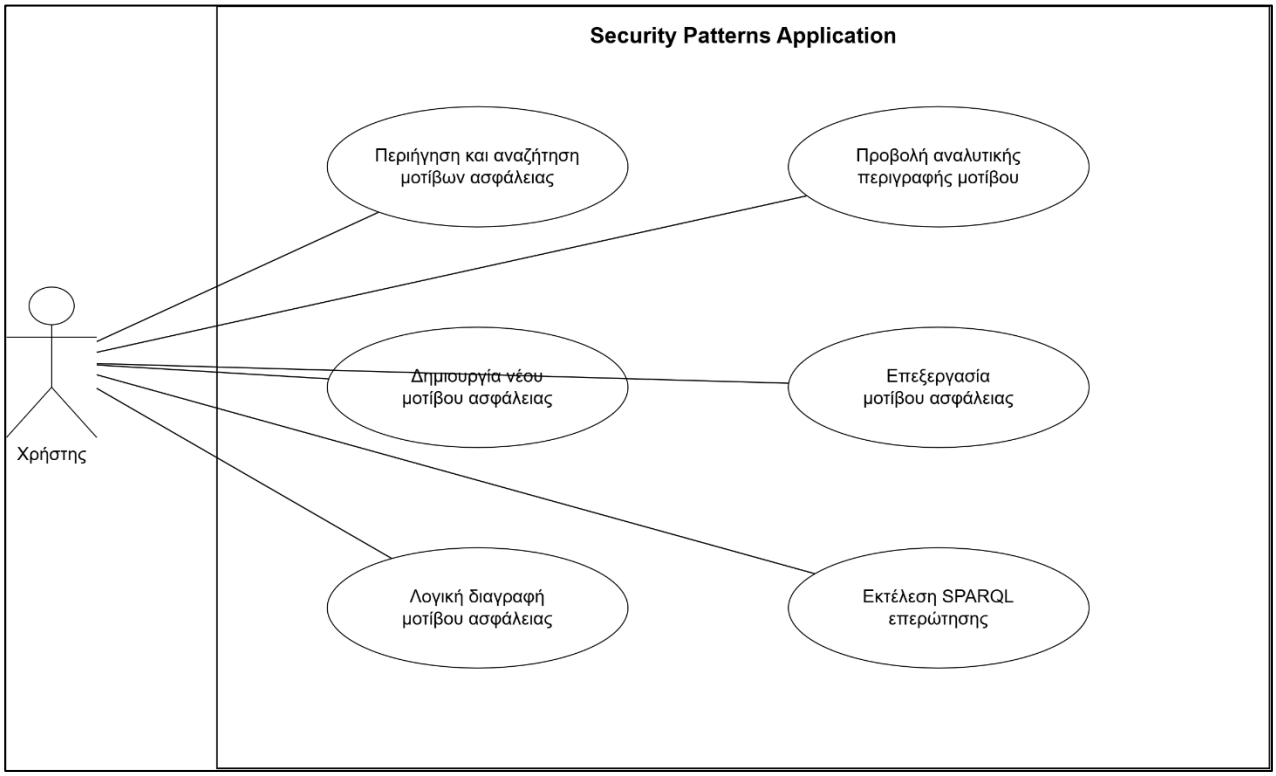
5. ***Απόδοση και απόκριση***: Η ανάκτηση των μοτίβων και των συχνά χρησιμοποιούμενων οντολογικών τιμών πρέπει να πραγματοποιείται με τρόπο που να μην απαιτεί επαναφόρτωση ολόκληρης της οντολογίας για κάθε ενέργεια του χρήστη. Η εφαρμογή πρέπει να μπορεί να εξυπηρετεί διαδοχικές αναζητήσεις και προβολές με αποδεκτό χρόνο απόκρισης ακόμη και όταν η βάση γνώσης περιέχει μεγάλο αριθμό RDF δηλώσεων. Στην τρέχουσα υλοποίηση το OWL γράφημα φορτώνεται σε in-memory dataset, χρησιμοποιούνται caches για συχνές αναζητήσεις και πραγματοποιείται αρχική προφόρτωση βασικών δεδομένων κατά την εκκίνηση.
6. ***Ασφαλής εκτέλεση επερωτήσεων***: Η δυνατότητα υποβολής SPARQL από τον χρήστη δεν πρέπει να επιτρέπει τη μεταβολή της βάσης γνώσης ή την πρόσβαση σε αυθαίρετες εξωτερικές πηγές. Επιτρέπονται μόνο επερωτήσεις της οικογένειας SELECT, ενώ πρέπει να απορρίπτονται federated SERVICE clauses και δηλώσεις FROM ή FROM NAMED. Επιπλέον, πρέπει να εφαρμόζεται ανώτατο όριο αποτελεσμάτων, ώστε ένα ερώτημα να μην μπορεί να επιστρέψει ανεξέλεγκτα μεγάλο αποτέλεσμα. Η τελική εφαρμογή εφαρμόζει όριο 500 αποτελεσμάτων για user-defined SPARQL queries.
7. ***Διατήρηση και ιχνηλασιμότητα των μεταβολών***: Οι επιτυχείς μεταβολές που πραγματοποιούνται μέσω της εφαρμογής πρέπει να αποθηκεύονται μόνιμα στη βάση γνώσης και να παραμένουν διαθέσιμες σε επόμενες εκτελέσεις της εφαρμογής. Παράλληλα, κατά την επεξεργασία και τη λογική διαγραφή πρέπει να διατηρείται βασική πληροφορία για την προηγούμενη κατάσταση της οντότητας. Η απαίτηση αυτή παρέχει στοιχειώδη ιχνηλασιμότητα, χωρίς να υποκαθιστά ένα πλήρες σύστημα ιστορικότητας ή διαχείρισης εκδόσεων.

3.3 Μοντελοποίηση Περιπτώσεων Χρήσης

Με βάση τις λειτουργικές απαιτήσεις της εφαρμογής, οι βασικές αλληλεπιδράσεις του χρήστη με το σύστημα μοντελοποιούνται ως περιπτώσεις χρήσης. Ο χρήστης αποτελεί τον βασικό actor του συστήματος, καθώς στην παρούσα έκδοση της εφαρμογής δεν υλοποιούνται διαφορετικοί ρόλοι ή μηχανισμός αυθεντικοποίησης. Οι βασικές περιπτώσεις χρήσης περιλαμβάνουν την περιήγηση και αναζήτηση στη βιβλιοθήκη μοτίβων ασφάλειας, την αναλυτική προβολή ενός μοτίβου, τη δημιουργία νέου μοτίβου, την επεξεργασία υπάρχοντος μοτίβου, τη λογική διαγραφή του και την εκτέλεση σημασιολογικών επερωτήσεων SPARQL.

Οι παραπάνω περιπτώσεις χρήσης παρουσιάζονται στην Εικόνα 8. Η διατήρηση της βιβλιογραφικής προέλευσης και των σημασιολογικών συσχετίσεων δεν μοντελοποιείται ως

ανεξάρτητη περίπτωση χρήσης, καθώς αποτελεί μέρος των λειτουργιών προβολής, δημιουργίας και επεξεργασίας των μοτίβων ασφάλειας.



Εικόνα 8: Διάγραμμα περιπτώσεων χρήσης των βασικών λειτουργιών της εφαρμογής.

3.4 Εννοιολογική Σχεδίαση

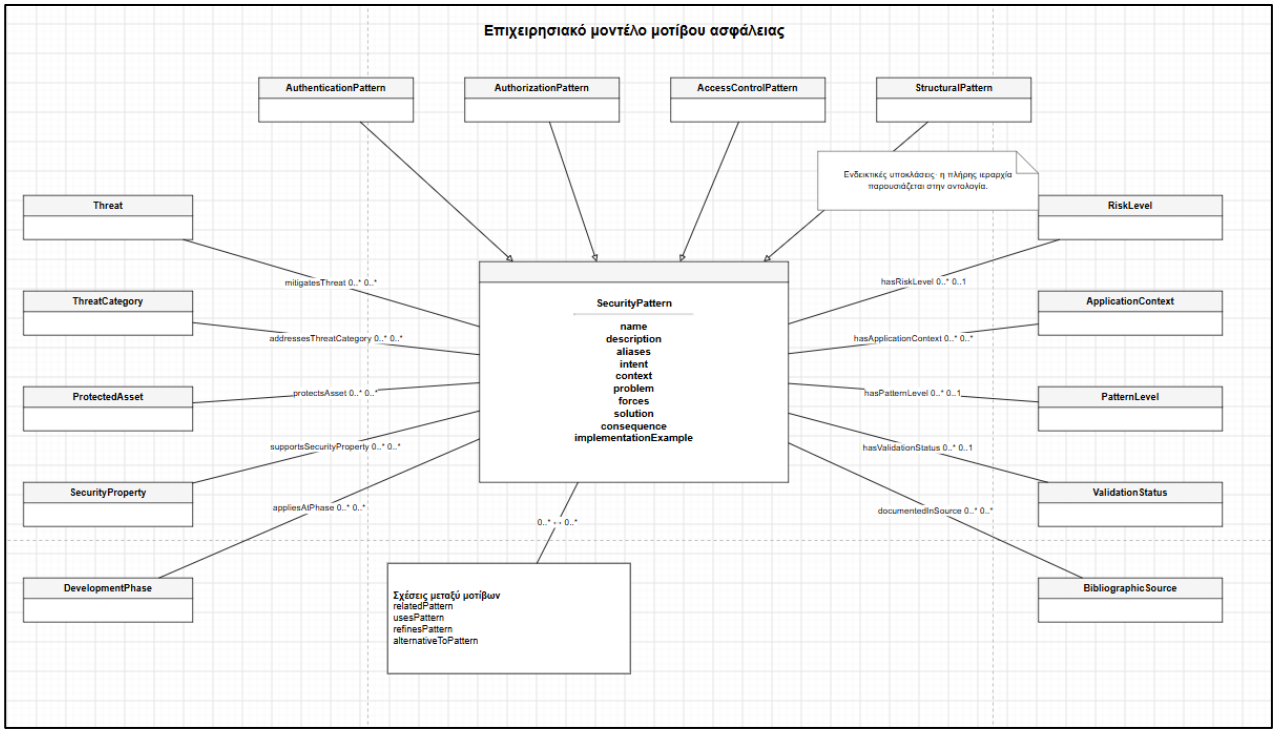
Η εννοιολογική σχεδίαση αποτελεί το στάδιο κατά το οποίο οι απαιτήσεις της εφαρμογής μεταφράστηκαν σε ένα αφηρημένο μοντέλο των βασικών εννοιών του πεδίου των μοτίβων ασφάλειας και των σχέσεων που αναπτύσσονται μεταξύ τους. Στόχος του σταδίου αυτού δεν ήταν ο προσδιορισμός της τεχνικής μορφής της OWL οντολογίας ή της δομής του πηγαίου κώδικα, αλλά ο καθορισμός του τι πρέπει να αναπαρίσταται ώστε ένα μοτίβο ασφάλειας να αποτελεί ολοκληρωμένη και αξιοποιήσιμη μονάδα γνώσης.

Η κεντρική έννοια του μοντέλου που σχεδιάστηκε είναι το Security Pattern (δηλαδή το μοτίβο ασφαλείας) που αναπαριστά ένα μοτίβο ασφάλειας. Σύμφωνα με τη βιβλιογραφία, τα μοτίβα ασφάλειας περιγράφουν επαναχρησιμοποιήσιμες λύσεις για επαναλαμβανόμενα προβλήματα ασφάλειας και συνήθως οργανώνονται γύρω από στοιχεία όπως το πλαίσιο, το πρόβλημα και η λύση (Jafari & Rasoolzadegan, 2020). Στην παρούσα εργασία η βασική αυτή δομή διατηρείται, αλλά εμπλουτίζεται ώστε να υποστηρίζει τόσο την αναλυτική περιγραφή ενός μοτίβου όσο και τη σημασιολογική του συσχέτιση με άλλες έννοιες του πεδίου.

Κάθε μοτίβο διαθέτει αρχικά ένα σύνολο περιγραφικών χαρακτηριστικών. Σε αυτά περιλαμβάνονται το όνομα (Name), η περιγραφή (Description), πιθανές εναλλακτικές ονομασίες (Aliases), η πρόθεση (Intent), το πλαίσιο εφαρμογής (Context), το πρόβλημα (Problem), οι δυνάμεις

ή περιορισμοί που επηρεάζουν την επιλογή της λύσης (Forces), η προτεινόμενη λύση (Solution), οι συνέπειες της εφαρμογής της (Consequence) και ένα ενδεικτικό παράδειγμα υλοποίησης (ImplementationExample). Τα στοιχεία αυτά συγκροτούν την ανθρώπινα αναγνώσιμη περιγραφή του μοτίβου και επιτρέπουν στον χρήστη να αντιληφθεί όχι μόνο ποια λύση προτείνεται αλλά και υπό ποιες συνθήκες είναι κατάλληλη.

Η τελική εφαρμογή αποτυπώνει τα στοιχεία αυτά απευθείας στο μοντέλο SecurityPattern, μαζί με πρόσθετα στοιχεία τεκμηρίωσης και τις σημασιολογικές σχέσεις του μοτίβου.



Εικόνα 9: UML διάγραμμα κλάσεων του επιχειρησιακού μοντέλου ενός μοτίβου ασφάλειας και των βασικών σημασιολογικών του συσχετίσεων.

Το UML διάγραμμα κλάσεων της Εικόνας 9 παρουσιάζει το επιχειρησιακό μοντέλο ενός μοτίβου ασφάλειας και τις βασικές έννοιες με τις οποίες αυτό συσχετίζεται. Η SecurityPattern αποτελεί την κεντρική κλάση του μοντέλου και περιλαμβάνει τα βασικά περιγραφικά χαρακτηριστικά ενός μοτίβου, ενώ οι συσχετίσεις της με τις υπόλοιπες κλάσεις αποτυπώνουν τις σημασιολογικές σχέσεις που υποστηρίζονται από την προτεινόμενη λύση.

Ιδιαίτερη σημασία έχει η ταξινόμηση των μοτίβων ασφάλειας. Η κατηγορία ενός μοτίβου δεν θεωρείται απλό χαρακτηριστικό κειμένου, αλλά τμήμα μιας ιεραρχίας εννοιών. Η γενική έννοια SecurityPattern εξειδικεύεται σε κατηγορίες όπως AuthenticationPattern, AuthorizationPattern, AccessControlPattern, InputValidationPattern, CryptographyPattern, NetworkSecurityPattern, SecureArchitecturePattern, StructuralPattern, ProceduralPattern και άλλες. Η ιεραρχική αυτή δομή επιτρέπει σε ένα συγκεκριμένο μοτίβο να ανήκει σε μία ή περισσότερες εξειδικευμένες κατηγορίες και ταυτόχρονα να αναγνωρίζεται ως γενικό SecurityPattern μέσω της σχέσης γενίκευσης των

κλάσεων. Το συνολικό διάγραμμα που έχει δημιουργηθεί για την εργασία περιλαμβάνει ήδη τις συγκεκριμένες κατηγορίες και τη γενική κλάση SecurityPattern.

Η πολλαπλή ταξινόμηση είναι σημαντική, επειδή ένα μοτίβο μπορεί να εξυπηρετεί περισσότερους από έναν σκοπούς. Για παράδειγμα, ένα μοτίβο μπορεί ταυτόχρονα να σχετίζεται με την αυθεντικοποίηση και με τη διαχείριση ταυτότητας ή να αποτελεί τόσο δομικό όσο και περισσότερο εξειδικευμένο μοτίβο ασφάλειας. Συνεπώς, το μοντέλο δεν επιβάλλει σε κάθε μοτίβο μία μοναδική κατηγορία, αλλά επιτρέπει την έκφραση των πραγματικών σχέσεων ταξινόμησης του πεδίου. Η τελική εφαρμογή ανακτά τις κατηγορίες ως συλλογή Categories και κατασκευάζει τις πλήρεις διαδρομές ταξινόμησης στην ιδιότητα ClassificationPaths.

Πέρα από την ταξινόμηση, βασικός στόχος του μοντέλου είναι να αναπαριστά τι αντιμετωπίζει ένα μοτίβο και τι προστατεύει. Για τον σκοπό αυτό διακρίνονται οι έννοιες Threat, ThreatCategory, ProtectedAsset και SecurityProperty. Ένα security pattern μπορεί να σχετίζεται με μία ή περισσότερες συγκεκριμένες απειλές τις οποίες μετριάζει, με ευρύτερες κατηγορίες απειλών, με τα αγαθά που προστατεύει και με τις ιδιότητες ασφάλειας που υποστηρίζει. Με αυτόν τον τρόπο μπορεί, για παράδειγμα, το ίδιο προστατευόμενο αγαθό ή η ίδια ιδιότητα ασφάλειας να συσχετιστεί με διαφορετικά μοτίβα, επιτρέποντας την αναζήτηση κοινών χαρακτηριστικών μεταξύ διαφορετικών λύσεων.

Πίνακας 1: Βασικές σημασιολογικές σχέσεις του εννοιολογικού μοντέλου των μοτίβων ασφάλειας

Σχέση	***Από***	***Προς***	***Σημασία***
mitigatesThreat	SecurityPattern	Threat	Απειλή που μετριάζει το μοτίβο
addressesThreatCategory	SecurityPattern	ThreatCategory	Ευρύτερη κατηγορία απειλής
protectsAsset	SecurityPattern	ProtectedAsset	Αγαθό που προστατεύεται
supportsSecurityProperty	SecurityPattern	SecurityProperty	Ιδιότητα ασφάλειας που υποστηρίζεται
appliesAtPhase	SecurityPattern	DevelopmentPhase	Φάση ανάπτυξης στην οποία εφαρμόζεται
hasRiskLevel	SecurityPattern	RiskLevel	Σχετικό επίπεδο κινδύνου
hasApplicationContext	SecurityPattern	ApplicationContext	Πλαίσιο στο οποίο εφαρμόζεται
hasPatternLevel	SecurityPattern	PatternLevel	Επίπεδο αφαίρεσης του μοτίβου
hasValidationStatus	SecurityPattern	ValidationStatus	Κατάσταση αξιολόγησης ή επικύρωσης

Οι συγκεκριμένες σχέσεις αντιστοιχούν στις σημασιολογικές ιδιότητες που χρησιμοποιούνται πράγματι στην τελική εφαρμογή και δεν αποτελούν πλέον απλά πεδία κειμένου.

Η διάκριση αυτή μεταξύ περιγραφικής πληροφορίας και σχέσεων μεταξύ εννοιών αποτελεί μία από τις σημαντικότερες σχεδιαστικές αποφάσεις του μοντέλου. Για παράδειγμα, η πρόταση λύσης ενός pattern είναι κείμενο που αφορά το ίδιο το μοτίβο. Αντίθετα, η απειλή που μετριάζει δεν είναι απλώς ένα λεκτικό χαρακτηριστικό· αποτελεί ανεξάρτητη έννοια του πεδίου, η οποία μπορεί να σχετίζεται με πολλά διαφορετικά patterns. Το ίδιο ισχύει για τα προστατευόμενα αγαθά, τις ιδιότητες ασφάλειας και τις φάσεις ανάπτυξης. Στο επίπεδο της OWL, η διάκριση αυτή θα μεταφερθεί αντίστοιχα σε Datatype Properties και Object Properties, όπως περιγράφεται αναλυτικά στην επόμενη ενότητα.

Το μοντέλο περιλαμβάνει επίσης σχέσεις μεταξύ των ίδιων των μοτίβων ασφάλειας. Μια γενική συσχέτιση μπορεί να εκφράζεται μέσω της relatedPattern, ενώ πιο συγκεκριμένες σχέσεις περιγράφονται μέσω των usesPattern, refinesPattern και alternativeToPattern. Με αυτές είναι δυνατό να δηλωθεί ότι ένα μοτίβο αξιοποιεί κάποιο άλλο ως μέρος της λύσης του, ότι αποτελεί εξειδίκευση ενός γενικότερου μοτίβου ή ότι μπορεί να χρησιμοποιηθεί ως εναλλακτική επιλογή. Οι σχέσεις αυτές είναι σημαντικές για την αναπαράσταση ενός δικτύου αλληλοσυνδεόμενων λύσεων και όχι απλώς μιας επίπεδης λίστας μοτίβων. Στο αντικειμενοστραφές μοντέλο της εφαρμογής αποτυπώνονται αντίστοιχα ως RelatedPatterns, UsedPatterns, RefinedPatterns και AlternativePatterns.

Ένας ακόμη διακριτός άξονας του εννοιολογικού σχεδιασμού είναι η βιβλιογραφική προέλευση της γνώσης. Για τον σκοπό αυτό η BibliographicSource μοντελοποιείται ως ανεξάρτητη έννοια και όχι ως ένα απλό πεδίο citation μέσα στο pattern. Με αυτόν τον τρόπο η ίδια πηγή μπορεί να τεκμηριώνει περισσότερα από ένα μοτίβα, ενώ για κάθε πηγή είναι δυνατό να διατηρούνται στοιχεία όπως τίτλος, συγγραφείς, έτος έκδοσης, τύπος πηγής, DOI και URL. Επιπροσθέτως, χρησιμοποιείται η σχέση documentedInSource ώστε να συνδέσει ένα μοτίβο με την αντίστοιχη βιβλιογραφική πηγή. Παράλληλα, το μοτίβο μπορεί να διατηρεί ειδικότερες πληροφορίες τεκμηρίωσης, όπως SourceCitation, OriginalSourceCitation, CatalogCitation, SourcePage, SourceUrl και ValidationNote. Η τελική δομή του μοντέλου υποστηρίζει τόσο τις δομημένες βιβλιογραφικές οντότητες όσο και τα συγκεκριμένα στοιχεία προέλευσης κάθε pattern.

Η σχεδίαση προβλέπει επίσης ότι αρκετές από τις παραπάνω σχέσεις είναι πολλαπλής πληθικότητας. Ένα pattern μπορεί να μετριάζει περισσότερες από μία απειλές, να προστατεύει περισσότερα από ένα αγαθά, να υποστηρίζει περισσότερες από μία ιδιότητες ασφάλειας, να εφαρμόζεται σε περισσότερες φάσεις ανάπτυξης και να συνδέεται με περισσότερα patterns ή βιβλιογραφικές πηγές. Η επιλογή αυτή αποτυπώνεται και στο μοντέλο εισόδου της εφαρμογής, όπου οι αντίστοιχες σημασιολογικές τιμές αναπαρίστανται ως λίστες IRI και όχι ως μοναδικά strings.

Αντίθετα, ορισμένες πληροφορίες έχουν εκ φύσεως διαφορετική σημασιολογία. Το RiskLevel, για παράδειγμα, αντιμετωπίζεται στην τρέχουσα εφαρμογή ως μία επιλεγμένη οντολογική αναφορά για το pattern, ενώ οι απειλές, τα assets και οι φάσεις ανάπτυξης μπορούν να είναι πολλαπλές. Η διάκριση αυτή μεταφέρεται τόσο στο εννοιολογικό μοντέλο όσο και στις φόρμες δημιουργίας και επεξεργασίας της εφαρμογής.

Τέλος, είναι απαραίτητο να διαχωρίζεται η σημασιολογική γνώση του πεδίου από την πληροφορία που αφορά αποκλειστικά τη λειτουργία της εφαρμογής. Χαρακτηριστικό παράδειγμα αποτελεί η ένδειξη isDeleted. Η ιδιότητα αυτή δεν περιγράφει κάποιο εγγενές χαρακτηριστικό ενός security pattern, αλλά χρησιμοποιείται για την εφαρμογή της λογικής διαγραφής. Επομένως, θεωρείται πληροφορία διαχείρισης του κύκλου ζωής της εγγραφής και όχι μέρος της θεωρητικής περιγραφής του pattern.

Το τελικό εννοιολογικό μοντέλο μπορεί συνεπώς να θεωρηθεί ως ένα δίκτυο γνώσης με κεντρικό κόμβο το SecurityPattern. Γύρω από αυτό οργανώνονται η περιγραφή του προβλήματος και της λύσης, η ταξινόμηση του μοτίβου, οι απειλές που αντιμετωπίζει, τα αγαθά και οι ιδιότητες ασφάλειας που προστατεύει, το πλαίσιο και η φάση εφαρμογής του, οι σχέσεις του με άλλα patterns και οι βιβλιογραφικές πηγές που το τεκμηριώνουν. Η σχεδίαση αυτή αποτέλεσε τη βάση για την κατασκευή της OWL οντολογίας που παρουσιάζεται στην επόμενη ενότητα.

3.5 Δημιουργία Οντολογίας

Με βάση το εννοιολογικό μοντέλο της προηγούμενης ενότητας, αναπτύχθηκε η οντολογία που χρησιμοποιείται για τη σημασιολογική αναπαράσταση των μοτίβων ασφάλειας. Η οντολογία υλοποιήθηκε σε OWL 2 και αναπτύχθηκε και επιθεωρήθηκε μέσω του Protégé, το οποίο παρέχει εργαλεία για τη διαχείριση κλάσεων, ιδιοτήτων, ατόμων και αξιωμάτων μιας οντολογίας (Musen, 2015). Η επιλογή της OWL επιτρέπει τη σαφή διάκριση μεταξύ κλάσεων, individuals, Datatype Properties και Object Properties, καθώς και την έκφραση ιεραρχικών και σημασιολογικών σχέσεων με τυπικά καθορισμένο τρόπο (W3C OWL Working Group, 2012).

Το τελικό οντολογικό μοντέλο αποθηκεύεται στο αρχείο security_patterns.owl. Το περιεχόμενο του αρχείου αυτού καλύπτει το κύριο αποθηκευμένο τμήμα της σημασιολογικής βάσης γνώσης που αξιοποιεί η εφαρμογή και περιλαμβάνει τόσο το σχήμα της οντολογίας όσο και τα individuals που αντιστοιχούν σε συγκεκριμένα μοτίβα ασφάλειας και στις υπόλοιπες έννοιες του πεδίου. Η εφαρμογή χρησιμοποιεί την ίδια οντολογία για την ανάκτηση και τη διαχείριση της σημασιολογικής πληροφορίας, όπως αναλύεται στις επόμενες ενότητες της αρχιτεκτονικής και της υλοποίησης.

3.5.1 Ιεραρχία κλάσεων

Η κατασκευή της οντολογίας ξεκίνησε με τον ορισμό της ιεραρχίας των κλάσεων. Κεντρική κλάση είναι η SecurityPattern, κάτω από την οποία οργανώνονται εξειδικευμένες κατηγορίες μοτίβων

ασφάλειας. Ενδεικτικά, η ιεραρχία περιλαμβάνει κλάσεις όπως AuthenticationPattern, AuthorizationPattern, AccessControlPattern, InputValidationPattern, CryptographyPattern, NetworkSecurityPattern, SecureArchitecturePattern, SecureDevelopmentPattern, LoggingAndMonitoringPattern, APISecurityPattern, StructuralPattern και ProceduralPattern. Το πραγματικό διάγραμμα της οντολογίας περιλαμβάνει τις συγκεκριμένες κλάσεις και τις μεταξύ τους ιεραρχικές σχέσεις.

Η σχέση μεταξύ των κατηγοριών εκφράζεται μέσω της rdfs:subClassOf. Κατά συνέπεια, ένα individual το οποίο έχει ταξινομηθεί ως μέλος μιας εξειδικευμένης υποκλάσης αναγνωρίζεται ταυτόχρονα ως security pattern μέσω της ιεραρχίας. Η προσέγγιση αυτή επιτρέπει επίσης ένα συγκεκριμένο pattern να ανήκει σε περισσότερες από μία κλάσεις όταν αυτό ανταποκρίνεται καλύτερα στη σημασιολογία του.

Εικόνα 10: Ιεραρχία των βασικών κλάσεων και των κατηγοριών μοτίβων ασφάλειας στην οντολογία, όπως απεικονίζεται στο Protégé.

Η εικόνα αποτυπώνει την asserted ιεραρχία που ορίστηκε κατά την κατασκευή της οντολογίας. Η ύπαρξη της SecurityPattern ως κοινής γενικής κλάσης παρέχει ένα ενιαίο σημείο αναφοράς, χωρίς να επιβάλλει όλα τα μοτίβα να διαθέτουν την ίδια εξειδίκευση.

Πέρα από την SecurityPattern, δημιουργήθηκαν ανεξάρτητες κλάσεις για έννοιες που έχουν αυτοτελή σημασιολογική υπόσταση. Σε αυτές περιλαμβάνονται οι Threat, ThreatCategory, ProtectedAsset, SecurityProperty, DevelopmentPhase, RiskLevel, ApplicationContext, PatternLevel, ValidationStatus και BibliographicSource. Οι κλάσεις αυτές δεν αποτελούν υποκατηγορίες της SecurityPattern, αλλά διαφορετικούς τύπους οντοτήτων οι οποίοι μπορούν να συνδέονται με τα μοτίβα μέσω Object Properties. Η ίδια διάκριση χρησιμοποιείται και από το τελικό μοντέλο της εφαρμογής.

Εικόνα 11: Η κλάση SecurityPattern και η θέση της στην οντολογική δομή όπως παρουσιάζεται στο Protégé.

Η διάκριση μεταξύ της ιεραρχίας των κατηγοριών και των ανεξάρτητων σημασιολογικών κλάσεων είναι ουσιώδης. Η πρώτη χρησιμοποιείται για να εκφράσει τι είδους pattern είναι μια οντότητα, ενώ οι δεύτερες επιτρέπουν να εκφραστεί με ποιες έννοιες σχετίζεται.

3.5.2 Datatype Properties και Object Properties

Κατά την υλοποίηση του μοντέλου έγινε σαφής διάκριση μεταξύ Datatype Properties και Object Properties. Οι Datatype Properties χρησιμοποιούνται όταν η τιμή αποτελεί λεκτικό ή άλλο απλό δεδομένο που περιγράφει το ίδιο το security pattern, δηλαδή αποτελούν βασικές περιγραφικές ιδιότητες. Αντίθετα, οι Object Properties χρησιμοποιούνται όταν η τιμή διαθέτει αυτοτελή οντολογική ταυτότητα και επομένως αναπαρίσταται ως διαφορετικό individual.

Οι βασικές περιγραφικές ιδιότητες που χρησιμοποιούνται για τα security patterns συνοψίζονται στον Πίνακα 2.

Πίνακας 2: Βασικές περιγραφικές ιδιότητες (Datatype Properties) των μοτίβων ασφάλειας στην οντολογία

Datatype Property	***Πληροφορία που αναπαριστά***
hasName	Όνομα του μοτίβου
hasAlias	Εναλλακτική ονομασία
hasDescription	Συνοπτική περιγραφή
hasIntent	Πρόθεση και βασικός σκοπός
hasContext	Πλαίσιο εφαρμογής
hasProblem	Πρόβλημα που αντιμετωπίζεται
hasForces	Δυνάμεις, περιορισμοί και συμβιβασμοί
hasSolution	Προτεινόμενη λύση
hasConsequence	Συνέπειες της εφαρμογής
hasImplementationExample	Παράδειγμα υλοποίησης

Πέρα από τις παραπάνω ιδιότητες, στο μοντέλο έχουν οριστεί πεδία που αφορούν την τεκμηρίωση και την προέλευση της πληροφορίας, όπως hasSourceCitation, hasOriginalSourceCitation, hasCatalogCitation, hasSourcePage, hasSourceUrl και hasValidationNote. Οι ιδιότητες αυτές παραμένουν περιγραφικές και διαφέρουν από τη δομημένη συσχέτιση ενός pattern με ένα individual της κλάσης BibliographicSource. Η διάκριση αυτή υπάρχει και στην τελική υλοποίηση.

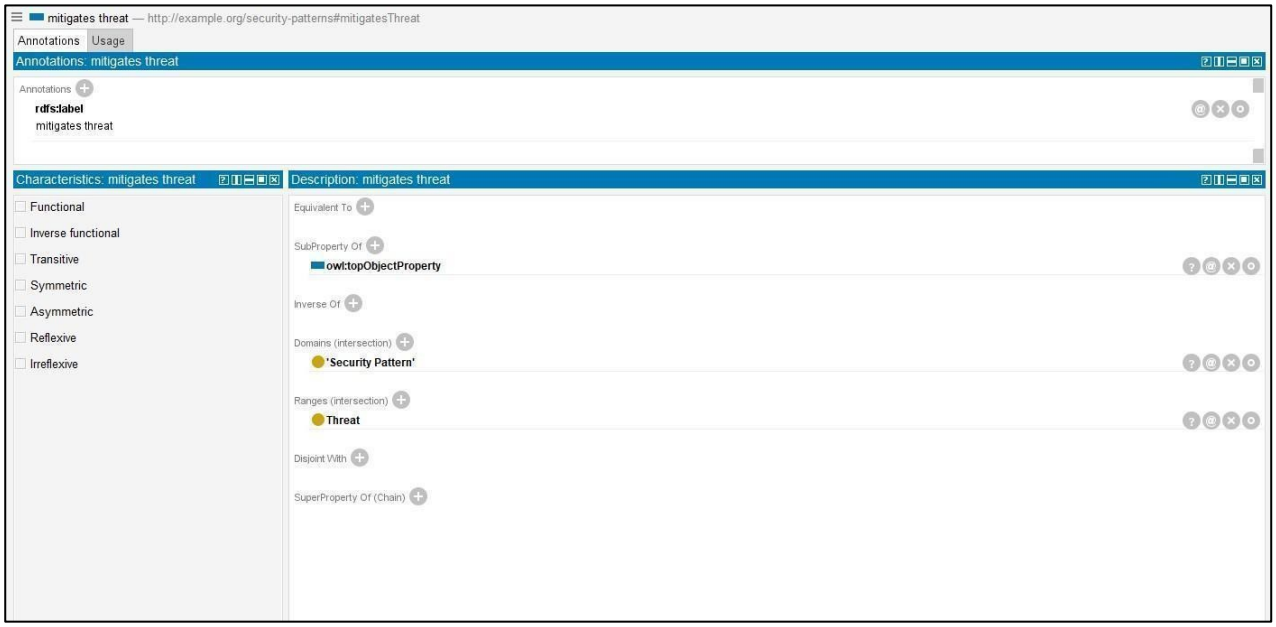
Οι σχέσεις που παρουσιάστηκαν εννοιολογικά στον Πίνακα 1 υλοποιήθηκαν ως Object Properties. Για παράδειγμα, η mitigatesThreat δεν αποθηκεύει πλέον το όνομα μιας απειλής ως κείμενο, αλλά συνδέει ένα SecurityPattern με συγκεκριμένο individual της κλάσης Threat. Αντίστοιχα, οι protectsAsset, supportsSecurityProperty και appliesAtPhase συνδέουν ένα pattern με individuals των κλάσεων ProtectedAsset, SecurityProperty και DevelopmentPhase. Το ίδιο ισχύει για τα RiskLevel, ApplicationContext, PatternLevel και ValidationStatus.

Για τις Object Properties ορίστηκαν κατάλληλα domain και range, ώστε να αποτυπώνεται με μεγαλύτερη ακρίβεια ο επιτρεπόμενος σημασιολογικός ρόλος κάθε σχέσης. Ενδεικτικά, η mitigatesThreat (δείτε Εικόνα 12) έχει ως domain την SecurityPattern και ως range την Threat, ενώ η documentedInSource συνδέει ένα SecurityPattern με μια BibliographicSource. Παράλληλα έχει οριστεί η αντίστροφη σχέση documentsPattern, η οποία επιτρέπει την πλοήγηση από μια βιβλιογραφική πηγή προς τα patterns που αυτή τεκμηριώνει. Η ίδια δομή καταγράφεται ήδη στην τρέχουσα ενότητα του εγγράφου.

Ιδιαίτερη έμφαση δόθηκε επίσης στην πληρότητα των σχέσεων μεταξύ των ίδιων των μοτίβων ασφάλειας. Κατά την τελική επιμέλεια της βάσης γνώσης εξετάστηκαν οι υπάρχουσες

συσχετίσεις και εμπλουτίστηκαν όπου υπήρχε ουσιαστική εννοιολογική συνάφεια. Στόχος δεν ήταν η τεχνητή αύξηση του αριθμού των συνδέσεων, αλλά η αποτύπωση σχέσεων που μπορούν να υποστηρίξουν τη σημασιολογική ανακάλυψη συγγενών ή συμπληρωματικών λύσεων ασφάλειας. Οι σχέσεις αυτές αναπαρίστανται ως Object Property Assertions προς άλλα SecurityPattern individuals και αξιοποιούνται τόσο στην αναλυτική παρουσίαση ενός μοτίβου όσο και στην αναζήτηση βάσει σχετιζόμενου pattern.

Αντίστοιχη επιμέλεια πραγματοποιήθηκε για τη συσχέτιση των μοτίβων με επίπεδα κινδύνου. Τα RiskLevel individuals χρησιμοποιήθηκαν ως κοινό, ελεγχόμενο λεξιλόγιο και τα ενεργά μοτίβα εμπλουτίστηκαν με τις κατάλληλες σημασιολογικές συσχετίσεις όπου αυτές μπορούσαν να τεκμηριωθούν από το περιεχόμενο και τον σκοπό τους. Η συμπλήρωση αυτή είναι ουσιώδης για την πρακτική αξιοποίηση της ιδιότητας hasRiskLevel, καθώς επιτρέπει την ανάκτηση μοτίβων με βάση το επίπεδο κινδύνου χωρίς να βασίζεται σε μη δομημένες λεκτικές αναφορές.



Εικόνα 12: Ορισμός της Object Property mitigatesThreat στο Protégé, με domain την SecurityPattern και range την Threat.

3.5.3 Individuals σχετικών εννοιών

Μετά τον ορισμό του σχήματος της οντολογίας δημιουργήθηκαν τα individuals που χρησιμοποιούνται ως πραγματικές τιμές των σημασιολογικών σχέσεων. Σε αυτά περιλαμβάνονται συγκεκριμένες απειλές, προστατευόμενα αγαθά, ιδιότητες ασφάλειας, φάσεις ανάπτυξης, επίπεδα κινδύνου, πλαίσια εφαρμογής, επίπεδα μοτίβου και βιβλιογραφικές πηγές.

Οι βιβλιογραφικές πηγές μοντελοποιήθηκαν ως individuals της κλάσης BibliographicSource. Για κάθε πηγή μπορούν να καταγράφονται στοιχεία όπως ο τίτλος, οι συγγραφείς, το έτος, ο τύπος της πηγής, το DOI και το URL. Η ανεξάρτητη μοντελοποίησή τους

επιτρέπει στην ίδια πηγή να τεκμηριώνει περισσότερα από ένα security patterns χωρίς να επαναλαμβάνονται τα βιβλιογραφικά δεδομένα σε κάθε pattern ξεχωριστά.

Εικόνα 13: Παράδειγμα βιβλιογραφικής πηγής και των μοτίβων ασφάλειας που τεκμηριώνονται από αυτή μέσω της σχέσης documentsPattern.

Η σχέση αυτή επιτρέπει να διατηρείται ρητά η προέλευση της γνώσης. Όταν διαφορετικές πηγές χρησιμοποιούν διαφορετικές ονομασίες ή περιγραφές για παρόμοια μοτίβα ασφαλείας, μπορούν να διατηρηθούν οι αντίστοιχες βιβλιογραφικές αναφορές και εναλλακτικές ονομασίες χωρίς να θεωρείται αυτομάτως ότι οι έννοιες είναι πλήρως ταυτόσημες. Το υπάρχον κείμενο της εργασίας ήδη αποτυπώνει αυτή τη σχεδιαστική επιλογή.

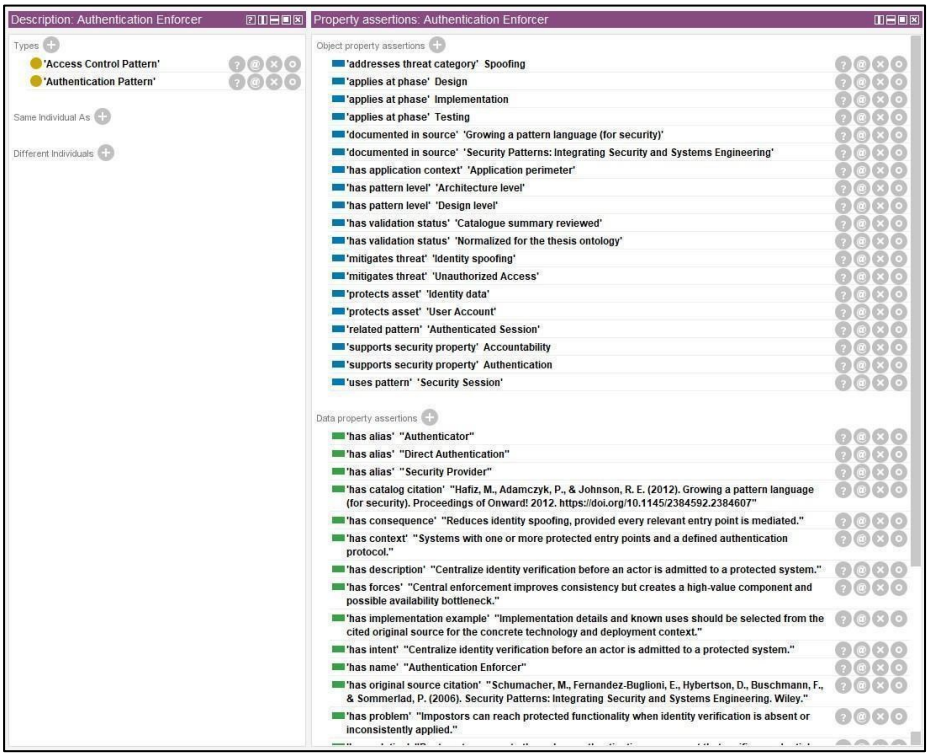
3.5.4 Security pattern individuals

Τα ίδια τα security patterns δημιουργήθηκαν ως Named Individuals της οντολογίας. Για κάθε pattern ορίζονται μία ή περισσότερες Class Assertions, οι περιγραφικές Datatype Property Assertions και οι απαιτούμενες Object Property Assertions προς τα υπόλοιπα individuals της βάσης γνώσης.

Η προσέγγιση αυτή σημαίνει ότι η δημιουργία ενός pattern δεν περιορίζεται στην αποθήκευση ενός ονόματος και μιας περιγραφής. Ένα pattern αποκτά δικό του IRI, ταξινομείται στις κατάλληλες κλάσεις και συνδέεται με τις υπόλοιπες έννοιες του μοντέλου. Η ίδια δομή διατηρείται και όταν δημιουργείται ένα νέο pattern μέσω της τελικής εφαρμογής: η υλοποίηση δημιουργεί Named Individual, Class Assertions, Data Property Assertions και Object Property Assertions.

Χαρακτηριστικό παράδειγμα αποτελεί το individual AuthenticationEnforcer. Το συγκεκριμένο μοτίβο έχει ταξινομηθεί ως AuthenticationPattern και AccessControlPattern. Συνδέεται με τις απειλές Identity Spoofing και Unauthorized Access, με τα προστατευόμενα αγαθά Identity Data και User Account και με τις ιδιότητες ασφάλειας Authentication και Accountability. Επιπλέον, συνδέεται με τις φάσεις Design, Implementation και Testing, με το Application Perimeter ως πλαίσιο εφαρμογής και με τα επίπεδα Architecture και Design. Παράλληλα, οι πληροφορίες που

αφορούν την πρόθεση, το πλαίσιο, το πρόβλημα, τις δυνάμεις, τη λύση και τις συνέπειες αποθηκεύονται μέσω των αντίστοιχων Datatype Properties.



Εικόνα 14: Παράδειγμα πλήρους σημασιολογικής περιγραφής του μοτίβου Authentication Enforcer στο Protégé.

Το παράδειγμα αυτό είναι σημαντικό επειδή δείχνει στην πράξη τη συνύπαρξη των δύο επιπέδων πληροφορίας: αφενός της ανθρώπινα αναγνώσιμης περιγραφής και αφετέρου των τυπικών σημασιολογικών σχέσεων με άλλες οντότητες.

```
<ClassAssertion>
    <Class IRI="#AuthenticationPattern" />
    <NamedIndividual IRI="#AuthenticationEnforcer" />
</ClassAssertion>

<DataPropertyAssertion>
    <DataProperty IRI="#hasName" />
    <NamedIndividual IRI="#AuthenticationEnforcer" />
    <Literal>Authentication Enforcer</Literal>
</DataPropertyAssertion>

<DataPropertyAssertion>
    <DataProperty IRI="#hasProblem" />
    <NamedIndividual IRI="#AuthenticationEnforcer" />
    <Literal>Impostors can reach protected functionality when identity
verification is absent or inconsistently applied.</Literal>
</DataPropertyAssertion>

<ObjectPropertyAssertion>
```

```
    <ObjectProperty IRI="#mitigatesThreat" />
    <NamedIndividual IRI="#AuthenticationEnforcer" />
    <NamedIndividual IRI="#IdentitySpoofing" />
</ObjectPropertyAssertion>

<ObjectPropertyAssertion>
    <ObjectProperty IRI="#protectsAsset" />
    <NamedIndividual IRI="#AuthenticationEnforcer" />
    <NamedIndividual IRI="#IdentityData" />
</ObjectPropertyAssertion>
```

Απόσπασμα OWL: Ενδεικτική σημασιολογική αναπαράσταση του μοτίβου Authentication Enforcer στην οντολογία.

3.5.5 Επικύρωση της οντολογίας

Η ολοκλήρωση του οντολογικού μοντέλου δεν θεωρήθηκε επαρκής χωρίς ανεξάρτητο έλεγχο της λογικής του συνέπειας. Η επικύρωση της οντολογίας πραγματοποιήθηκε στο Protégé πριν από την αξιοποίησή της από την εφαρμογή. Με αυτόν τον τρόπο διαχωρίζεται ο έλεγχος της ορθότητας του ίδιου του σημασιολογικού μοντέλου από τον έλεγχο του λογισμικού που το καταναλώνει.

Αρχικά επιθεωρήθηκαν η asserted ιεραρχία των κλάσεων, οι Object Properties και τα δηλωμένα domain και range των βασικών σχέσεων. Ελέγχθηκαν επίσης αντιπροσωπευτικά individuals, όπως το AuthenticationEnforcer, ώστε να διαπιστωθεί ότι οι ταξινομήσεις και οι σημασιολογικές συνδέσεις τους συμφωνούν με τον σχεδιασμό.

Στη συνέχεια χρησιμοποιήθηκε ο HermiT reasoner για την ταξινόμηση της οντολογίας και την παραγωγή της inferred class hierarchy. Ο HermiT είναι reasoner για OWL 2 και μπορεί να χρησιμοποιηθεί, μεταξύ άλλων, για έλεγχο συνέπειας και ταξινόμηση κλάσεων (Glimm et al., 2014).

Κατά τον συγκεκριμένο έλεγχο δεν προέκυψε ένδειξη ασυνέπειας και δεν εμφανίστηκαν ονομαστικές κλάσεις ως μη ικανοποιήσιμες κάτω από την owl:Nothing. Η inferred ιεραρχία που προέκυψε ήταν συμβατή με την asserted ιεραρχία που είχε σχεδιαστεί. Το αποτέλεσμα αυτό τεκμηριώνει ότι, για το περιεχόμενο και τα αξιώματα της συγκεκριμένης έκδοσης της οντολογίας, ο reasoner δεν εντόπισε λογική αντίφαση.

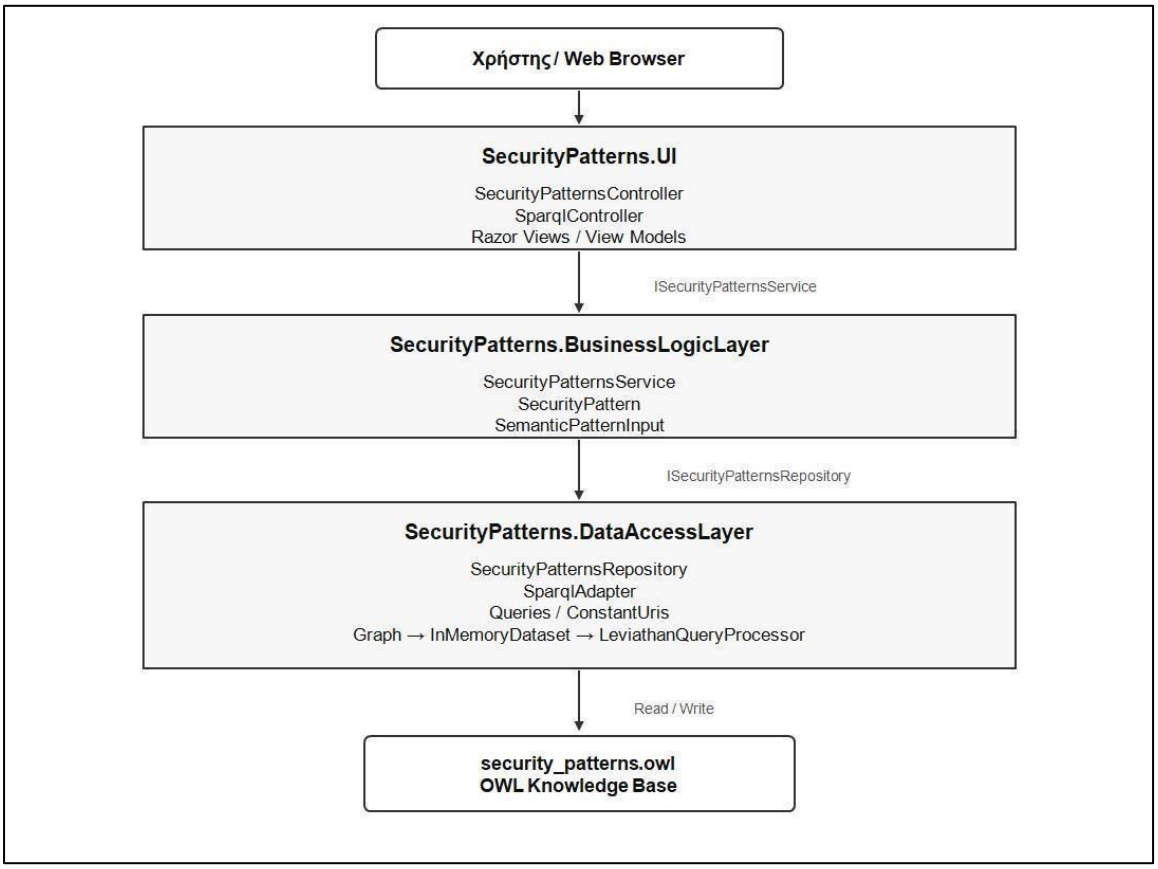
Εικόνα 15: Inferred ιεραρχία κλάσεων μετά την ταξινόμηση της οντολογίας με τον HermiT reasoner.

Η διαδικασία αυτή ολοκληρώνει το στάδιο δημιουργίας και αρχικής επικύρωσης της οντολογίας. Το αποτέλεσμα είναι ένα σημασιολογικό μοντέλο στο οποίο συνυπάρχουν η ιεραρχική ταξινόμηση των security patterns, οι περιγραφικές τους πληροφορίες, οι σχέσεις τους με έννοιες του πεδίου ασφάλειας, οι μεταξύ τους συσχετίσεις και η βιβλιογραφική τους προέλευση. Στην επόμενη ενότητα παρουσιάζεται η αρχιτεκτονική μέσω της οποίας η εφαρμογή αξιοποιεί το μοντέλο αυτό χωρίς να εξαρτά τη διεπαφή χρήστη από τις λεπτομέρειες της OWL αναπαράστασης.

3.6 Αρχιτεκτονική

Η εφαρμογή έχει σχεδιαστεί με πολυεπίπεδη αρχιτεκτονική, ώστε να διαχωρίζονται η παρουσίαση και η αλληλεπίδραση με τον χρήστη, η επιχειρησιακή επεξεργασία της σημασιολογικής πληροφορίας και η πρόσβαση στην οντολογία/σημασιολογική βάση γνώσης. Ο διαχωρισμός αυτός περιορίζει τη σύζευξη μεταξύ των επιμέρους τμημάτων του συστήματος και επιτρέπει κάθε επίπεδο να διαχειρίζεται διαφορετικό σύνολο αρμοδιοτήτων.

Η κύρια εφαρμογή αποτελείται από τρία έργα: SecurityPatterns.UI, SecurityPatterns.BusinessLogicLayer και SecurityPatterns.DataAccessLayer. Η κατεύθυνση των εξαρτήσεων ακολουθεί τη σειρά UI → Business Logic Layer → Data Access Layer. Το UI project αναφέρεται στο Business Logic Layer, ενώ το Business Logic Layer αναφέρεται στο Data Access Layer. Το τελευταίο δεν εξαρτάται από τα δύο ανώτερα επίπεδα, αλλά περιλαμβάνει τις βιβλιοθήκες που απαιτούνται για την επεξεργασία της σημασιολογικής πληροφορίας. Η φυσική δομή των project references της τελικής λύσης ακολουθεί ακριβώς αυτή την κατεύθυνση.



Εικόνα 16: Πολυεπίπεδη αρχιτεκτονική της εφαρμογής και ροή επικοινωνίας με τη σημασιολογική βάση γνώσης.

Η ***Εικόνα 16*** αποτυπώνει τόσο τα τρία επίπεδα της εφαρμογής όσο και τη θέση της οντολογίας ως μόνιμου σημασιολογικού πόρου. Ο χρήστης δεν επικοινωνεί απευθείας με το OWL αρχείο. Όλες οι ενέργειες πραγματοποιούνται μέσω της web διεπαφής και διασχίζουν διαδοχικά τα επίπεδα της εφαρμογής μέχρι να καταλήξουν στους μηχανισμούς ανάγνωσης ή τροποποίησης της οντολογίας.

3.6.1 Επίπεδο Παρουσίασης

Το SecurityPatterns.UI αποτελεί το επίπεδο παρουσίασης της εφαρμογής και έχει υλοποιηθεί ως ASP.NET Core MVC εφαρμογή σε .NET 8. Περιλαμβάνει τους controllers, τα view models και τα Razor views μέσω των οποίων ο χρήστης αλληλεπιδρά με το σύστημα. Το project εξαρτάται από το Business Logic Layer και δεν πραγματοποιεί απευθείας πρόσβαση στο RDF graph ή στο OWL αρχείο.

Κεντρικό στοιχείο του επιπέδου αυτού είναι ο SecurityPatternsController, ο οποίος διαχειρίζεται τις λειτουργίες προβολής, αναλυτικής εμφάνισης, δημιουργίας, επεξεργασίας και λογικής διαγραφής των μοτίβων. Κατά τη δημιουργία και την επεξεργασία, ο controller συγκεντρώνει τα δεδομένα της φόρμας σε ένα SemanticPatternInput, το οποίο περιέχει τόσο τις περιγραφικές τιμές όσο και τα IRI των επιλεγμένων οντολογικών οντοτήτων. Η επεξεργασία της σημασιολογικής πληροφορίας δεν πραγματοποιείται στον controller αλλά μεταβιβάζεται στην υπηρεσία του Business Logic Layer.

Στο ίδιο επίπεδο βρίσκεται και ο SparqlController, ο οποίος παρέχει τη χωριστή διεπαφή εκτέλεσης SPARQL ερωτημάτων. Ο controller αυτός χρησιμοποιεί την ίδια επιχειρησιακή υπηρεσία με την υπόλοιπη εφαρμογή και συνεπώς δεν παρακάμπτει την αρχιτεκτονική δομή για να αποκτήσει άμεση πρόσβαση στη βάση γνώσης. Η διεπαφή αυτή χρησιμοποιείται αποκλειστικά για επερωτήσεις ανάγνωσης.

3.6.2 Επίπεδο Επιχειρησιακής Λογικής

Το SecurityPatterns.BusinessLogicLayer αποτελεί το ενδιάμεσο επίπεδο μεταξύ της διεπαφής χρήστη και του μηχανισμού πρόσβασης στην οντολογία. Κεντρικό στοιχείο του είναι η SecurityPatternsService, η οποία υλοποιεί το ISecurityPatternsService (δηλαδή μια συγκεκριμένη προγραμματιστική διεπαφή ιστού) και αναλαμβάνει να μετατρέπει τα αποτελέσματα του Data Access Layer σε αντικείμενα που μπορούν να χρησιμοποιηθούν από την εφαρμογή.

Η υπηρεσία δεν περιορίζεται στην απλή μεταφορά δεδομένων. Κατά την ανάκτηση ενός pattern συνδυάζει πληροφορίες από διαφορετικές σημασιολογικές επερωτήσεις: τα περιγραφικά χαρακτηριστικά του, τις κλάσεις στις οποίες ανήκει, τις διαδρομές ταξινόμησης στην ιεραρχία rdfs:subClassOf, τις Object Property σχέσεις και τις βιβλιογραφικές πηγές. Τα αποτελέσματα μετατρέπονται σε αντικείμενα SecurityPattern, OntologyReference, OntologyHierarchyPath και

BibliographicSource, ώστε το UI να εργάζεται με ένα αντικειμενοστραφές μοντέλο αντί με ακατέργαστα SPARQL αποτελέσματα.

Η ίδια υπηρεσία αναλαμβάνει και την προετοιμασία των δεδομένων στις λειτουργίες μεταβολής. Το SemanticPatternInput που προέρχεται από το UI μετατρέπεται σε διακριτές συλλογές Data Properties και Object Properties πριν διαβιβαστεί στο repository. Με τον τρόπο αυτό το Data Access Layer δεν χρειάζεται να εξαρτάται από τα business models της εφαρμογής. Το interface του repository δέχεται γενικότερες δομές δεδομένων, όπως IRI, λίστες και dictionaries ιδιοτήτων, διατηρώντας σαφέστερο τον διαχωρισμό των επιπέδων.

3.6.3 Επίπεδο Πρόσβασης Δεδομένων

Το SecurityPatterns.DataAccessLayer συγκεντρώνει τους μηχανισμούς πρόσβασης, αναζήτησης και μεταβολής της σημασιολογικής πληροφορίας. Κύρια στοιχεία του είναι το ISecurityPatternsRepository, η υλοποίηση SecurityPatternsRepository, ο SparqlAdapter, οι προκαθορισμένες SPARQL επερωτήσεις του Queries.cs και οι σταθερές IRI του ConstantUris.cs.

Το SecurityPatternsRepository αποτελεί το σημείο επικοινωνίας του Business Logic Layer με το σημασιολογικό υποσύστημα. Για τις λειτουργίες ανάγνωσης επιλέγει τις κατάλληλες SPARQL επερωτήσεις και επιστρέφει τα αποτελέσματα προς την υπηρεσία. Για συχνά χρησιμοποιούμενα δεδομένα, όπως η συνολική λίστα των patterns, οι διαθέσιμες κλάσεις, οι απειλές και τα individuals συγκεκριμένων κλάσεων, διατηρεί προσωρινά αποτελέσματα σε cache. Μετά από semantic Create, Edit ή Delete, οι σχετικές caches ακυρώνονται ώστε οι επόμενες αναγνώσεις να αντανακλούν την ενημερωμένη κατάσταση.

Σε χαμηλότερο επίπεδο, ο SparqlAdapter αναλαμβάνει την πραγματική επικοινωνία με τη σημασιολογική αναπαράσταση. Κατά την αρχικοποίησή του φορτώνει την canonical ontology σε RDF Graph, δημιουργεί ένα InMemoryDataset και επάνω σε αυτό έναν LeviathanQueryProcessor. Οι αναγνώσεις πραγματοποιούνται μέσω SPARQL, ενώ οι semantic λειτουργίες δημιουργίας και επεξεργασίας μεταβάλλουν τις αντίστοιχες class, data και object property assertions και ενημερώνουν το in-memory graph ώστε η νέα κατάσταση να είναι άμεσα διαθέσιμη στις επόμενες επερωτήσεις.

Το Data Access Layer χρησιμοποιεί τη βιβλιοθήκη dotNetRDF 3.3.2 για την επεξεργασία RDF και SPARQL και περιλαμβάνει επίσης την OWLSharp 5.0.0 στις εξαρτήσεις του. Οι συγκεκριμένες τεχνολογίες αποτελούν λεπτομέρειες της υλοποίησης του κατώτερου επιπέδου και δεν εκτίθενται προς τα ανώτερα επίπεδα της εφαρμογής.

3.6.4 Σημασιολογική Βάση Γνώσης και Αναπαράσταση στη Μνήμη

Στο κατώτερο άκρο της αρχιτεκτονικής βρίσκεται το security_patterns.owl. Το αρχείο αυτό περιέχει το κύριο μόνιμο περιεχόμενο της σημασιολογικής βάσης γνώσης, δηλαδή το οντολογικό σχήμα, τα security pattern individuals, τις υπόλοιπες σημασιολογικές οντότητες και τις μεταξύ τους σχέσεις.

Δεν αποτελεί όμως από μόνο του το Data Access Layer. Η πρόσβαση και η διαχείρισή του πραγματοποιούνται αποκλειστικά μέσω των αντίστοιχων μηχανισμών του Data Access Layer.

Κατά την εκτέλεση της εφαρμογής δημιουργείται παράλληλα μία in-memory αναπαράσταση της οντολογίας. Η αναπαράσταση αυτή χρησιμοποιείται από τον SPARQL query processor για την εξυπηρέτηση των αναζητήσεων χωρίς να απαιτείται νέα φόρτωση του αρχείου για κάθε request. Μετά τις μεταβολές της οντολογίας ενημερώνεται και το ενεργό RDF graph, ώστε το μόνιμο και το λειτουργικό επίπεδο δεδομένων να παραμένουν συγχρονισμένα.

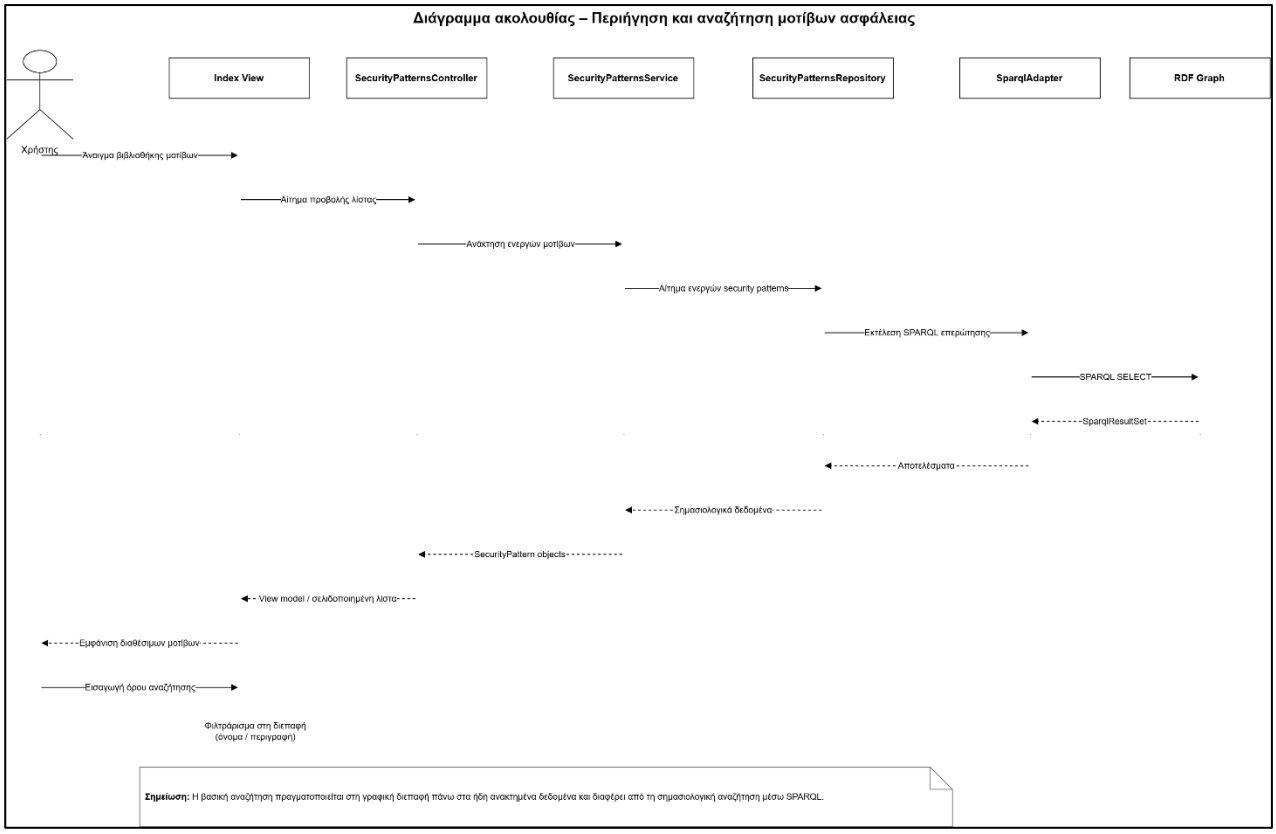
Κατά την εκκίνηση της web εφαρμογής πραγματοποιείται επιπλέον προφόρτωση ορισμένων συχνά χρησιμοποιούμενων δεδομένων. Συγκεκριμένα, ανακτώνται η συνολική λίστα των μοτίβων, οι κατηγορίες security patterns και τα threat individuals πριν από το πρώτο browser request. Η επιλογή αυτή περιορίζει την αρχική επιβάρυνση κατά την πρώτη πραγματική χρήση της εφαρμογής.

3.6.5 Ροή Ανάγνωσης

Η τυπική ροή ανάγνωσης ξεκινά από αίτημα του χρήστη προς το UI. Για παράδειγμα, κατά την αναλυτική προβολή ενός security pattern, ο SecurityPatternsController καλεί την αντίστοιχη μέθοδο της διεπαφής ISecurityPatternsService. Η υπηρεσία ζητά μέσω της διεπαφής ISecurityPatternsRepository τις απαιτούμενες πληροφορίες και το repository χρησιμοποιεί τον SparqlAdapter για την εκτέλεση των κατάλληλων SPARQL queries επάνω στο in-memory dataset.

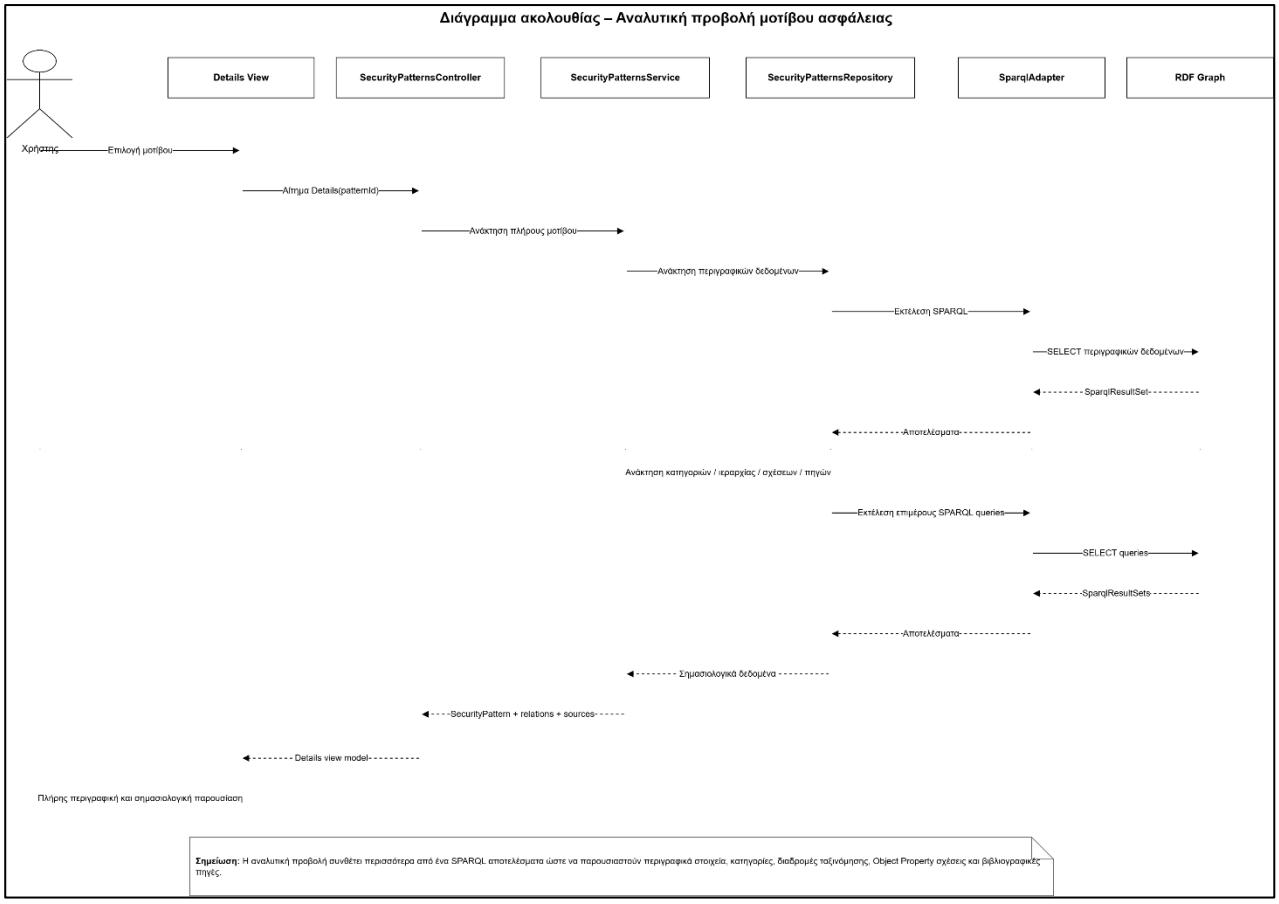
Τα αποτελέσματα επιστρέφουν στη συνέχεια προς την αντίθετη κατεύθυνση. Το Business Logic Layer τα μετατρέπει από SparqlResultSet σε αντικείμενα του μοντέλου και το UI τα αποδίδει στον χρήστη μέσω του αντίστοιχου view.

Η λειτουργία αναζήτησης εξειδικεύει την παραπάνω ροή ανάλογα με το επιλεγμένο κριτήριο. Οι αναζητήσεις ελεύθερου κειμένου αξιοποιούν περιγραφικά χαρακτηριστικά των μοτίβων, ενώ τα σημασιολογικά κριτήρια, όπως η φάση ανάπτυξης, το επίπεδο κινδύνου, η κλάση και οι σχέσεις μεταξύ μοτίβων, μεταφέρονται μέσω του Business Logic Layer στο επίπεδο πρόσβασης δεδομένων και αντιστοιχίζονται στις αντίστοιχες οντολογικές δομές. Έτσι, η γραφική διεπαφή παρέχει έναν ελεγχόμενο μηχανισμό σημασιολογικής αναζήτησης, συμπληρωματικό προς την ελεύθερη εκτέλεση επερωτήσεων από τη διεπαφή SPARQL.



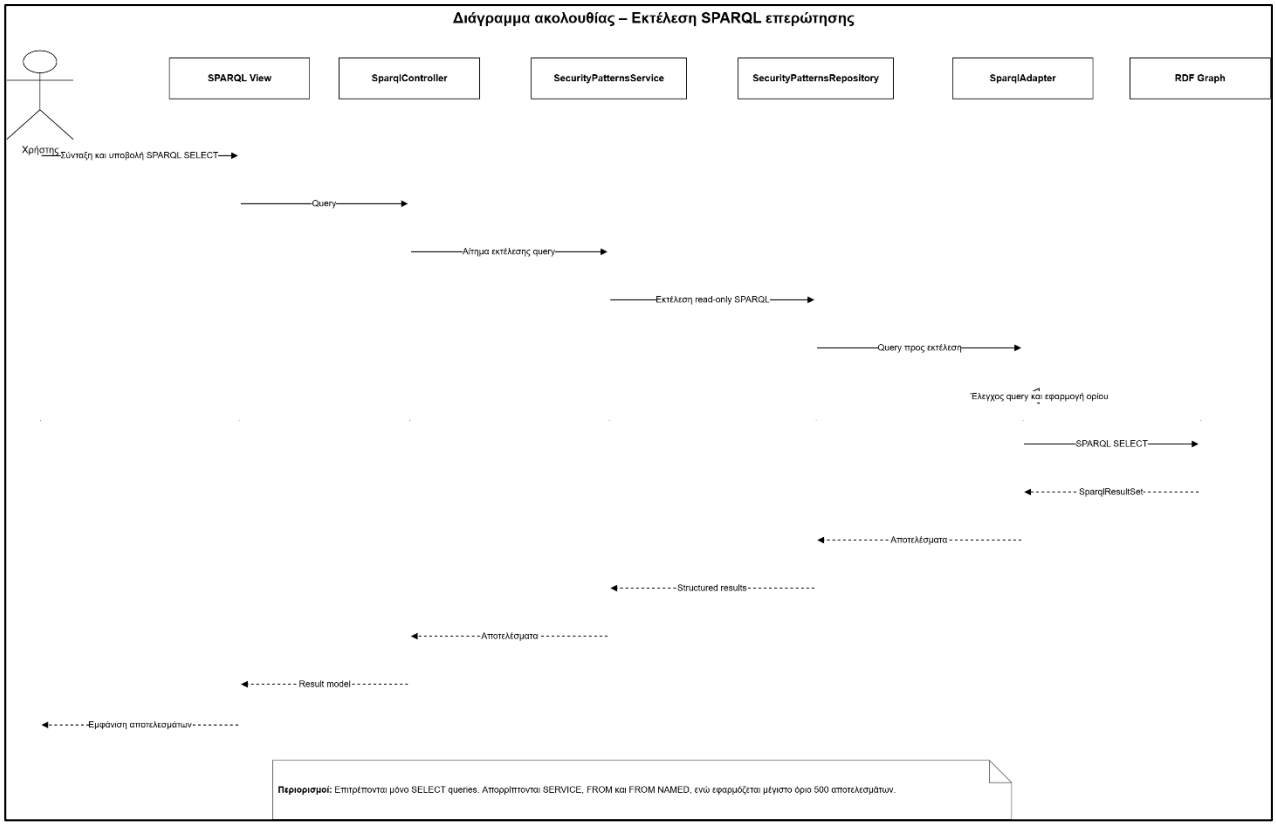
Εικόνα 17: Διάγραμμα ακολουθίας για την περιήγηση και αναζήτηση μοτίβων ασφάλειας.

Η αναλυτική προβολή ενός μοτίβου απαιτεί τη σύνθεση πληροφορίας από περισσότερες από μία σημασιολογικές επερωτήσεις. Εκτός από τα βασικά περιγραφικά στοιχεία του μοτίβου, ανακτώνται οι κατηγορίες και οι διαδρομές ταξινόμησής του, οι σχέσεις του με άλλες οντολογικές οντότητες και οι βιβλιογραφικές πηγές που το τεκμηριώνουν. Η ροή αυτή παρουσιάζεται στην Εικόνα 18.



Εικόνα 18: Διάγραμμα ακολουθίας για την αναλυτική προβολή ενός μοτίβου ασφάλειας.

Η τρίτη βασική ροή ανάγνωσης αφορά τη δυνατότητα υποβολής επερωτήσεων SPARQL από εξειδικευμένο χρήστη. Το ερώτημα διαβιβάζεται μέσω της ίδιας πολυεπίπεδης αρχιτεκτονικής και εκτελείται επάνω στην ενεργή RDF αναπαράσταση της οντολογίας. Πριν από την εκτέλεση εφαρμόζονται οι περιορισμοί της διεπαφής, ώστε να επιτρέπονται αποκλειστικά επερωτήσεις SELECT, να απορρίπτονται τα SERVICE, FROM και FROM NAMED και να εφαρμόζεται μέγιστο όριο 500 αποτελεσμάτων. Η αντίστοιχη αλληλουχία παρουσιάζεται στην Εικόνα 19.



Εικόνα 19: Διάγραμμα ακολουθίας για την εκτέλεση read-only SPARQL επερώτησης.

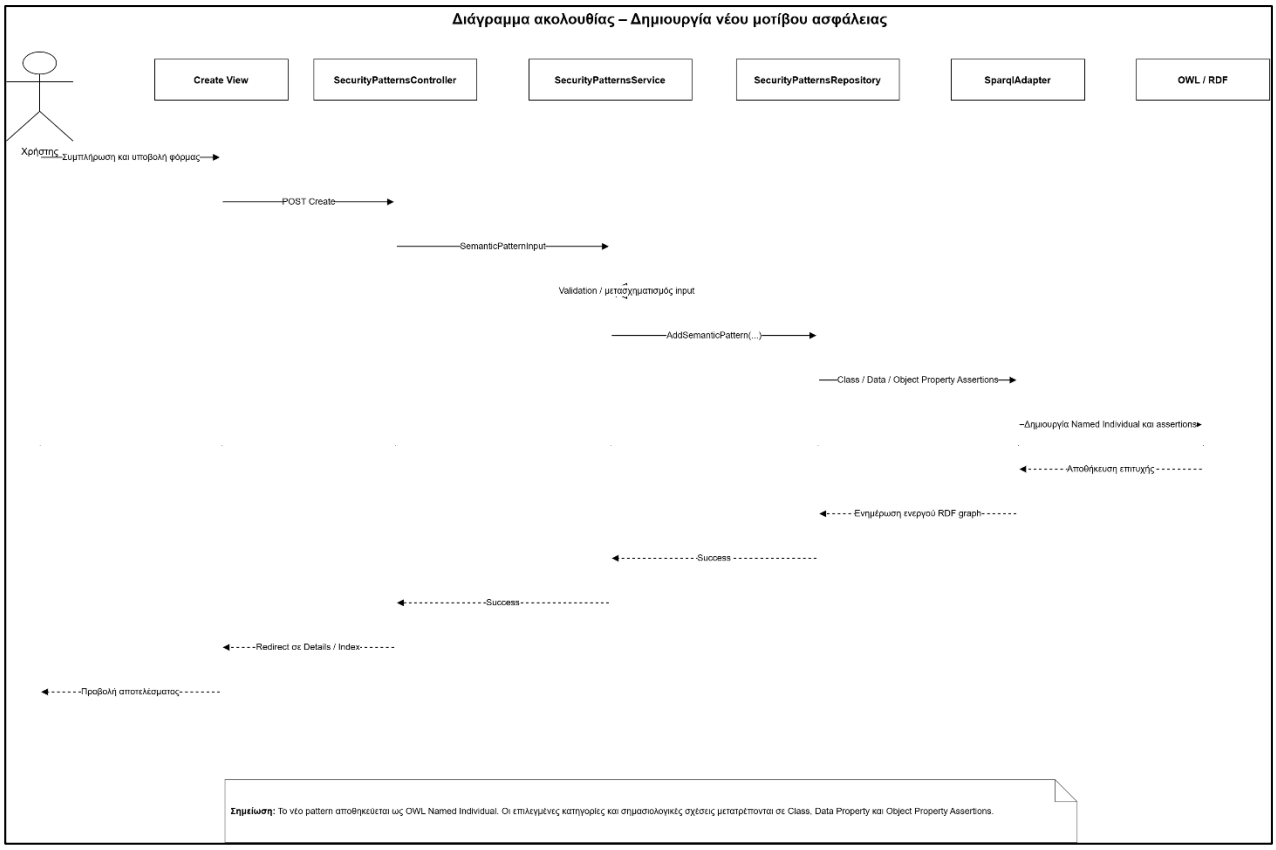
3.6.6 Ροή Μεταβολής

Η ροή δημιουργίας ή επεξεργασίας ακολουθεί την ίδια κατεύθυνση μεταξύ των επιπέδων, αλλά διαφοροποιείται ως προς την τελική ενέργεια στο Data Access Layer. Ο controller κατασκευάζει ένα SemanticPatternInput, η υπηρεσία ελέγχει και μετασχηματίζει τα δεδομένα και το repository μεταβιβάζει στο SparqlAdapter τις Class Assertions, Data Property Assertions και Object Property Assertions που πρέπει να προστεθούν ή να αντικατασταθούν. Η τρέχουσα Create/Edit ροή χρησιμοποιεί τα semantic methods AddSemanticPattern και EditSemanticPattern και όχι την παλαιότερη απλή μέθοδο AddPattern.

Μετά την επιτυχή μόνιμη αποθήκευση, ενημερώνεται το ενεργό RDF graph και ανακατασκευάζεται ο μηχανισμός SPARQL πάνω στην ενημερωμένη κατάσταση. Με αυτόν τον τρόπο η ίδια μεταβολή γίνεται ορατή τόσο σε επόμενες σελίδες της εφαρμογής όσο και στη read-only SPARQL διεπαφή χωρίς να απαιτείται επανεκκίνηση του συστήματος.

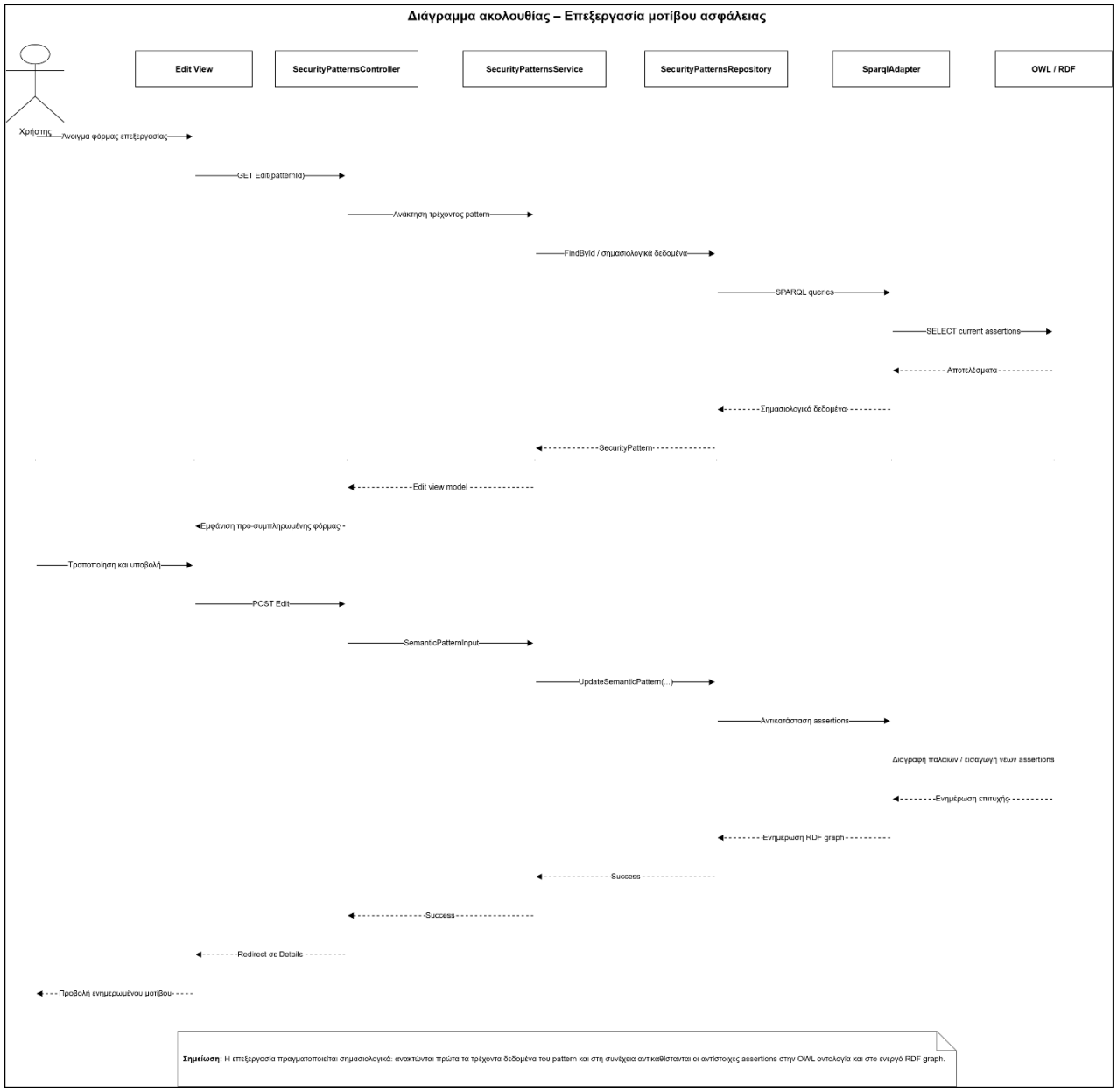
Η λογική διαγραφή ακολουθεί την ίδια αρχιτεκτονική διαδρομή. Το UI δεν αφαιρεί ούτε επεξεργάζεται απευθείας RDF triples, αλλά υποβάλλει το αίτημα στην υπηρεσία, η οποία καταλήγει μέσω του repository στους μηχανισμούς του Data Access Layer. Έτσι οι λεπτομέρειες της OWL αναπαράστασης παραμένουν απομονωμένες από το επίπεδο παρουσίασης.

Η δημιουργία νέου μοτίβου ασφάλειας ξεκινά από τη συμπλήρωση της αντίστοιχης φόρμας από τον χρήστη. Τα δεδομένα μεταφέρονται στον SecurityPatternsController και συγκεντρώνονται σε ένα SemanticPatternInput. Στη συνέχεια, η SecurityPatternsService πραγματοποιεί τους απαραίτητους ελέγχους και μετασχηματισμούς και μεταβιβάζει τα δεδομένα στο repository μέσω της σημασιολογικής λειτουργίας AddSemanticPattern. Στο Data Access Layer δημιουργείται το νέο OWL Named Individual και προστίθενται οι αντίστοιχες Class, Data Property και Object Property Assertions. Μετά την επιτυχή αποθήκευση ενημερώνεται και η ενεργή RDF αναπαράσταση. Η ροή αυτή παρουσιάζεται στην Εικόνα 20.



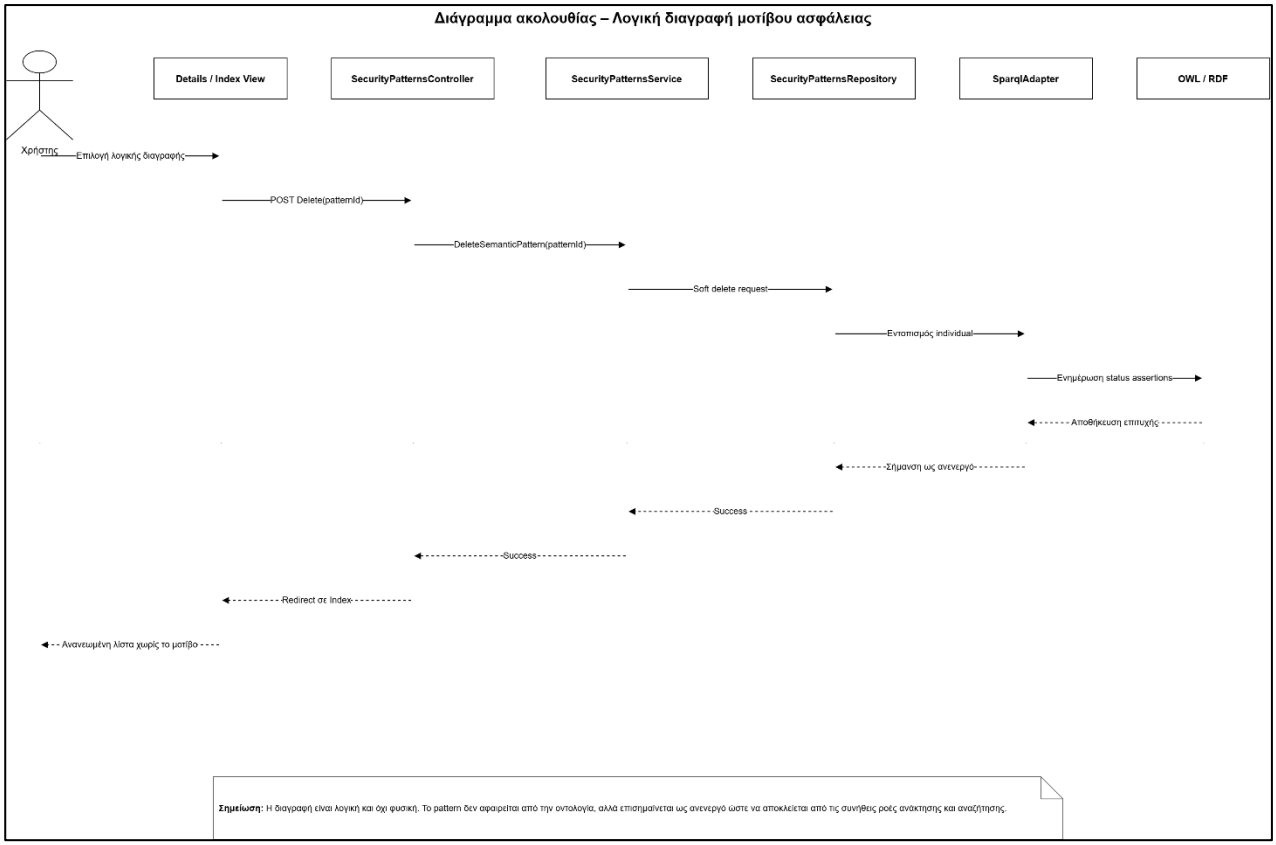
Εικόνα 20: Διάγραμμα ακολουθίας για τη δημιουργία νέου μοτίβου ασφάλειας.

Η επεξεργασία ενός υπάρχοντος μοτίβου ασφάλειας πραγματοποιείται σε δύο φάσεις. Αρχικά, ανακτώνται τα τρέχοντα σημασιολογικά δεδομένα του μοτίβου ώστε να παρουσιαστούν στον χρήστη σε προ-συμπληρωμένη φόρμα. Στη συνέχεια, μετά την υποβολή των τροποποιήσεων, η εφαρμογή μετασχηματίζει τα νέα δεδομένα σε σημασιολογικές δηλώσεις και ενημερώνει την OWL οντολογία αντικαθιστώντας τις αντίστοιχες assertions του συγκεκριμένου individual. Παράλληλα, ενημερώνεται και η ενεργή RDF αναπαράσταση της οντολογίας. Η ροή αυτή παρουσιάζεται στην Εικόνα 21.



Εικόνα 21: Διάγραμμα ακολουθίας για την επεξεργασία μοτίβου ασφάλειας.

Η απομάκρυνση ενός μοτίβου από τις βασικές ροές της εφαρμογής υλοποιείται ως λογική διαγραφή. Αντί να αφαιρεθεί πλήρως από την οντολογία, το αντίστοιχο individual επισημαίνεται ως ανενεργό μέσω της κατάλληλης σημασιολογικής ενημέρωσης. Με τον τρόπο αυτό διατηρείται η πληροφορία για λόγους ιχνηλασιμότητας και συνέπειας, ενώ το μοτίβο παύει να εμφανίζεται στις συνήθεις λειτουργίες περιήγησης και αναζήτησης. Η ακολουθία αυτή παρουσιάζεται στην Εικόνα 22.



Εικόνα 22: Διάγραμμα ακολουθίας για τη λογική διαγραφή μοτίβου ασφάλειας.

3.6.7 Διαχείριση Εξαρτήσεων

Η σύνδεση των επιπέδων πραγματοποιείται μέσω Dependency Injection και interfaces. Οι controllers εξαρτώνται από το ISecurityPatternsService και όχι από συγκεκριμένη υλοποίηση πρόσβασης στην οντολογία, ενώ η SecurityPatternsService εξαρτάται από το ISecurityPatternsRepository. Κατά την εκκίνηση της εφαρμογής εγγράφονται οι αντίστοιχες υλοποιήσεις στο dependency injection container.

Η δομή αυτή επιτρέπει στα ανώτερα επίπεδα να γνωρίζουν μόνο τις υπηρεσίες που απαιτούν για την εκτέλεση των λειτουργιών τους και όχι τον τρόπο με τον οποίο η πληροφορία φορτώνεται από OWL, μετατρέπεται σε RDF graph ή αναζητείται μέσω SPARQL.

Οι βασικές αρμοδιότητες των αρχιτεκτονικών στοιχείων συνοψίζονται στον Πίνακα 3.

Πίνακας 3: Βασικά στοιχεία και αρμοδιότητες της αρχιτεκτονικής της εφαρμογής

Αρχιτεκτονικό στοιχείο	***Κύρια στοιχεία***	***Βασική αρμοδιότητα***
SecurityPatterns.UI	Controllers, View Models, Razor Views	Αλληλεπίδραση με τον χρήστη και παρουσίαση της πληροφορίας
SecurityPatterns.BusinessLogic Layer	SecurityPatternsService, SecurityPattern, SemanticPatternInput	Επιχειρησιακή ροή, validation, μετασχηματισμός SPARQL αποτελεσμάτων και σημασιολογικών inputs
SecurityPatterns.DataAccessLayer	Repository, SparqlAdapter, Queries, URI constants	Ανάγνωση, επερώτηση και μεταβολή της οντολογίας
In-memory semantic representation	Graph, InMemoryDataset, LeviathanQueryProcessor	Εκτέλεση SPARQL και άμεση πρόσβαση στην ενεργή κατάσταση της γνώσης
Persistent ontology	security_patterns.owl	Μόνιμη αποθήκευση του οντολογικού σχήματος, των individuals και των assertions

Η αρχιτεκτονική αυτή επιτρέπει, επομένως, στη σημασιολογική γνώση να παραμένει απομονωμένη από τη διεπαφή χρήστη χωρίς να χάνεται η δυνατότητα πλήρους αξιοποίησής της. Το UI χειρίζεται την αλληλεπίδραση, το Business Logic Layer μεταφράζει μεταξύ του μοντέλου της εφαρμογής και της σημασιολογικής πληροφορίας, ενώ το Data Access Layer αναλαμβάνει τη διαχείριση RDF/OWL και SPARQL. Στην επόμενη ενότητα παρουσιάζεται αναλυτικά ο τρόπος με τον οποίο οι παραπάνω αρχιτεκτονικές επιλογές υλοποιούνται στον κώδικα.

3.7 Υλοποίηση

Η υλοποίηση της εφαρμογής πραγματοποιήθηκε σε .NET 8 με χρήση του ASP.NET Core MVC για το επίπεδο παρουσίασης και των τεχνολογιών OWL, RDF και SPARQL για τη διαχείριση της σημασιολογικής πληροφορίας. Όπως περιγράφηκε στην προηγούμενη ενότητα, η εφαρμογή οργανώνεται σε τρία διακριτά επίπεδα. Στην παρούσα ενότητα παρουσιάζονται οι κυριότεροι μηχανισμοί μέσω των οποίων η αρχιτεκτονική αυτή μεταφέρεται στον πηγαίο κώδικα.

Στόχος της παρουσίασης δεν είναι η παράθεση του συνόλου του κώδικα της εφαρμογής, αλλά η τεκμηρίωση των σημείων που είναι καθοριστικά για τη λειτουργία του σημασιολογικού πλαισίου: η σύνδεση με την κεντρική οντολογία, η φόρτωση και επερώτηση του RDF γράφου, η μετατροπή των αποτελεσμάτων σε αντικείμενα της εφαρμογής, η σημασιολογική δημιουργία και επεξεργασία patterns, η λογική διαγραφή και η παροχή ελεγχόμενης πρόσβασης μέσω SPARQL.

3.7.1 Αρχικοποίηση της εφαρμογής και σύνδεση με την οντολογία

Η εφαρμογή χρησιμοποιεί ως κεντρική πηγή δεδομένων το αρχείο security_patterns.owl που βρίσκεται στο root directory της λύσης. Η διαδρομή του αρχείου δεν έχει ενσωματωθεί στα

επιμέρους components, αλλά ορίζεται στο configuration του UI project μέσω της παραμέτρου OwlFilePath: "OwlFilePath": "../security_patterns.owl"

Η επιλογή αυτή επιτρέπει την αλλαγή της θέσης της οντολογίας χωρίς τροποποίηση του κώδικα του Business Logic Layer ή του Data Access Layer. Η πραγματική ρύθμιση της τελικής εφαρμογής δείχνει στο root security_patterns.owl.

Κατά την εκκίνηση, το Program.cs ανακτά την τιμή αυτή, ελέγχει ότι έχει οριστεί και, εάν πρόκειται για σχετική διαδρομή, τη μετατρέπει σε απόλυτη με βάση το content root της εφαρμογής. Η σχετική υλοποίηση παρουσιάζεται στο Παράρτημα Ι, Ενότητα Ι.1, όπου παρατίθεται το απόσπασμα του Program.cs που ανακτά και κανονικοποιεί τη διαδρομή της οντολογίας και αρχικοποιεί τις υπηρεσίες της εφαρμογής.

Η διαδρομή μεταβιβάζεται στη συνέχεια μέσω του μηχανισμού Dependency Injection μέχρι το Data Access Layer. Έτσι, τα ανώτερα επίπεδα της εφαρμογής δεν χρειάζεται να γνωρίζουν ούτε τη φυσική θέση της οντολογίας ούτε τον τρόπο με τον οποίο αυτή φορτώνεται. Η ίδια διαδικασία χρησιμοποιείται στην τελική έκδοση του Program.cs.

Κατά την εκκίνηση πραγματοποιείται επίσης προφόρτωση ορισμένων συχνά χρησιμοποιούμενων δεδομένων, όπως η λίστα των security patterns, οι διαθέσιμες κατηγορίες και τα threat individuals. Με αυτόν τον τρόπο οι σχετικές caches έχουν ήδη αρχικοποιηθεί πριν από το πρώτο πραγματικό αίτημα του χρήστη.

3.7.2 Φόρτωση της OWL οντολογίας και δημιουργία του RDF γράφου

Η πραγματική φόρτωση του αρχείου πραγματοποιείται από τον SparqlAdapter. Κατά την κατασκευή του αντικειμένου, η οντολογία φορτώνεται σε ένα RDF Graph. Πάνω από το graph δημιουργείται ένα InMemoryDataset και στη συνέχεια ένας LeviathanQueryProcessor, ο οποίος χρησιμοποιείται για την εκτέλεση των SPARQL queries. Ενδεικτικό απόσπασμα της υλοποίησης του SparqlAdapter και της δημιουργίας του in-memory RDF/SPARQL μηχανισμού παρατίθεται στο Παράρτημα Ι, Ενότητα Ι.2.

Η χρήση in-memory representation σημαίνει ότι η εφαρμογή δεν επαναδιαβάζει ολόκληρο το OWL αρχείο σε κάθε απλό αίτημα. Οι επερωτήσεις εκτελούνται επάνω στο ήδη φορτωμένο γράφημα, ενώ μετά από μεταβολές ο query engine ανακατασκευάζεται ώστε να λειτουργεί με την ενημερωμένη κατάσταση.

3.7.3 Ανάκτηση μοτίβων μέσω SPARQL

Η ανάγνωση της σημασιολογικής πληροφορίας πραγματοποιείται μέσω προκαθορισμένων SPARQL queries που βρίσκονται στο Queries.cs. Η βασική επερώτηση για την ανάκτηση των ενεργών security patterns δεν αναζητά αποκλειστικά individuals τα οποία έχουν δηλωθεί απευθείας ως SecurityPattern. Αντίθετα, αναζητά individuals που ανήκουν σε οποιαδήποτε κλάση της ιεραρχίας κάτω από την SecurityPattern.

Η χρήση του property path rdfs:subClassOf* είναι σημαντική, επειδή επιτρέπει στην εφαρμογή να αναγνωρίζει ως security pattern ένα individual που έχει ταξινομηθεί, για παράδειγμα, ως AuthenticationPattern ή ως κάποια βαθύτερη εξειδίκευση (συνεπώς γίνεται εκμετάλλευση της rdfs:subClassOf αναδρομικά εντός του γράφου για να εντοπιστούν οι έμμεσες υποκλάσεις της κλάσης SecurityPattern). Η πλήρης βασική επερώτηση που χρησιμοποιείται για την ανάκτηση των ενεργών μοτίβων ασφάλειας παρατίθεται στο Παράρτημα Ι, Ενότητα Ι.3.

Το ίδιο query εξαιρεί τα patterns τα οποία έχουν επισημανθεί ως λογικά διαγραμμένα. Η εφαρμογή συνεπώς δεν χρειάζεται να αφαιρεί ένα individual από την οντολογία για να πάψει να εμφανίζεται στην κανονική λειτουργία της. Η συγκεκριμένη λογική είναι η πραγματική ShowAllQuery της τελικής εφαρμογής.

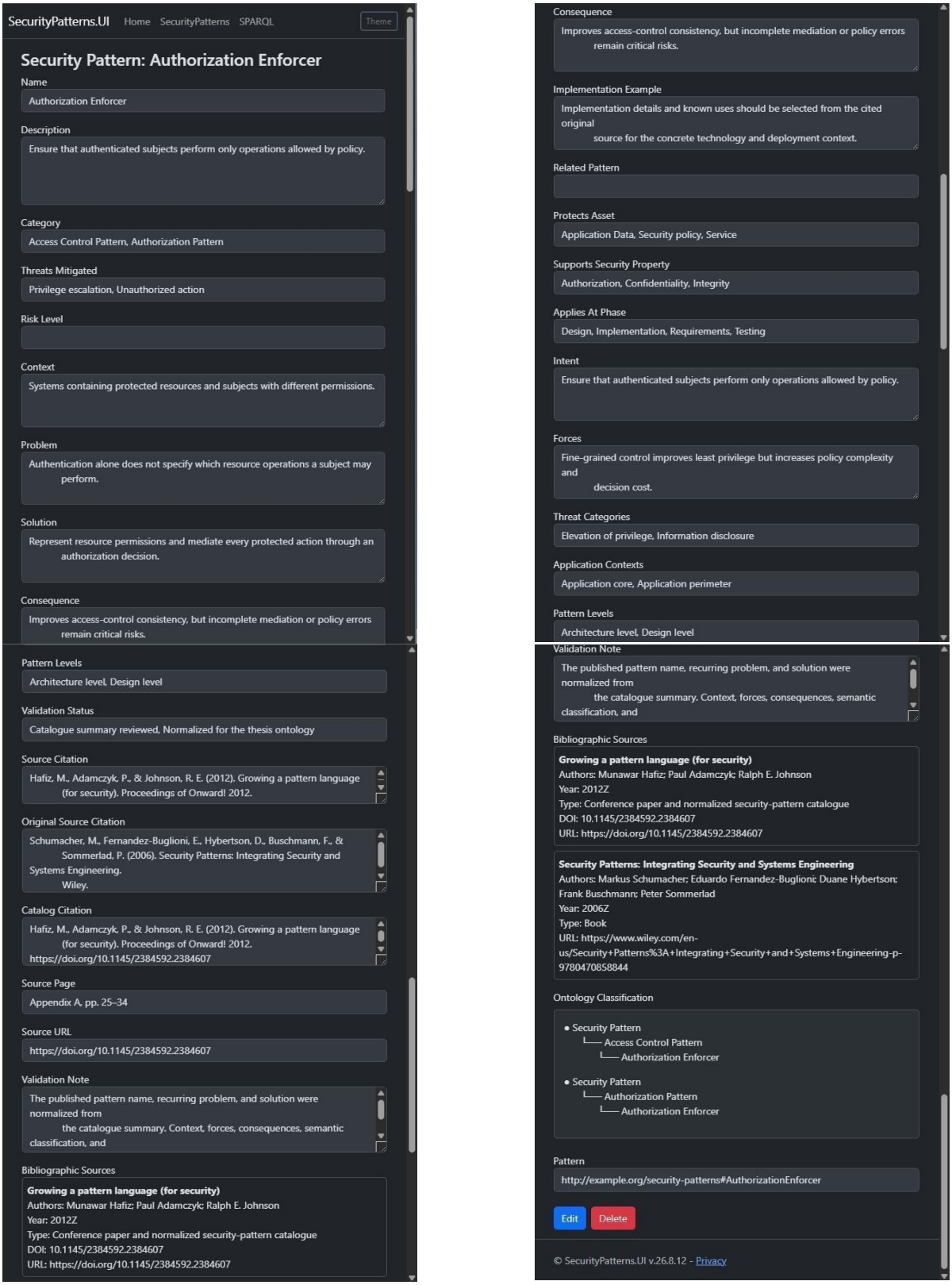
Για την αναλυτική προβολή ενός pattern χρησιμοποιούνται περισσότερες από μία επερωτήσεις. Ανακτώνται χωριστά η περιγραφική πληροφορία, οι κατηγορίες, η ιεραρχία ταξινόμησης, οι Object Property σχέσεις και οι βιβλιογραφικές πηγές. Η επιλογή αυτή αποφεύγει τη δημιουργία ενός πολύ μεγάλου συνδυαστικού αποτελέσματος όταν ένα pattern διαθέτει πολλαπλές κατηγορίες, απειλές, assets, phases και πηγές.

3.7.4 Μετατροπή των σημασιολογικών αποτελεσμάτων στο μοντέλο της εφαρμογής

Τα SparqlResultSet που επιστρέφονται από το Data Access Layer δεν μεταφέρονται απευθείας στο UI. Η SecurityPatternsService μετατρέπει τα αποτελέσματα σε αντικείμενα του επιχειρησιακού μοντέλου.

Κατά την ανάκτηση ενός συγκεκριμένου pattern, η υπηρεσία αρχικά δημιουργεί το αντικείμενο SecurityPattern από τα περιγραφικά δεδομένα. Στη συνέχεια φορτώνει τις κατηγορίες του από το rdf:type, δημιουργεί τις διαδρομές ταξινόμησης με βάση τις σχέσεις rdfs:subClassOf, ανακτά τις Object Property σχέσεις και τέλος φορτώνει τις συνδεδεμένες βιβλιογραφικές πηγές. Η ιεραρχία δεν είναι hard-coded στην υπηρεσία, αλλά προκύπτει από τις πραγματικές σχέσεις της οντολογίας.

Η προσέγγιση αυτή επιτρέπει στο UI να χειρίζεται έννοιες όπως Categories, MitigatedThreats, ProtectedAssets, SecurityProperties, DevelopmentPhases και BibliographicSources ως δομημένα αντικείμενα με IRI και label και όχι ως ακατέργαστες RDF τιμές.



Εικόνα 23: Αναλυτική παρουσίαση της σημασιολογικής πληροφορίας του Authorization Enforcer στη διεπαφή της εφαρμογής.

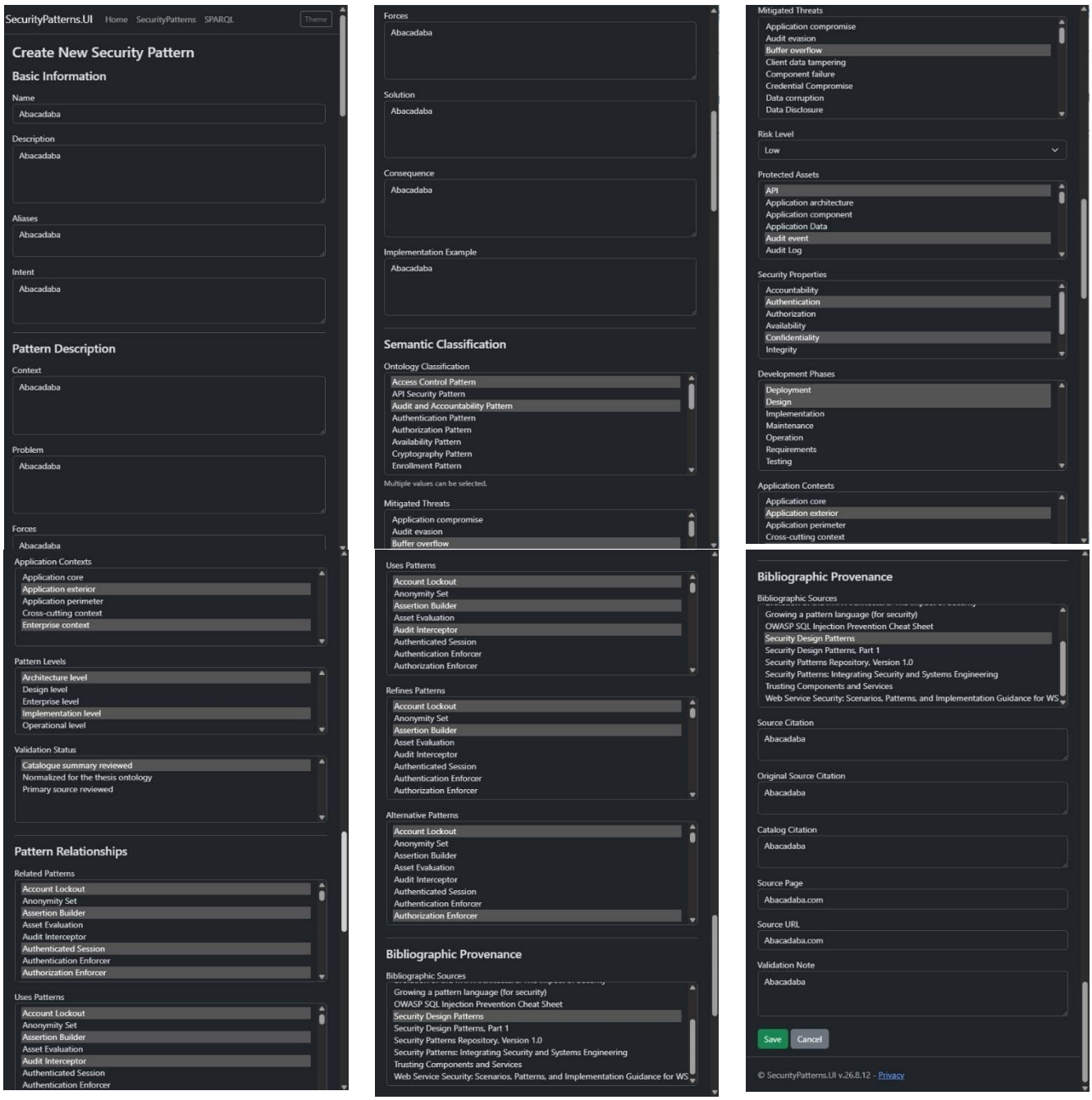
3.7.5 Δημιουργία και επεξεργασία security patterns

Η τελική εφαρμογή δεν δημιουργεί πλέον patterns μέσω της παλαιότερης απλής ροής AddPattern(name, description, category, threatsMitigated). Η δημιουργία και η επεξεργασία πραγματοποιούνται μέσω του SemanticPatternInput, το οποίο διαχωρίζει ρητά τις περιγραφικές τιμές από τα IRI των σημασιολογικών οντοτήτων.

Στα περιγραφικά δεδομένα περιλαμβάνονται μεταξύ άλλων το Name, Description, Intent, Context, Problem, Forces, Solution και Consequence. Αντίθετα, οι κατηγορίες, οι απειλές, τα assets, οι ιδιότητες ασφάλειας, οι φάσεις ανάπτυξης και οι σχέσεις προς άλλα patterns μεταφέρονται ως IRI. Με αυτόν τον τρόπο η γραφική φόρμα μπορεί να δημιουργήσει πραγματικές OWL σχέσεις χωρίς να απαιτεί από τον χρήστη να γνωρίζει ή να πληκτρολογεί RDF/OWL syntax.

Ο SecurityPatternsController ανακτά τις επιτρεπόμενες τιμές από την ίδια την οντολογία και τις χρησιμοποιεί για τη συμπλήρωση των επιλογών των Create και Edit views. Οι διαθέσιμες κατηγορίες, threats, risk levels, assets, security properties, phases, contexts, pattern levels, validation statuses, related patterns και bibliographic sources δεν αποτελούν στατικές λίστες του UI.

Στην τελική έκδοση της βάσης γνώσης δόθηκε επιπλέον έμφαση στην πληρότητα των πεδίων Problem, Solution και ImplementationExample. Τα πεδία αυτά δεν αντιμετωπίζονται ως προαιρετικό συνοδευτικό κείμενο, αλλά ως βασικά στοιχεία της περιγραφής ενός αξιοποιήσιμου security pattern. Για τον λόγο αυτό εμπλουτίστηκε το περιεχόμενο των ενεργών μοτίβων και προσαρμόστηκαν οι φόρμες δημιουργίας και επεξεργασίας, ώστε η καταχώριση να καθοδηγεί τον χρήστη προς σαφή περιγραφή του προβλήματος, της προτεινόμενης λύσης και ενός εφαρμόσιμου παραδείγματος υλοποίησης.



Εικόνα 24: Φόρμα δημιουργίας security pattern με περιγραφικά στοιχεία και επιλογές που προέρχονται από την οντολογία.

Στο Data Access Layer η δημιουργία ενός μοτίβου οδηγεί στην μετατροπή του αντίστοιχου αντικειμένου σε πραγματικές OWL assertions. Ο SparqlAdapter δημιουργεί νέο Named Individual, προσθέτει τις Class Assertions, το label, τις Data Property Assertions και τις Object Property Assertions και στη συνέχεια αποθηκεύει την ενημερωμένη οντολογία. Ενδεικτικό απόσπασμα της διαδικασίας δημιουργίας ενός νέου OWL Named Individual και των αντίστοιχων σημασιολογικών assertions παρατίθεται στο Παράρτημα Ι, Ενότητα Ι.4.

Μετά την αποθήκευση στο OWL αρχείο, οι ίδιες πληροφορίες προστίθενται και στο ενεργό RDF graph και ανακατασκευάζεται ο query processor. Έτσι το νέο pattern είναι άμεσα διαθέσιμο χωρίς επανεκκίνηση της εφαρμογής.

Η επεξεργασία ακολουθεί αντίστοιχη λογική. Αντί να δημιουργείται νέο individual, αντικαθίστανται ελεγχόμενα οι Class, Data Property και Object Property Assertions του υπάρχοντος pattern. Πριν από την αντικατάσταση αποθηκεύεται σχόλιο με το προηγούμενο όνομα και την προηγούμενη περιγραφή, παρέχοντας ένα βασικό ίχνος της μεταβολής.

3.7.6 Λογική διαγραφή

Η διαγραφή ενός pattern έχει υλοποιηθεί ως soft delete. Το individual δεν αφαιρείται φυσικά από την οντολογία και οι υπάρχουσες σημασιολογικές του σχέσεις δεν καταστρέφονται. Αντίθετα, καταγράφεται σχόλιο που περιλαμβάνει βασικά προηγούμενα στοιχεία και η isDeleted λαμβάνει την boolean τιμή true. Η υλοποίηση της λογικής διαγραφής και της ενημέρωσης της ιδιότητας isDeleted παρουσιάζεται στο Παράρτημα Ι, Ενότητα Ι.5.

Μετά την αποθήκευση (του «διεγραμμένου» μοτίβου εντός της οντολογίας) ενημερώνεται και η in-memory αναπαράσταση και ανακατασκευάζεται το query engine. Οι κανονικές queries (επερωτήσεις) της εφαρμογής αποκλείουν τα entities/individuals με isDeleted = true, με αποτέλεσμα το pattern να εξαφανίζεται από τη συνήθη διεπαφή χωρίς να χάνεται από τη βάση γνώσης.

3.7.7 Διεπαφή SPARQL

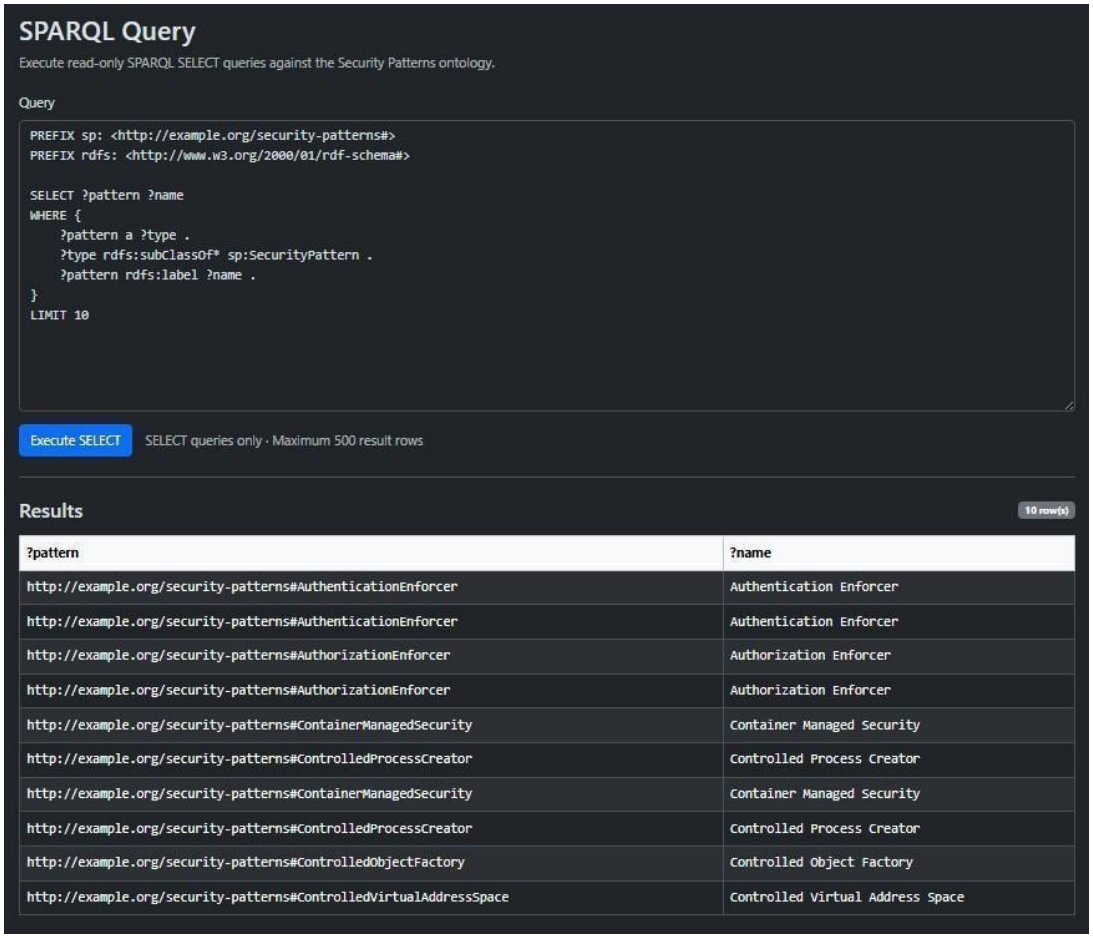
Εκτός από τις προκαθορισμένες λειτουργίες της web εφαρμογής, υλοποιήθηκε ξεχωριστή σελίδα μέσω της οποίας ένας εξειδικευμένος χρήστης μπορεί να υποβάλει δική του SPARQL επερώτηση.

Η λειτουργία αυτή χρησιμοποιεί την ίδια διεπαφή ISecurityPatternsService και συνεπώς ακολουθεί την ίδια αρχιτεκτονική διαδρομή με τις υπόλοιπες λειτουργίες. Ειδικότερα, ο SparqlController λαμβάνει το query από το αντίστοιχο view model και το μεταβιβάζει στην υπηρεσία για εκτέλεση, μέσω της προαναφερόμενης διεπαφής.

Επειδή η διεπαφή δέχεται query που έχει συντάξει ο χρήστης, εφαρμόζονται πρόσθετοι περιορισμοί, για λόγους ασφάλειας και απόδοσης. Επιτρέπονται αποκλειστικά queries της οικογένειας SELECT, απαγορεύονται τα SERVICE, FROM και FROM NAMED, ενώ εφαρμόζεται μέγιστο όριο 500 αποτελεσμάτων ανεξάρτητα από το query που υποβάλλεται. Οι έλεγχοι που

εφαρμόζονται πριν από την εκτέλεση μιας επερώτησης που έχει υποβληθεί από τον χρήστη παρουσιάζονται στο Παράρτημα Ι, Ενότητα Ι.6.

Η επιλογή αυτή διατηρεί τη δυνατότητα διερεύνησης της οντολογίας χωρίς να μετατρέπει τη σελίδα σε γενικό μηχανισμό τροποποίησης ή απομακρυσμένης πρόσβασης RDF. Οι συγκεκριμένοι περιορισμοί εφαρμόζονται στην τελική ExecuteSelectQuery.



Εικόνα 25: Εκτέλεση read-only SPARQL SELECT query μέσω της ειδικής διεπαφής της εφαρμογής.

3.7.8 Διεπαφή χρήστη και πολυκριτηριακή αναζήτηση μοτίβων

Η κύρια σελίδα της εφαρμογής λειτουργεί ως σημείο περιήγησης και αναζήτησης στη βιβλιοθήκη των μοτίβων ασφάλειας. Εκτός από την παρουσίαση των διαθέσιμων μοτίβων, παρέχει μηχανισμό αναζήτησης στον οποίο ο χρήστης επιλέγει αρχικά το είδος του κριτηρίου που επιθυμεί να χρησιμοποιήσει. Η τελική υλοποίηση υποστηρίζει αναζήτηση βάσει ονόματος (Name), περιγραφής (Description), φάσης ανάπτυξης (Development Phase), επιπέδου κινδύνου (Risk Level), κλάσης μοτίβου (Pattern Class) και συσχετιζόμενου μοτίβου (Related Pattern).

Η επιλογή του κριτηρίου καθορίζει τόσο τον τρόπο εισαγωγής της τιμής όσο και τη διαδικασία αναζήτησης. Για πεδία ελεύθερου κειμένου, όπως το όνομα και η περιγραφή, ο χρήστης εισάγει τον αντίστοιχο όρο, ενώ για χαρακτηριστικά που αντιστοιχούν σε οντολογικές οντότητες

χρησιμοποιούνται οι διαθέσιμες τιμές της βάσης γνώσης. Με τον τρόπο αυτό αποφεύγεται η εισαγωγή αυθαίρετων τιμών και η αναζήτηση παραμένει συνεπής με το σημασιολογικό μοντέλο.

Ιδιαίτερη σημασία έχουν τα κριτήρια Development Phase, Risk Level, Pattern Class και Related Pattern, καθώς η αναζήτηση δεν περιορίζεται στην απλή σύγκριση εμφανιζόμενου κειμένου. Αντίθετα, αξιοποιεί τις αντίστοιχες κλάσεις και σχέσεις που έχουν αποθηκευτεί στην OWL οντολογία. Επομένως, η γραφική διεπαφή παρέχει στον χρήστη μέρος της δυνατότητας σημασιολογικής διερεύνησης χωρίς να απαιτεί γνώση της SPARQL ή της εσωτερικής RDF αναπαράστασης.

Η εκτέλεση της αναζήτησης μπορεί να πραγματοποιηθεί τόσο μέσω του αντίστοιχου κουμπιού όσο και με το πλήκτρο Enter, διατηρώντας ενιαία συμπεριφορά ανεξάρτητα από τον επιλεγμένο τύπο αναζήτησης. Παράλληλα, η διάταξη της κύριας σελίδας προσαρμόστηκε ώστε τα βασικά στοιχεία αναζήτησης και περιήγησης να παραμένουν άμεσα προσβάσιμα σε τυπική προβολή επιτραπέζιου υπολογιστή.

Εικόνα 26: Κύρια σελίδα προβολής και αναζήτησης της βιβλιοθήκης security patterns.

3.7.9 Συνολική ροή υλοποίησης

Η τελική υλοποίηση συνδέει τη γραφική διεπαφή και την OWL βάση γνώσης χωρίς να εκθέτει στον απλό χρήστη τις τεχνικές λεπτομέρειες RDF και SPARQL. Οι ενέργειες του χρήστη μετατρέπονται αρχικά σε αντικείμενα του επιχειρησιακού μοντέλου και στη συνέχεια σε Class, Data Property και Object Property assertions. Αντίστροφα, κατά την ανάγνωση, τα SPARQL αποτελέσματα μετασχηματίζονται σε δομημένα αντικείμενα που μπορούν να παρουσιαστούν από τα Razor views.

Η προσέγγιση αυτή επιτρέπει το ίδιο σημασιολογικό μοντέλο να χρησιμοποιείται με δύο διαφορετικούς τρόπους: μέσω συμβατικών web φορμών για τις συνηθισμένες εργασίες και μέσω

SPARQL για πιο εξειδικευμένη διερεύνηση. Παράλληλα, η αποθήκευση όλων των μεταβολών στο κεντρικό security_patterns.owl και η ενημέρωση της in-memory αναπαράστασης διατηρούν ενιαία την κατάσταση της εφαρμογής και της βάσης γνώσης.

4 *Εγκατάσταση και Επίδειξη της Εφαρμογής*

4.1 Σκοπός και περιβάλλον εκτέλεσης

Στο παρόν κεφάλαιο παρουσιάζονται οι απαραίτητες προϋποθέσεις για την τοπική εκτέλεση της εφαρμογής και επιδεικνύεται η αξιοποίηση των βασικών λειτουργιών της από τη σκοπιά του χρήστη. Σε αντίθεση με το προηγούμενο κεφάλαιο, όπου εξετάστηκαν οι αρχιτεκτονικές αποφάσεις και οι μηχανισμοί υλοποίησης, εδώ δίνεται έμφαση στα βήματα εκκίνησης, στην περιήγηση στη βιβλιοθήκη μοτίβων ασφάλειας και στην αλληλεπίδραση με τη σημασιολογική βάση γνώσης μέσω της διαδικτυακής διεπαφής.

Η εφαρμογή έχει αναπτυχθεί με το ASP.NET Core MVC και στοχεύει στο .NET 8. Η τοπική εκτέλεσή της απαιτεί εγκατεστημένο .NET 8 SDK, πρόσβαση στον πηγαίο κώδικα της λύσης και το αρχείο της οντολογίας security_patterns.owl. Για την εκτέλεση των σχετικών εντολών μπορεί να χρησιμοποιηθεί τερματικό, ενώ η διεπαφή προσπελαύνεται από ένα σύγχρονο πρόγραμμα περιήγησης. Δεν απαιτείται χωριστή εγκατάσταση σχεσιακής βάσης δεδομένων για το συγκεκριμένο πρωτότυπο, καθώς η κύρια αποθηκευμένη σημασιολογική γνώση βρίσκεται στο αρχείο OWL.

4.2 Προετοιμασία και εκκίνηση

Η λύση περιλαμβάνει τα έργα SecurityPatterns.UI, SecurityPatterns.BusinessLogicLayer και SecurityPatterns.DataAccessLayer, καθώς και το αρχείο security_patterns.owl στον ριζικό κατάλογό της. Στο αρχείο SecurityPatterns.UI/appsettings.json η παράμετρος OwlFilePath έχει οριστεί ως ../security_patterns.owl, ώστε το επίπεδο παρουσίασης να χρησιμοποιεί το κοινό αρχείο της οντολογίας. Η διαδρομή ερμηνεύεται με βάση τον κατάλογο περιεχομένου της εφαρμογής, όπως αναλύθηκε στην ενότητα αρχικοποίησης της υλοποίησης.

Η μεταγλώττιση και η εκκίνηση μπορούν να πραγματοποιηθούν από τον ριζικό κατάλογο της λύσης με τις ακόλουθες εντολές:

```
dotnet --list-sdks
dotnet build
dotnet run --project SecurityPatterns.UI
```

Η πρώτη εντολή επιτρέπει την επιβεβαίωση της διαθεσιμότητας του SDK, η δεύτερη μεταγλωττίζει τη λύση και η τρίτη εκκινεί τη διαδικτυακή εφαρμογή. Το προφίλ τοπικής εκτέλεσης της εφαρμογής προβλέπει, μεταξύ άλλων, τη διεύθυνση http://localhost:5297, συνεπώς η κύρια βιβλιοθήκη μπορεί να προσπελαστεί στη διαδρομή http://localhost:5297/SecurityPatterns, εφόσον ενεργοποιηθεί το αντίστοιχο προφίλ. Σε διαφορετική περίπτωση χρησιμοποιείται η πραγματική διεύθυνση που εμφανίζεται στο τερματικό μετά την εκκίνηση. Εναλλακτικά, το βοηθητικό αρχείο .scripts/abacadaba.ps1 εκτελεί διαδοχικά τη μεταγλώττιση και, εφόσον αυτή επιτύχει, την εκκίνηση του έργου UI.

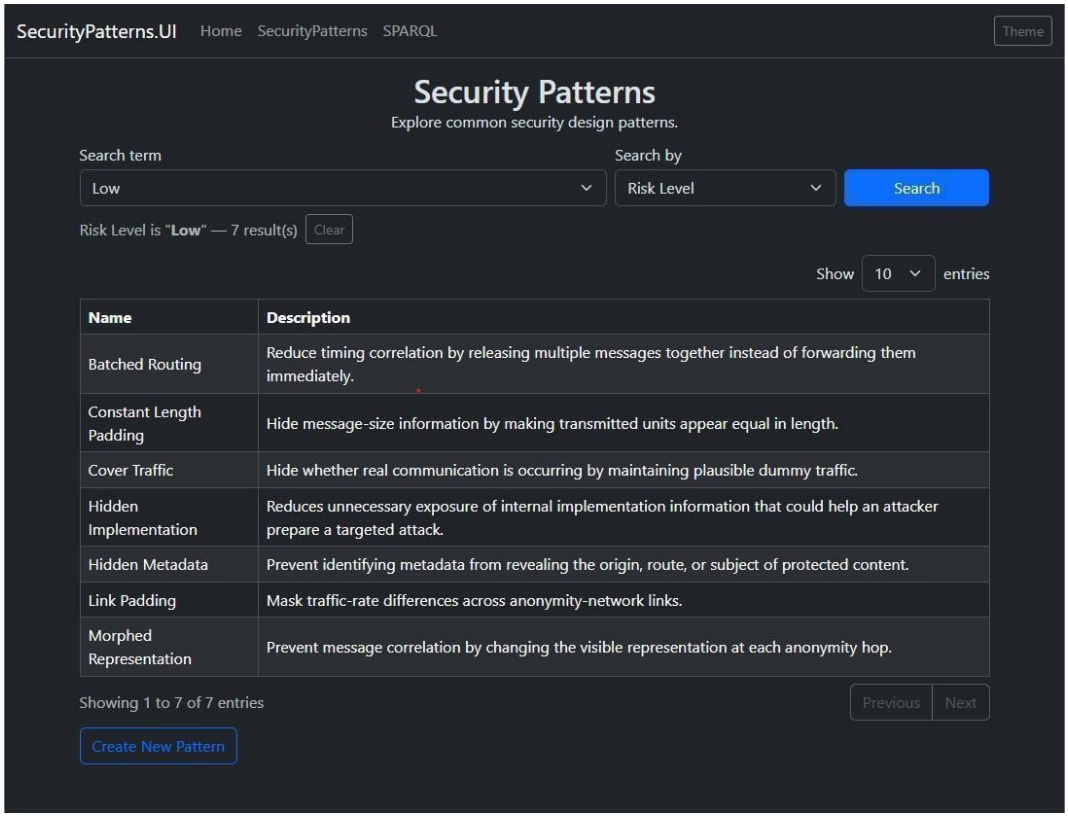
Κατά την αρχικοποίηση φορτώνεται η οντολογία και προετοιμάζονται ορισμένα δεδομένα τα οποία χρησιμοποιούνται συχνά από τις λειτουργίες προβολής και αναζήτησης. Μετά την εκκίνηση ο χρήστης αλληλεπιδρά με την αποθηκευμένη γνώση μέσω της διαδικτυακής διεπαφής, χωρίς να απαιτείται άμεση επεξεργασία του αρχείου OWL.

4.3 Περιήγηση στη βιβλιοθήκη και αναζήτηση μοτίβων

Η κύρια σελίδα SecurityPatterns παρουσιάζει τα ενεργά μοτίβα ασφάλειας σε πίνακα με βασικές πληροφορίες, όπως το όνομα και η περιγραφή τους. Παρέχεται επίσης τμηματική προβολή και πλοήγηση στα αποτελέσματα. Η επιλογή ενός μοτίβου οδηγεί στην αναλυτική σελίδα του, ενώ η δημιουργία νέας καταχώρισης είναι προσβάσιμη από την ίδια διεπαφή. Η βασική οθόνη της βιβλιοθήκης έχει ήδη παρουσιαστεί στην ενότητα υλοποίησης και δεν επαναλαμβάνεται εδώ ως δεύτερη εικόνα.

Για την αναζήτηση ο χρήστης επιλέγει ένα από τα έξι διαθέσιμα κριτήρια: Name, Description, Development Phase, Risk Level, Pattern Class ή Related Pattern. Στις αναζητήσεις ονόματος και περιγραφής εισάγεται όρος κειμένου. Για τα υπόλοιπα κριτήρια, οι επιλογές προκύπτουν από τις αντίστοιχες οντολογικές οντότητες, επιτρέποντας αναζήτηση με βάση αποθηκευμένα σημασιολογικά χαρακτηριστικά και σχέσεις. Η εκτέλεση ενεργοποιείται με το πλήκτρο Enter ή με το κουμπί Search.

Η λειτουργία μπορεί να επιδειχθεί, αρχικά, με αναζήτηση ενός όρου όπως SQL βάσει Name. Στη συνέχεια ο χρήστης μπορεί να επιλέξει Risk Level και να διαλέξει ένα από τα διαθέσιμα επίπεδα, ώστε τα αποτελέσματα να προκύπτουν από τη σχετική σημασιολογική σύνδεση του μοτίβου με το επίπεδο κινδύνου. Η διαδικασία αυτή αναδεικνύει τη διαφορά μεταξύ αναζήτησης ελεύθερου κειμένου και αναζήτησης πάνω σε δομημένη οντολογική πληροφορία.



Εικόνα 27: Στιγμιότυπο πραγματικών αποτελεσμάτων αναζήτησης με Risk Level

Ένα δεύτερο ενδεικτικό σενάριο αφορά την επιλογή του κριτηρίου Related Pattern. Ο χρήστης επιλέγει ένα υφιστάμενο μοτίβο από τις διαθέσιμες τιμές και η εφαρμογή επιστρέφει τα μοτίβα που αντιστοιχούν στη συγκεκριμένη σχέση, όπως αυτή είναι καταγεγραμμένη στη βάση γνώσης. Με αυτόν τον τρόπο αναδεικνύεται ότι οι σχέσεις μεταξύ μοτίβων δεν λειτουργούν αποκλειστικά ως περιγραφικές αναφορές, αλλά μπορούν να αποτελέσουν κριτήριο ανακάλυψης σχετικού περιεχομένου.

Εικόνα 28: Στιγμιότυπο πραγματικών αποτελεσμάτων αναζήτησης με Related Pattern

4.4 Αναλυτική προβολή σημασιολογικής περιγραφής

Με την επιλογή ενός μοτίβου εμφανίζεται η σελίδα αναλυτικών στοιχείων, η οποία συγκεντρώνει τη βασική περιγραφή, το πρόβλημα ασφάλειας, την προτεινόμενη λύση, τις συνέπειες εφαρμογής και, όπου είναι διαθέσιμα, παραδείγματα υλοποίησης. Παρουσιάζονται επίσης πληροφορίες ταξινόμησης και σχέσεις με απειλές, επίπεδα κινδύνου, προστατευόμενα αγαθά, ιδιότητες ασφάλειας, φάσεις ανάπτυξης, άλλα μοτίβα και βιβλιογραφικές πηγές. Η παρουσίαση αυτή επιτρέπει στον χρήστη να εξετάσει τη σημασιολογική περιγραφή ως ενιαίο σύνολο, χωρίς να χρειάζεται να αναζητά επιμέρους RDF δηλώσεις.

Η αντίστοιχη διεπαφή απεικονίζεται ήδη στην ενότητα Μετατροπή των σημασιολογικών αποτελεσμάτων στο μοντέλο της εφαρμογής του Κεφαλαίου 3. Η αναφορά της εδώ εξυπηρετεί την περιγραφή του σεναρίου χρήσης και όχι την επανάληψη της τεχνικής υλοποίησης.

4.5 Δημιουργία, επεξεργασία και λογική διαγραφή

Η λειτουργία Create New Pattern επιτρέπει την εισαγωγή περιγραφικών στοιχείων ενός νέου μοτίβου και την επιλογή κατηγοριών ή σχετικών οντολογικών οντοτήτων μέσω των διαθέσιμων πεδίων της φόρμας. Η υποβολή της καταχώρισης οδηγεί στη δημιουργία νέου Named Individual και στην αποθήκευση των αντίστοιχων σημασιολογικών δηλώσεων, σύμφωνα με τους κανόνες της εφαρμογής. Η φόρμα έχει ήδη παρουσιαστεί εικονογραφικά στην ενότητα δημιουργίας και επεξεργασίας του Κεφαλαίου 3.

Αντίστοιχα, η λειτουργία Edit επιτρέπει τη μεταβολή υπάρχοντος μοτίβου, ενώ η επιλογή Delete ακολουθεί διαδικασία επιβεβαίωσης πριν από τη λογική διαγραφή. Στη δεύτερη περίπτωση η οντότητα διατηρείται στην OWL βάση γνώσης, αλλά εξαιρείται από τις συνήθεις λειτουργίες ανάκτησης ενεργών μοτίβων. Οι διαδικασίες αυτές αξιοποιούν τον ίδιο σημασιολογικό μηχανισμό διαχείρισης που παρουσιάστηκε στην Υλοποίηση. Η επίδειξή τους πρέπει να πραγματοποιείται σε αντίγραφο της οντολογίας ή με κατάλληλη λήψη αντιγράφου ασφαλείας, ώστε να μη μεταβληθούν ανεξέλεγκτα τα δεδομένα της τελικής εργασίας.

4.6 Εξειδικευμένη διερεύνηση μέσω SPARQL

Εκτός από την αναζήτηση μέσω των διαθέσιμων πεδίων, η εφαρμογή προσφέρει ξεχωριστή διεπαφή SPARQL για χρήστες που γνωρίζουν τη γλώσσα επερωτήσεων. Μέσω της συγκεκριμένης σελίδας είναι δυνατή η εκτέλεση read-only επερωτήσεων SELECT πάνω στον φορτωμένο RDF γράφο της οντολογίας. Η δυνατότητα αυτή επιτρέπει τη διατύπωση ερωτημάτων που συνδυάζουν σημασιολογικές σχέσεις χωρίς να απαιτείται τροποποίηση του κώδικα ή του αρχείου της οντολογίας.

Για παράδειγμα, η ακόλουθη επερώτηση ανακτά ονόματα ενεργών μοτίβων που περιλαμβάνουν τον όρο sql. Χρησιμοποιείται το rdfs:subClassOf* ώστε να συμπεριλαμβάνονται στιγμιότυπα υποκλάσεων της SecurityPattern, ανεξάρτητα από το βάθος της ιεραρχίας.

```
PREFIX sp: <http://example.org/security-patterns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT DISTINCT ?name
WHERE {
  ?pattern a ?type .
  ?type rdfs:subClassOf* sp:SecurityPattern .
  ?pattern rdfs:label ?name .
  OPTIONAL { ?pattern sp:isDeleted ?isDeleted . }
  FILTER(!BOUND(?isDeleted) || LCASE(STR(?isDeleted)) != "true")
  FILTER(CONTAINS(LCASE(STR(?name)), "sql"))
}
LIMIT 10
```

Η διεπαφή επιστρέφει τις δεσμεύσεις των μεταβλητών σε πίνακα, επιτρέποντας στον χρήστη να εξετάσει τα αποτελέσματα. Η εκτέλεση SPARQL στην εφαρμογή είναι ελεγχόμενη: επιτρέπονται επερωτήσεις SELECT, ενώ απαγορεύονται ρήτρες πρόσβασης σε εξωτερικές πηγές μέσω SERVICE, FROM και FROM NAMED και εφαρμόζεται μέγιστο όριο 500 γραμμών αποτελεσμάτων. Η λειτουργία αυτή παρουσιάζεται ήδη σε σχετικό στιγμιότυπο οθόνης της ενότητας Υλοποίησης.

4.7 Σύνοψη της επίδειξης

Τα παραπάνω σενάρια δείχνουν τους δύο συμπληρωματικούς τρόπους αξιοποίησης της σημασιολογικής βάσης γνώσης: αφενός τη χρήση προκαθορισμένων λειτουργιών προβολής και αναζήτησης, χωρίς απαίτηση γνώσης της OWL ή της SPARQL, και αφετέρου τη δυνατότητα άμεσης διατύπωσης ερωτημάτων από εξειδικευμένους χρήστες. Η λειτουργική επίδειξη τεκμηριώνει την αντιστοίχιση των διαθέσιμων διεπαφών με τις λειτουργίες του λογισμικού. Δεν αποτελεί από μόνη της ποσοτική αξιολόγηση απόδοσης, μελέτη ευχρηστίας με συμμετέχοντες ή έλεγχο επιστημονικής εγκυρότητας του περιεχομένου της οντολογίας. Τα πλεονεκτήματα, οι περιορισμοί και η αποτίμηση της προτεινόμενης λύσης εξετάζονται στο επόμενο κεφάλαιο.

5 Αποτίμηση Προτεινόμενης Λύσης

Η προτεινόμενη λύση αξιολογείται ως λειτουργικό σημασιολογικό πλαίσιο για την αναπαράσταση και διαχείριση μοτίβων ασφάλειας. Ο βασικός στόχος της εργασίας επιτυγχάνεται, καθώς η οντολογική αναπαράσταση δεν παραμένει απομονωμένη σε ένα OWL αρχείο, αλλά αξιοποιείται από εφαρμογή που επιτρέπει την προβολή, αναζήτηση, δημιουργία, επεξεργασία και λογική διαγραφή security patterns.

Ιδιαίτερη σημασία έχει ότι η τελική λύση διατηρεί τη σημασιολογική δομή της γνώσης. Οι κατηγορίες των patterns αναπαρίστανται μέσω της ιεραρχίας κλάσεων, ενώ έννοιες όπως οι απειλές, τα προστατευόμενα αγαθά, οι ιδιότητες ασφάλειας, οι φάσεις ανάπτυξης και οι βιβλιογραφικές πηγές συνδέονται μέσω Object Properties. Έτσι, η εφαρμογή δεν λειτουργεί ως απλός κατάλογος κειμενικών εγγραφών, αλλά ως διεπαφή προς μια δομημένη βάση γνώσης.

Παράλληλα, επιτυγχάνεται διαχωρισμός μεταξύ της συνήθους χρήσης και της εξειδικευμένης σημασιολογικής διερεύνησης. Οι βασικές λειτουργίες είναι προσβάσιμες μέσω της web διεπαφής, ενώ η ξεχωριστή read-only SPARQL διεπαφή επιτρέπει την εκτέλεση περισσότερο σύνθετων ερωτημάτων χωρίς να εκθέτει τη βάση γνώσης σε αυθαίρετες μεταβολές. Η τελική υλοποίηση περιορίζει τις user-defined επερωτήσεις στην οικογένεια SELECT, απαγορεύει SERVICE, FROM και FROM NAMED και εφαρμόζει μέγιστο όριο αποτελεσμάτων.

Η οντολογία αξιολογήθηκε ανεξάρτητα από τη λειτουργία της εφαρμογής μέσω του Protégé και του HermiT reasoner. Με τον τρόπο αυτό διαχωρίστηκε ο έλεγχος της λογικής συνέπειας του σημασιολογικού μοντέλου από τον έλεγχο της ορθής λειτουργίας του λογισμικού. Η διάκριση αυτή αποτελεί σημαντικό στοιχείο της συνολικής αποτίμησης, καθώς η επιτυχής λειτουργία της εφαρμογής από μόνη της δεν θα ήταν επαρκής για την αξιολόγηση της οντολογίας.

5.1 Ικανοποίηση Απαιτήσεων

Η τελική εφαρμογή ικανοποιεί το μεγαλύτερο μέρος των λειτουργικών και μη λειτουργικών απαιτήσεων που τέθηκαν κατά την ανάλυση. Η αντιστοίχιση των βασικών απαιτήσεων με την τελική υλοποίηση παρουσιάζεται συνοπτικά στον Πίνακα 4.

Πίνακας 4: Αποτίμηση της ικανοποίησης των βασικών απαιτήσεων της εφαρμογής

Απαίτηση	**Κατάσταση**	**Τρόπος ικανοποίησης**
Προβολή βιβλιοθήκης patterns	Ικανοποιείται	Παρουσίαση ενεργών patterns μέσω της κύριας σελίδας
Αναζήτηση μοτίβων	Ικανοποιείται	Πολυκριτηριακή αναζήτηση βάσει Name, Description, Development Phase, Risk Level, Pattern Class και Related Pattern, με αξιοποίηση περιγραφικών και σημασιολογικών χαρακτηριστικών της οντολογίας.
Αναλυτική προβολή	Ικανοποιείται	Παρουσίαση περιγραφικών και σημασιολογικών χαρακτηριστικών
Δημιουργία pattern	Ικανοποιείται	Δημιουργία Named Individual με Class, Data και Object Property Assertions
Επεξεργασία pattern	Ικανοποιείται	Τροποποίηση περιγραφικών και σημασιολογικών σχέσεων
Λογική διαγραφή	Ικανοποιείται	Χρήση isDeleted=true χωρίς φυσική αφαίρεση του individual
Σημασιολογικές επερωτήσεις	Ικανοποιείται	Read-only SPARQL SELECT interface
Βιβλιογραφική τεκμηρίωση	Ικανοποιείται	Σύνδεση patterns με BibliographicSource individuals
Σημασιολογική συνέπεια	Ικανοποιείται σε σημαντικό βαθμό	Επιλογή υφιστάμενων ontology entities και έλεγχος της οντολογίας με reasoner
Συντηρησιμότητα	Ικανοποιείται	Διαχωρισμός UI, Business Logic και Data Access layers
Επεκτασιμότητα	Ικανοποιείται σε σημαντικό βαθμό	Δυναμική ανάκτηση κλάσεων και individuals από την ontology
Αυθεντικοποίηση / ρόλοι χρηστών	Δεν υλοποιείται	Όλοι οι χρήστες της τρέχουσας έκδοσης έχουν τις ίδιες δυνατότητες
Πλήρες versioning	Δεν υλοποιείται	Διατηρείται μόνο βασικό ιστορικό μέσω comments και soft delete

Η τελική έκδοση της εφαρμογής ενισχύει ιδιαίτερα τη σύνδεση μεταξύ της σημασιολογικής μοντελοποίησης και της πρακτικής αναζήτησης. Χαρακτηριστικά όπως το επίπεδο κινδύνου, η φάση ανάπτυξης, η κλάση και οι σχέσεις μεταξύ μοτίβων δεν χρησιμοποιούνται μόνο για την περιγραφή των δεδομένων στην οντολογία, αλλά αξιοποιούνται απευθείας ως κριτήρια ανάκτησης μέσω της γραφικής διεπαφής. Με τον τρόπο αυτό η σημασιολογική πληροφορία μετατρέπεται από στοιχείο εσωτερικής αναπαράστασης σε λειτουργικό μηχανισμό ανακάλυψης κατάλληλων security patterns.

Οι βασικές λειτουργικές απαιτήσεις καλύπτονται πλήρως από την τελική εφαρμογή. Ιδιαίτερα σημαντική είναι η αναβάθμιση της δημιουργίας και επεξεργασίας των patterns, καθώς η τελική ροή δεν περιορίζεται πλέον στα βασικά πεδία, αλλά υποστηρίζει το εμπλουτισμένο σημασιολογικό μοντέλο μέσω του SemanticPatternInput.

Στις μη λειτουργικές απαιτήσεις, η συντηρησιμότητα και η επεκτασιμότητα υποστηρίζονται κυρίως από την πολυεπίπεδη αρχιτεκτονική και τη δυναμική ανάκτηση δεδομένων από την οντολογία. Αντίθετα, η απουσία authentication και role-based authorization παραμένει περιορισμός της παρούσας έκδοσης. Αντίστοιχα, η διατήρηση προηγούμενων τιμών και η λογική διαγραφή προσφέρουν βασική ιχνηλασιμότητα, αλλά δεν αποτελούν ολοκληρωμένο μηχανισμό versioning.

5.2 Σύγκριση με Σχετικές Ερευνητικές Εργασίες

Η προτεινόμενη λύση κινείται στην ίδια γενική κατεύθυνση με προηγούμενες οντολογικές προσεγγίσεις για τα security patterns, αλλά εστιάζει ιδιαίτερα στη σύνδεση του σημασιολογικού μοντέλου με μια λειτουργική εφαρμογή διαχείρισης.

Οι Guan, Yang και Wang (2016) παρουσιάζουν μια ontology-based προσέγγιση για την επιλογή security patterns, χρησιμοποιώντας OWL ώστε απαιτήσεις ασφάλειας να μπορούν να συνδεθούν με κατάλληλες λύσεις. Η παρούσα εργασία συμμερίζεται την ιδέα της σημασιολογικής αναπαράστασης, αλλά εστιάζει περισσότερο στη συνολική διαχείριση της βάσης γνώσης και όχι αποκλειστικά στην επιλογή pattern.

Αντίστοιχα, οι Pereira Vale και Fernández (2019) προτείνουν οντολογία για την περιγραφή security patterns, ενσωματώνοντας στοιχεία όπως το πρόβλημα, το πλαίσιο, η λύση, το στάδιο του κύκλου ζωής και το αρχιτεκτονικό επίπεδο. Η παρούσα εργασία ακολουθεί παρόμοια σημασιολογική κατεύθυνση, επεκτείνοντάς την στο πλαίσιο της συγκεκριμένης υλοποίησης με σχέσεις προς threats, protected assets, security properties, άλλα patterns και bibliographic sources, καθώς και με web διεπαφή για τη διαχείριση των αντίστοιχων individuals.

Η συστηματική μελέτη των Jafari και Rasoolzadegan (2018) αναδεικνύει την ποικιλία των security patterns και των τρόπων ταξινόμησής τους. Η ανάγκη αυτή αντιμετωπίζεται στην παρούσα εργασία μέσω μιας ιεραρχικής και πολλαπλής ταξινόμησης, όπου ένα pattern μπορεί να ανήκει σε περισσότερες από μία υποκλάσεις της SecurityPattern.

Συνεπώς, η κύρια διαφοροποίηση της προτεινόμενης λύσης δεν βρίσκεται απλώς στη χρήση μιας OWL οντολογίας, η οποία έχει ήδη εξεταστεί στη σχετική βιβλιογραφία, αλλά στη δημιουργία μιας ολοκληρωμένης ροής από τη σημασιολογική μοντελοποίηση και επικύρωση έως την πρακτική διαχείριση και διερεύνηση της γνώσης μέσω web εφαρμογής.

5.3 Πλεονεκτήματα και Περιορισμοί της Προτεινόμενης Λύσης

Η υλοποίηση έδειξε ότι μια οντολογική βάση γνώσης μπορεί να αξιοποιηθεί ως λειτουργικός πυρήνας μιας εφαρμογής διαχείρισης μοτίβων ασφάλειας. Τα security patterns δεν αποθηκεύονται ως ανεξάρτητες περιγραφές κειμένου, αλλά ως σημασιολογικά οργανωμένες οντότητες, οι οποίες διαθέτουν ταξινομήσεις, περιγραφικά χαρακτηριστικά, σχέσεις με έννοιες του πεδίου ασφάλειας και συνδέσεις με βιβλιογραφικές πηγές.

Βασικό πλεονέκτημα της λύσης είναι η δομημένη αναπαράσταση της γνώσης. Η διάκριση μεταξύ Datatype και Object Properties επιτρέπει την αποτύπωση τόσο της περιγραφής ενός pattern όσο και των πραγματικών σημασιολογικών του σχέσεων με απειλές, προστατευόμενα αγαθά, ιδιότητες ασφάλειας, φάσεις ανάπτυξης και άλλα patterns. Παράλληλα, η ιεραρχική ταξινόμηση μέσω rdf:type και rdfs:subClassOf επιτρέπει την οργάνωση των patterns χωρίς να περιορίζεται κάθε οντότητα σε μία απλή τιμή κατηγορίας.

Επιπλέον, η εφαρμογή αποκρύπτει από τον τελικό χρήστη την πολυπλοκότητα των OWL και RDF. Η προβολή, δημιουργία, επεξεργασία και λογική διαγραφή πραγματοποιούνται μέσω συμβατικής web διεπαφής, ενώ οι αντίστοιχες ενέργειες μετατρέπονται εσωτερικά σε Class, Data Property και Object Property Assertions. Για εξειδικευμένη διερεύνηση παρέχεται επιπλέον read-only διεπαφή SPARQL, με περιορισμούς ώστε να επιτρέπονται μόνο ασφαλείς επερωτήσεις SELECT.

Θετικό στοιχείο αποτελεί επίσης ο διαχωρισμός της εφαρμογής σε επίπεδα παρουσίασης, επιχειρησιακής λογικής και πρόσβασης δεδομένων. Η αρχιτεκτονική αυτή περιορίζει την εξάρτηση της διεπαφής από τις λεπτομέρειες της οντολογίας και καθιστά ευκολότερη τη συντήρηση και την επέκταση της εφαρμογής. Αντίστοιχα, η χρήση λογικής διαγραφής διατηρεί την υπάρχουσα γνώση στην οντολογία και αποφεύγει την οριστική απώλεια ενός pattern.

Η λύση παρουσιάζει, ωστόσο, και περιορισμούς. Η χρήση ενός τοπικού OWL αρχείου ως κύριου μηχανισμού μόνιμης αποθήκευσης είναι κατάλληλη για το πλαίσιο της παρούσας εργασίας, αλλά δεν αποτελεί ιδανική επιλογή για περιβάλλον με πολλούς ταυτόχρονους χρήστες ή αυξημένες απαιτήσεις κλιμάκωσης. Επιπλέον, η εφαρμογή δεν διαθέτει αυθεντικοποίηση και διαφοροποίηση δικαιωμάτων χρηστών, ενώ η καταγραφή προηγούμενων τιμών και η λογική διαγραφή παρέχουν μόνο βασική ιχνηλασιμότητα και όχι πλήρες σύστημα versioning.

Τέλος, αν και η οντολογία έχει ελεγχθεί ανεξάρτητα στο Protégé με reasoner, η ίδια η web εφαρμογή δεν εκτελεί OWL reasoning ή SHACL validation κατά τη λειτουργία της. Η ποιότητα των αποτελεσμάτων εξαρτάται επίσης από την πληρότητα και την ορθότητα της γνώσης που έχει ενσωματωθεί στην οντολογία. Οι περιορισμοί αυτοί δεν αναιρούν τη λειτουργικότητα της λύσης, αλλά καθορίζουν με σαφήνεια το πεδίο εφαρμογής της και τις κατευθύνσεις στις οποίες μπορεί να επεκταθεί.

6 *Συμπεράσματα και Μελλοντική Εργασία*

6.1 Συμπεράσματα

Η εκπόνηση της παρούσας εργασίας ανέδειξε ότι η αξία των security patterns δεν περιορίζεται στην καταγραφή μεμονωμένων λύσεων για γνωστά προβλήματα ασφάλειας. Η πραγματική χρησιμότητά τους αυξάνεται όταν οργανώνονται με τρόπο που επιτρέπει στον χρήστη να αντιλαμβάνεται όχι μόνο ποια λύση προτείνεται, αλλά και σε ποιο πλαίσιο, για ποια απειλή, ποιο αγαθό προστατεύει, ποιες ιδιότητες ασφάλειας υποστηρίζει και πώς σχετίζεται με άλλες διαθέσιμες λύσεις.

Μέσα από τη διαδικασία της εργασίας έγινε επίσης εμφανές ότι η οργάνωση των security patterns δεν είναι ένα απλό πρόβλημα κατηγοριοποίησης. Ένα μοτίβο μπορεί να ανήκει ταυτόχρονα σε περισσότερες κατηγορίες, να αντιμετωπίζει περισσότερες απειλές και να σχετίζεται με διαφορετικές φάσεις ανάπτυξης ή άλλα patterns. Η σημασιολογική αναπαράσταση επιτρέπει αυτές οι σχέσεις να εκφράζονται ρητά, αντί να χάνονται μέσα σε ελεύθερο κείμενο ή σε μία μοναδική ετικέτα ταξινόμησης.

Ένα ακόμη σημαντικό συμπέρασμα αφορά την προέλευση της γνώσης. Κατά τη μελέτη και οργάνωση των patterns διαπιστώθηκε ότι διαφορετικές πηγές μπορεί να χρησιμοποιούν διαφορετικές ονομασίες, διαφορετικό επίπεδο λεπτομέρειας ή διαφορετική ταξινόμηση για συγγενείς έννοιες. Για τον λόγο αυτό η βιβλιογραφική τεκμηρίωση και η δυνατότητα διατήρησης της προέλευσης μιας περιγραφής αποτελούν ουσιώδες μέρος μιας αξιόπιστης βάσης γνώσης και όχι απλώς συμπληρωματική πληροφορία.

Από την πλευρά του τελικού χρήστη, η εργασία δείχνει ότι η σημασιολογική τεχνολογία μπορεί να αξιοποιηθεί χωρίς να απαιτείται άμεση γνώση OWL, RDF ή SPARQL. Το σημαντικότερο αποτέλεσμα δεν είναι επομένως μόνο η κατασκευή μιας οντολογίας ή μιας εφαρμογής, αλλά η δημιουργία μιας γέφυρας μεταξύ της θεωρητικής γνώσης γύρω από τα security patterns και της πρακτικής αξιοποίησής της κατά την αναζήτηση και μελέτη λύσεων ασφάλειας.

Συνολικά, η εργασία υποστηρίζει την άποψη ότι τα security patterns μπορούν να αντιμετωπιστούν αποτελεσματικότερα ως συνδεδεμένη γνώση και όχι ως απομονωμένα τεχνικά κείμενα. Η προσέγγιση αυτή δεν επιλύει αυτομάτως το πρόβλημα της επιλογής του κατάλληλου

pattern, αλλά παρέχει μια περισσότερο οργανωμένη βάση πάνω στην οποία μπορούν να αναπτυχθούν καλύτεροι μηχανισμοί αναζήτησης, σύγκρισης και υποστήριξης αποφάσεων.

6.2 Μελλοντική Εργασία

Σε επίπεδο εφαρμογής, η επόμενη φυσική εξέλιξη είναι η αξιοποίηση της ήδη διαθέσιμης σημασιολογικής πληροφορίας για καθοδηγούμενη αναζήτηση και επιλογή patterns. Αντί ο χρήστης να χρειάζεται να γνωρίζει το όνομα ενός pattern ή να συντάσσει SPARQL, θα μπορούσε να δηλώνει χαρακτηριστικά του προβλήματός του, όπως την απειλή που αντιμετωπίζει, το προστατευόμενο αγαθό, την επιθυμητή ιδιότητα ασφάλειας ή τη φάση ανάπτυξης, και η εφαρμογή να περιορίζει ή να κατατάσσει τα κατάλληλα patterns.

Η εφαρμογή θα μπορούσε επίσης να εξελιχθεί σε περισσότερο ολοκληρωμένο σύστημα συνεργατικής διαχείρισης γνώσης. Η προσθήκη αυθεντικοποίησης και ρόλων χρηστών, πλήρους ιστορικού αλλαγών, εκδόσεων των patterns και διαδικασίας έγκρισης νέου περιεχομένου θα επέτρεπε τη χρήση της από μεγαλύτερες ομάδες. Αντίστοιχα, η μετάβαση από το τοπικό OWL αρχείο σε RDF triple store θα επέτρεπε καλύτερη διαχείριση μεγαλύτερου όγκου δεδομένων και ταυτόχρονων χρηστών.

Μία ακόμη κατεύθυνση αφορά την ενσωμάτωση reasoning και validation στην ίδια την εφαρμογή. Η αυτόματη αξιοποίηση inferred γνώσης, καθώς και η εφαρμογή κανόνων εγκυρότητας πριν από την αποθήκευση νέων patterns, θα μπορούσαν να ενισχύσουν τη συνέπεια της βάσης γνώσης και να βοηθούν τον χρήστη στον εντοπισμό ελλιπών ή αντιφατικών περιγραφών.

Σε επίπεδο επιστημονικής έρευνας, παραμένει ανοικτό το ζήτημα της δημιουργίας ενός περισσότερο κοινού και αποδεκτού μοντέλου περιγραφής και ταξινόμησης των security patterns. Η βιβλιογραφία έχει καταγράψει σημαντικό αριθμό patterns και διαφορετικών σχημάτων ταξινόμησης, χωρίς όμως να υπάρχει ένα ενιαίο σχήμα που να χρησιμοποιείται καθολικά (Jafari & Rasoolzadegan, 2020). Μελλοντικές εργασίες θα μπορούσαν να εξετάσουν τη σύγκλιση διαφορετικών καταλόγων και οντολογιών και τη δημιουργία κοινών λεξιλογίων για έννοιες όπως απειλές, assets, security properties και lifecycle phases.

Ιδιαίτερο ενδιαφέρον παρουσιάζει επίσης η εμπειρική αξιολόγηση της αποτελεσματικότητας των ίδιων των security patterns. Πέρα από την περιγραφή του τι προτείνει ένα pattern, απαιτούνται περισσότερες μελέτες που να εξετάζουν πότε μια λύση είναι πραγματικά αποτελεσματική, ποιες προϋποθέσεις πρέπει να ισχύουν για την εφαρμογή της και ποιοι συμβιβασμοί προκύπτουν στην πράξη. Τέτοιες εργασίες θα μπορούσαν να οδηγήσουν σε πλουσιότερες βάσεις γνώσης, στις οποίες η επιλογή ενός pattern θα βασίζεται όχι μόνο στη θεματική του συνάφεια αλλά και σε εμπειρικά δεδομένα για τη χρήση του.

Τέλος, μελλοντικές έρευνες και συγγράμματα μπορούν να εξετάσουν περισσότερο συστηματικά τη θέση των security patterns μέσα στον σύγχρονο κύκλο ανάπτυξης λογισμικού. Η σύνδεσή τους με threat modelling, security requirements, αρχιτεκτονικές αποφάσεις, secure

development practices και πραγματικά περιστατικά ασφάλειας θα μπορούσε να μετατρέψει τους καταλόγους patterns από στατικά αποθετήρια λύσεων σε ενεργά εργαλεία υποστήριξης της ασφαλούς ανάπτυξης.

Βιβλιογραφία

Alvi, A. K., & Zulkernine, M. (2012). A comparative study of software security pattern classifications. In 2012 Seventh International Conference on Availability, Reliability and Security (ARES) (pp. 582–589). IEEE. doi:10.1109/ARES.2012.43.

Alvi, A. K., & Zulkernine, M. (2021). A security pattern detection framework for building more secure software. Journal of Systems and Software, 171, 110838. doi:10.1016/j.jss.2020.110838.

Atkinson, C., & Kühne, T. (2002). Rearchitecting the UML infrastructure. ACM Transactions on Modeling and Computer Simulation, 12(4), 290–321.

Atkinson, C., & Kühne, T. (2003). Model-driven development: A metamodeling foundation. IEEE Software, 20(5), 36–41.

Baclawski, K., Kokar, M. M., Kogut, P. A., Hart, L., Smith, J. E., Holmes III, W. S., Letkowski, J., & Aronson, M. L. (2001). Extending UML to support ontology engineering for the Semantic Web. In UML 2001 – The Unified Modeling Language (LNCS 2185, pp. 342–360). Springer.

Beckett, D., & McBride, B. (Eds.). (2004). RDF/XML Syntax Specification (Revised). W3C Recommendation.

Blakley, B., Heath, C., & The Open Group Security Forum. (2004). Security Design Patterns (Technical Guide G031). The Open Group.

Blanco, C., Lasheras, J., Valencia-García, R., Fernández-Medina, E., Toval, A., & Piattini, M. (2008). A systematic review and comparison of security ontologies. In Third International Conference on Availability, Reliability and Security (ARES 2008) (pp. 813–820). IEEE. doi:10.1109/ARES.2008.33.

Brickley, D., & Guha, R. V. (Eds.). (2014). RDF Schema 1.1. W3C Recommendation.

Calì, A., Calvanese, D., De Giacomo, G., & Lenzerini, M. (2002). A formal framework for reasoning on UML class diagrams. In Foundations of Intelligent Systems: ISMIS 2002 (LNCS 2366, pp. 503–513). Springer. doi:10.1007/3-540-48050-1_54.

Carlson, D. (2001). Modeling XML Applications with UML. Addison-Wesley.

Cyganiak, R., Wood, D., & Lanthaler, M. (Eds.). (2014). RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation.

Denker, G., Kagal, L., & Finin, T. (2005). Security in the Semantic Web using OWL. Information Security Technical Report, 10(1), 51–58. doi:10.1016/j.istr.2004.11.002.

Djurić, D., Gašević, D., Devedžić, V., & Damjanović, V. (2005). A UML profile for OWL ontologies. In European MDA Workshops: Foundations and Applications, MDAFA 2004 (LNCS 3599, pp. 204–219). Springer. doi:10.1007/11538097_14.

Dobson, G., & Sawyer, P. (2006). Revisiting ontology-based requirements engineering in the age of the Semantic Web. In Proceedings of the International Seminar on Dependable Requirements Engineering of Computerised Systems at NPPs, Halden, Norway.

Donner, M. (2003). Toward a security ontology. IEEE Security & Privacy, 1(3), 6–7.

Eclipse Foundation. (n.d.). OCL in Papyrus. Eclipse OCL Documentation. Retrieved September 27, 2026.

Eclipse Foundation. (n.d.). Xtext Documentation. Retrieved September 27, 2026.

Elahi, G., Yu, E., & Zannone, N. (2009). A modeling ontology for integrating vulnerabilities into security requirements conceptual foundations. In Conceptual Modeling – ER 2009 (pp. 99–114). Springer. doi:10.1007/978-3-642-04840-1_10.

Fernandez-Buglioni, E. B. (2013). Security Patterns in Practice: Designing Secure Architectures Using Software Patterns. Wiley.

Fenz, S., & Ekelhart, A. (2009). Formalizing information security knowledge. In Proceedings of the 2009 ACM Symposium on Information, Computer and Communications Security (pp. 183–194). ACM. doi:10.1145/1533057.1533084.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley.

Gašević, D., Djurić, D., Devedžić, V., & Damjanović, V. (2004). Converting UML to OWL ontologies. In Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters (pp. 488–489). ACM. doi:10.1145/1013367.1013539.

Glimm, B., Horrocks, I., Motik, B., Stoilos, G., & Wang, Z. (2014). HermiT: An OWL 2 reasoner. Journal of Automated Reasoning, 53(3), 245–269. doi:10.1007/s10817-014-9305-1.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing. International Journal of Human-Computer Studies, 43(5–6), 907–928. doi:10.1006/ijhc.1995.1081.

Guan, H., Yang, H., & Wang, J. (2016). An ontology-based approach to security pattern selection. International Journal of Automation and Computing, 13(2), 168–182. doi:10.1007/s11633-016-0950-1.

Harris, S., & Seaborne, A. (Eds.). (2013). SPARQL 1.1 Query Language. W3C Recommendation.

Herzog, A., Shahmehri, N., & Duma, C. (2007). An ontology of information security. International Journal of Information Security and Privacy, 1(4), 1–23. doi:10.4018/jisp.2007100101.

Heyman, T., Yskout, K., Scandariato, R., & Joosen, W. (2007). An analysis of the security patterns landscape. In Third International Workshop on Software Engineering for Secure Systems (SESS ’07). IEEE. doi:10.1109/SESS.2007.4.

Ito, Y., Washizaki, H., Yoshizawa, M., Fukazawa, Y., Okubo, T., Kaiya, H., Hazeyama, A., Yoshioka, N., & Fernandez, E. B. (2015). Systematic mapping of security patterns research. In Proceedings of the 22nd Conference on Pattern Languages of Programs (PLoP 2015). Hillside Group.

Jafari, A. J., & Rasoolzadegan, A. (2020). Security patterns: A systematic mapping study. *Journal of Computer Languages, 56*, 100938. https://doi.org/10.1016/j.cola.2019.100938

Jürjens, J. (2003). Developing safety-critical systems with UML. In UML 2003 – The Unified Modeling Language: Modeling Languages and Applications (LNCS 2863, pp. 360–372). Springer. doi:10.1007/978-3-540-45221-8_31.

Khoury, P., Mokhtari, A., Coquery, E., & Hacid, M.-S. (2008). An ontological interface for software developers to select security patterns. In DEXA Workshops 2008 (pp. 297–301). IEEE. doi:10.1109/DEXA.2008.110.

Kienzle, D. M., Elder, M. C., Tyree, D., & Edwards-Hewitt, J. (2002). Security Patterns Repository Version 1.0. DARPA.

Kiko, K., & Atkinson, C. (2008). A Detailed Comparison of UML and OWL (Technical Report TR-2008-004). University of Mannheim.

Konrad, S., Cheng, B. H. C., Campbell, L. A., & Wassermann, R. (2003). Using security patterns to model and analyze security requirements. In IEEE Workshop on Requirements for High Assurance Systems (RHAS ’03) (pp. 13–22).

Laverdière, M.-A., Mourad, A., Hanna, A., & Debbabi, M. (2006). Security design patterns: Survey and evaluation. In Canadian Conference on Electrical and Computer Engineering (CCECE 2006) (pp. 1605–1608). IEEE. doi:10.1109/CCECE.2006.277727.

Maña, A., & Pujol, G. (2008). Towards formal specification of abstract security properties. In Third International Conference on Availability, Reliability and Security (ARES 2008). IEEE. doi:10.1109/ARES.2008.202.

Mavroeidis, V., & Bromander, S. (2017). Cyber Threat Intelligence Model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence. In 2017 European Intelligence and Security Informatics Conference (EISIC) (pp. 91–98). IEEE. doi:10.1109/EISIC.2017.20.

Musen, M. A. (2015). The Protégé project: A look back and a look forward. AI Matters, 1(4), 4–12. doi:10.1145/2757001.2757003.

National Institute of Standards and Technology (NIST). (n.d.). Standard. Computer Security Resource Center Glossary. Retrieved July 20, 2026.

Object Management Group (OMG). (2014a). Object Constraint Language (OCL), Version 2.4.

Object Management Group (OMG). (2014b). Ontology Definition Metamodel (ODM), Version 1.1.

Object Management Group (OMG). (2017). OMG Unified Modeling Language (OMG UML), Version 2.5.1.

Pereira Vale, A., & Fernández, E. B. (2019). An ontology for security patterns. In 2019 38th International Conference of the Chilean Computer Science Society (SCCC) (pp. 1–8). IEEE. doi:10.1109/SCCC49216.2019.8966393.

PricewaterhouseCoopers (PwC). (2015). Information Security Breaches Survey 2015: Technical Report. Department for Business, Innovation & Skills.

Rajmohan, T., Nguyen, P. H., & Ferry, N. (2020). A systematic mapping of patterns and architectures for IoT security. In Proceedings of the 5th International Conference on Internet of Things, Big Data and Security (IoTBDS) (pp. 138–149). SciTePress. doi:10.5220/0009583001380149.

Raskin, V., Hempelmann, C. F., Triezenberg, K. E., & Nirenburg, S. (2001). Ontology in information security: A useful theoretical foundation and methodological tool. In Proceedings of the 2001 New Security Paradigms Workshop (pp. 53–59). ACM. doi:10.1145/508171.508180.

Schumacher, M. (2003). Security Engineering with Patterns: Origins, Theoretical Models, and New Applications. Springer. doi:10.1007/b11930.

Schumacher, M., Fernandez-Buglioni, E., Hybertson, D., Buschmann, F., & Sommerlad, P. (2006). Security Patterns: Integrating Security and Systems Engineering. Wiley.

Souag, A., Salinesi, C., & Comyn-Wattiau, I. (2012). Ontologies for security requirements: A literature survey and classification. In CAiSE Workshops 2012 (pp. 61–69). Springer. doi:10.1007/978-3-642-31069-0_5.

Steel, C., Nagappan, R., & Lai, R. (2005). Core Security Patterns: Best Practices and Strategies for J2EE, Web Services, and Identity Management. Prentice Hall PTR.

Tsoumas, B., & Gritzalis, D. (2006). Towards an ontology-based security management. In 20th International Conference on Advanced Information Networking and Applications (AINA 2006) (pp. 985–992). IEEE. doi:10.1109/AINA.2006.329.

Uzunov, A. V. (2016). A survey of security solutions for distributed publish/subscribe systems. Computers & Security, 61, 94–129. doi:10.1016/j.cose.2016.04.008.

Viega, J., & McGraw, G. (2001). Building Secure Software: How to Avoid Security Problems the Right Way. Addison-Wesley.

W3C OWL Working Group. (2012). OWL 2 Web Ontology Language: Document Overview (2nd ed.). W3C Recommendation.

Washizaki, H., Ogata, S., Hazeyama, A., Okubo, T., Fernandez, E. B., & Yoshioka, N. (2020). Landscape of architecture and design patterns for IoT systems. IEEE Internet of Things Journal, 7(10), 10091–10101. doi:10.1109/JIOT.2020.3003528.

Washizaki, H., Xia, T., Kamata, N., Fukazawa, Y., Kanuka, H., Kato, T., Yoshino, M., Okubo, T., Ogata, S., Kaiya, H., Hazeyama, A., Tanaka, T., Yoshioka, N., & Priyalakshmi, G. (2021). Systematic literature review of security pattern research. Information, 12(1), 36. doi:10.3390/info12010036.

World Wide Web Consortium (W3C). (1998). Extensible Markup Language (XML) 1.0. W3C Recommendation, 10 February 1998.

Yoder, J. W., & Barcalow, J. (1997). Architectural Patterns for Enabling Application Security. In Proceedings of the 4th Conference on Pattern Languages of Programs (PLoP ’97). Technical Report WUCS-97-34, Washington University.

Yoshioka, N., Washizaki, H., & Maruyama, K. (2008). A survey on security patterns. Progress in Informatics, 5, 35–47. doi:10.2201/NiiPi.2008.5.5.

Παράρτημα Ι - Επιλεγμένα Αποσπάσματα Κώδικα

Στο παρόν παράρτημα παρατίθενται επιλεγμένα αποσπάσματα κώδικα από την τελική υλοποίηση της εφαρμογής. Τα αποσπάσματα επιλέχθηκαν επειδή αποτυπώνουν βασικές τεχνικές πτυχές της προτεινόμενης λύσης, όπως την αρχικοποίηση και τη σύνδεση της εφαρμογής με την OWL οντολογία, τη φόρτωση της οντολογίας σε RDF γράφο, την ανάκτηση μοτίβων ασφάλειας μέσω SPARQL, τη δημιουργία σημασιολογικών οντοτήτων, τη λογική διαγραφή και την ελεγχόμενη εκτέλεση επερωτήσεων SPARQL από τον χρήστη. Το υλικό αυτό λειτουργεί συμπληρωματικά προς την περιγραφή της υλοποίησης του Κεφαλαίου 3 και παρέχεται για τον αναγνώστη που επιθυμεί να εξετάσει αναλυτικότερα επιλεγμένα σημεία της τεχνικής υλοποίησης.

I.1 Αρχικοποίηση της εφαρμογής και σύνδεση με την οντολογία

Κατά την εκκίνηση της εφαρμογής, η διαδρομή προς το αρχείο security_patterns.owl ανακτάται από το configuration του UI project. Εάν η τιμή δεν έχει οριστεί, η εφαρμογή τερματίζει με κατάλληλο μήνυμα σφάλματος, ενώ οι σχετικές διαδρομές μετατρέπονται σε απόλυτες με βάση το content root της εφαρμογής. Στη συνέχεια, η διαδρομή της οντολογίας μεταβιβάζεται στις υπηρεσίες μέσω Dependency Injection.

```
var owlFilePath =
    builder.Configuration.GetValue<string>("OwlFilePath");

if (string.IsNullOrWhiteSpace(owlFilePath))
{
    throw new InvalidOperationException(
        "OwlFilePath is missing from appsettings.json.");
}

if (!Path.IsPathRooted(owlFilePath))
{
    owlFilePath =
        Path.Combine(
            builder.Environment.ContentRootPath,
            owlFilePath);
}

builder.Services.AddServices(owlFilePath);
builder.Services.AddControllersWithViews();
```

Κώδικας I.1: Ανάκτηση της διαδρομής της οντολογίας και αρχικοποίηση των υπηρεσιών της εφαρμογής.

Η προσέγγιση αυτή αποσυνδέει τη φυσική θέση του αρχείου OWL από τα ανώτερα επίπεδα της εφαρμογής και επιτρέπει την αλλαγή της διαδρομής χωρίς τροποποίηση του Business Logic Layer ή του Data Access Layer.

I.2 Φόρτωση της OWL οντολογίας και δημιουργία του RDF γράφου

Η φόρτωση της κεντρικής οντολογίας πραγματοποιείται στον SparqlAdapter. Το αρχείο OWL μετατρέπεται σε RDF γράφο και στη συνέχεια δημιουργούνται ένα InMemoryDataset και ένας LeviathanQueryProcessor, πάνω στον οποίο εκτελούνται οι SPARQL επερωτήσεις της εφαρμογής.

```
public SparqlAdapter(string owlFilePath)
{
    _owlFilePath = owlFilePath;

    _graph =
        LoadGraphFromCanonicalOntology();

    _dataset =
        new InMemoryDataset(_graph);

    _processor =
        new LeviathanQueryProcessor(_dataset);
}

private Graph LoadGraphFromCanonicalOntology()
{
    return GraphBuilder
        .Create()
        .FromFile(_owlFilePath)
        .Build();
}
```

Κώδικας I.2: Φόρτωση της κεντρικής OWL οντολογίας και δημιουργία του in-memory μηχανισμού SPARQL.

Με τον τρόπο αυτό οι επερωτήσεις δεν απαιτούν την επαναφόρτωση ολόκληρου του OWL αρχείου σε κάθε αίτημα του χρήστη, αλλά εκτελούνται πάνω στην ήδη διαθέσιμη αναπαράσταση στη μνήμη.

I.3 Ανάκτηση ενεργών μοτίβων ασφάλειας μέσω SPARQL

Η βασική επερώτηση ανάκτησης των μοτίβων ασφάλειας αξιοποιεί το property path rdfs:subClassOf*, ώστε να επιστρέφονται individuals που ανήκουν όχι μόνο απευθείας στην SecurityPattern, αλλά και σε οποιαδήποτε υποκλάση της. Παράλληλα, αποκλείονται τα μοτίβα που έχουν επισημανθεί ως λογικά διαγραμμένα.

```
PREFIX sp: <http://example.org/security-patterns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT DISTINCT
    ?pattern
    ?name
    ?description
WHERE {
    ?pattern a ?type .
    ?type rdfs:subClassOf* sp:SecurityPattern .

    OPTIONAL { ?pattern rdfs:label ?name . }
    OPTIONAL { ?pattern sp:hasDescription ?description . }
    OPTIONAL { ?pattern sp:isDeleted ?isDeleted . }

    FILTER(
        !BOUND(?isDeleted) ||
        LCASE(STR(?isDeleted)) != "true"
    )
}
```

Κώδικας I.3: SPARQL επερώτηση για την ανάκτηση των ενεργών security patterns ανεξάρτητα από το βάθος ταξινόμησής τους.

Η συγκεκριμένη επερώτηση επιτρέπει στην εφαρμογή να ακολουθεί τη δυναμική ιεραρχία της οντολογίας, χωρίς η ταξινόμηση των μοτίβων να είναι ενσωματωμένη ως στατική λογική στον κώδικα της εφαρμογής.

I.4 Δημιουργία νέου μοτίβου ασφάλειας στην OWL οντολογία

Κατά τη δημιουργία ενός νέου μοτίβου ασφάλειας, το Data Access Layer δημιουργεί ένα νέο OWL Named Individual και προσθέτει τις αντίστοιχες Class Assertions, Data Property Assertions και Object Property Assertions. Με αυτόν τον τρόπο, το νέο μοτίβο δεν αποθηκεύεται ως απλή εγγραφή δεδομένων, αλλά εντάσσεται στο σημασιολογικό μοντέλο της οντολογίας.

```
writer.AddNamedIndividual(patternIri);

foreach (string categoryIri in categories){
    writer.AddClassAssertion(
        categoryIri,
        patternIri);
}

writer.AddLabel(
    patternIri,
    patternName);

foreach (var property in dataProperties){
    foreach (string value in property.Value
        .Where(value => !string.IsNullOrWhiteSpace(value))
        .Distinct(StringComparer.Ordinal))
    {
        writer.AddDataPropertyAssertion(
            property.Key,
            patternIri,
            value);
    }
}

foreach (var property in objectProperties){
    foreach (string objectIri in property.Value
        .Where(iri => !string.IsNullOrWhiteSpace(iri))
        .Distinct(StringComparer.Ordinal))
    {
        writer.AddObjectPropertyAssertion(
            property.Key,
            patternIri,
            objectIri);
    }
}

writer.Save();
```

Κώδικας I.4: Δημιουργία νέου security pattern ως OWL Named Individual με Class, Data Property και Object Property Assertions.

Μετά την αποθήκευση, η νέα πληροφορία ενσωματώνεται και στην ενεργή RDF αναπαράσταση, ώστε το νέο μοτίβο να είναι άμεσα διαθέσιμο στις λειτουργίες αναζήτησης και ανάκτησης.

I.5 Λογική διαγραφή μοτίβου ασφάλειας

Η διαγραφή ενός μοτίβου έχει υλοποιηθεί ως λογική διαγραφή (soft delete). Αντί να αφαιρείται το individual από την οντολογία και να χάνονται οι υπάρχουσες σημασιολογικές του σχέσεις, προστίθεται σχετικό σχόλιο και η ιδιότητα isDeleted λαμβάνει την τιμή true.

```
writer.AddComment(
    patternIri,
    $"[DELETED] Previous Name: {oldName} | " +
    $"Previous Description: {oldDescription}");

writer.ReplaceDataPropertyAssertions(
    ConstantUris.isDeletedUri,
    patternIri,
    new[] { "true" },
    "http://www.w3.org/2001/XMLSchema#boolean");

writer.Save();
```

Κώδικας I.5: Υλοποίηση της λογικής διαγραφής ενός security pattern στην OWL οντολογία.

Η επιλογή αυτή επιτρέπει τη διατήρηση της πληροφορίας στην οντολογία, ενώ οι κανονικές επερωτήσεις της εφαρμογής αποκλείουν τα logically deleted entities από τη συνήθη προβολή.

I.6 Περιορισμοί ασφαλούς εκτέλεσης SPARQL επερωτήσεων

Η ειδική διεπαφή SPARQL επιτρέπει σε έναν εξειδικευμένο χρήστη να υποβάλει δικές του επερωτήσεις. Για την αποφυγή μη επιθυμητών ενεργειών, επιτρέπονται μόνο επερωτήσεις SELECT, αποκλείονται τα SERVICE, FROM και FROM NAMED και εφαρμόζεται μέγιστο όριο αποτελεσμάτων.

```
if (!IsSelectQuery(query))
{
    throw new InvalidOperationException(
        "Only SPARQL SELECT queries are allowed.");
}

if (ContainsServiceClause(query))
{
    throw new InvalidOperationException(
        "SPARQL SERVICE clauses are not allowed.");
}

if (query.DefaultGraphNames.Any() ||
    query.NamedGraphNames.Any())
{
    throw new InvalidOperationException(
        "SPARQL FROM and FROM NAMED clauses are not allowed.");
}

if (query.Limit < 0 ||
    query.Limit > MaxUserQueryResults)
{
    query.Limit = MaxUserQueryResults;
}
```

Κώδικας I.6: Περιορισμοί ασφαλούς εκτέλεσης των SPARQL queries που υποβάλλονται από τον χρήστη.

Οι περιορισμοί αυτοί επιτρέπουν την ελεγχόμενη διερεύνηση της σημασιολογικής βάσης γνώσης, χωρίς η συγκεκριμένη διεπαφή να λειτουργεί ως γενικός μηχανισμός τροποποίησης της οντολογίας ή πρόσβασης σε απομακρυσμένες RDF πηγές.