«ΠΡΟΗΓΜΕΝΕΣ ΜΕΘΟΔΟΙ ΚΑΙ ΤΕΧΝΙΚΕΣ ΓΙΑ ΤΗΝ ΑΠΟΔΟΤΙΚΗ ΛΕΙΤΟΥΡΓΙΑ ΣΥΣΤΗΜΑΤΩΝ ΜΕ ΧΡΗΣΗ ΥΠΗΡΕΣΙΩΝ ΔΙΑΔΙΚΤΥΟΥ (WEB SERVICES)»

Πούλια Αδαμοπούλου

Διπλωματική Εργασία για το Μεταπτυχιακό
Δίπλωμα Ειδίκευσης στην:

Επιστήμη και Τεχνολογία Υπολογιστών

Επιβλέπων:
Καθηγητής κ. Αθανάσιος Τσακαλίδης

Εξεταστική Επιτροπή:
Καθηγητής κ. Αθανάσιος Τσακαλίδης
Καθηγητής κ. Σπυρίδων Λυκοθανάσης
Αν. Καθηγητής. κ. Ιωάννης Γαροφαλάκης

Τμήμα Μηχανικών Η/Υ και Πληροφορικής,
Πανεπιστήμιο Πατρών

Μάρτιος 2005

Περίληψη

Στη διπλωματική, παρουσιάζεται μια μελέτη και κριτική παρουσίαση των ερευνητικών τεχνικών που αποτελούν τις σύγχρονες επιστημονικές κατευθύνσεις για την αποδοτική λειτουργία συστημάτων με χρήση Υπηρεσιών Διαδικτύου (Web Services).

Μεγάλο μέρος των τεχνικών που παρουσιάζονται εστιάζουν στη βελτίωση του προτύπου UDDI με σκοπό την αποδοτικότερη δημοσίευση και ανακάλυψη Web Services. Οι τεχνικές αυτές περιλαμβάνουν την επέκταση των UDDI δομών δεδομένων ώστε να περιλαμβάνουν πληροφορίες Ποιότητας Υπηρεσίας, την εισαγωγή των «μπλε» σελίδων που θα περιέχουν σημασιολογικές πληροφορίες, και την σημασιολογική συσχέτιση των Web Services μέσο αλγορίθμου που εφαρμόζεται στα αποτελέσματα αναζήτησης με σκοπό την ανάκτηση εγκυρότερων αποτελεσμάτων.

Παρουσιάζεται επίσης ένα πλαίσιο με βάση οντολογίες που εξασφαλίζει την συντακτική και σημασιολογική συμβατότητα των Web Services, με σκοπό την αυτόματη σύνθεση υπηρεσιών, η οποία αποτελεί σημαντικό βήμα για την ενεργοποίηση του Σημασιολογικού Ιστου. Έμφαση δίνεται και στο δυναμικό χειρισμό εξαιρέσεων, όπως η παραβίαση των ποιοτικών περιορισμών, ώστε να διευκολύνεται η ενοποίηση ετερογενών και αυτόματων εφαρμογών.

Καθώς περισσότεροι από έναν πάροχοι μπορεί να προσφέρουν Web Services με την ίδια λειτουργικότητα, απαιτούνται προηγμένες μέθοδοι επιλογής των υπηρεσιών που ικανοποιούν τόσο τις λειτουργικές όσο και τις μη λειτουργικές απαιτήσεις των πελατών. Οι μεσολαβητές ποιότητας υπηρεσίας συλλέγουν πληροφορίες QoS για τους παρόχους και λαμβάνουν αποφάσεις επιλογής υπηρεσίας για τους πελάτες.

Στα πλαίσια της διπλωματικής σχεδιάστηκε και υλοποιήθηκε μια εφαρμογή μεσολαβητή διαχείρισης ποιότητας υπηρεσίας (QoS Broker), χρησιμοποιώντας την τεχνολογία των WS Resources. Ο broker διαχειρίζεται κλάσεις Web Services με τα ίδια λειτουργικά χαρακτηριστικά αλλά διαφορετικά ποιοτικά χαρακτηριστικά και εξυπηρετεί τους πελάτες χρησιμοποιώντας διαφορετικές πολιτικές επιλογής υπηρεσίας. Η λειτουργικότητα του WSRF.NET Broker επιβεβαιώνεται από πειραματικά αποτελέσματα

Abstract

In the thesis, there will be presented a study and review of several scientific techniques that constitute the current research directions for the efficient operation of systems using Web Services.

Most of the presented techniques focus on the improvement of the UDDI standard, aiming at the more efficient publication and discovery of Web Services. This techniques include the extension of the UDDI data structures so that they provide Quality of Service information, the introduction of the "blue" pages that contain semantic information, and the semantic cross-correlation of Web Services by applying an algorithm on the search results in order to retrieve better results.

There is also presented a framework based on ontologies that ensures the syntactic and semantic compatibility of Web Services, aiming to the automatic composition of Web Services, which constitutes an important step for enabling the Semantic Web. Emphasis is also given in the dynamic handling of exceptions, such as the violation of qualitative restrictions, in order to facilitate the integration of heterogeneous and automatic applications.

Since more than one providers can offer Web Services with the same functionality, there are presented advanced methods of service selection that satisfy not only the functional but also the qualitative requirements of the customers. The quality of service mediators collect QoS information for the providers and select services for the customers.

In the frames of this thesis there is designed and developed an application of a quality of service mediator (QoS Broker), using the technology of WS Resources. The broker manages groups of Web Services with the same functional characteristics and different qualitative characteristics and uses different service selection policies to serve the customers. The functionality of the WSRF.NET Broker is certified by experimental results.

Περιεχόμενα

Περίληψη i
Abstract ii
Περιεχόμενα iii
Λίστα Πινάκων vi
Ευχαριστίες viii
Εισαγωγή 1
1. Προαπαιτούμενες έννοιες 3
1.1. Επικοινωνία: SOAP 3
1.1.1. Μήνυμα SOAP 3
1.2. Περιγραφή: WSDL 6
1.2.1. Αφηρημένη περιγραφή 7
1.2.2. Συγκεκριμένες δεσμευτικές πληροφορίες 8
1.3. Κατάλογος: UDDI 9
1.3.1. Οργάνωση της δομής 9
1.3.2. Τεχνικές περιγραφές και tModels 12
1.3.3. Κατηγοριοποίηση 13
1.4. Ορισμοί Υπηρεσιών Διαδικτύου 13
1.4.1. Ρόλοι στο πρότυπο Υπηρεσιών Διαδικτύου 14
1.4.2. Διαδικασίες στο πρότυπο υπηρεσιών Διαδικτύου 14
1.4.3. Artefacts μιας Υπηρεσίας Διαδικτύου 14
2. Προδιαγραφές Υπηρεσιών Διαδικτύου με βελτιώσεις του UDDI 16
2.1. E-UDDI: Βελτιώσεις προς μια πλήρη Προδιαγραφή 17
2.2. Ένα πρότυπο Πλαίσιο Προδιαγραφών 17
2.2.1. Εφαρμογή των Βελτιώσεων σε Λευκές και Κίτρινες Σελίδες 18
2.2.2. Εφαρμογή των βελτιώσεων στις Πράσινες Σελίδες 18
2.2.3. Εισαγωγή των Μπλε Σελίδων για Σημασιολογικές Προδιαγραφές 19
2.2.4. Ένα Πρότυπο Δεδομένων Κατάλληλο για το E-UDDI 20
2.3. Συμπεράσματα 20
3. Ένα πρότυπο για Ανακάλυψη Υπηρεσιών Διαδικτύου με Ποιότητα Υπηρεσίας 21
3.1. Κατηγορίες ποιότητας υπηρεσίας (QoS) 21
3.1.1. Ποιότητα Υπηρεσίας σε χρόνο εκτέλεσης 21
3.1.2. Διαχείριση Προδιαγραφών και Ποιότητα Υπηρεσίας Κόστους 22
3.1.3. Ποιότητα Υπηρεσίας Ασφάλειας 22
3.2. Επέκταση του προτύπου UDDI 23
3.3. Εκτεταμένες UDDI δομές δεδομένων 25
3.4. Συμπεράσματα για το προτεινόμενο πρότυπο 26
4. Αλγόριθμος για αντιστοίχηση Υπηρεσιών Διαδικτύου 28
4.1. Ανακάλυψη Υπηρεσιών Διαδικτύου 28
4.2. Ο αλγόριθμος και η εφαρμογή 28
4.3. Αξιολόγηση 30
5. Σύνθεση Υπηρεσιών Διαδικτύου στον Σημασιολογικό Ιστό 32
5.1. Σημασιολογική περιγραφή Υπηρεσιών Διαδικτύου 32
5.1.1. Τρόποι λειτουργίας και μηνύματα 34
5.1.2. Σκοπός και κατηγορία 36
5.1.3. Ποιότητα Λειτουργίας 36

5.1.4. Καθορίζοντας λειτουργίες και Υπηρεσίες Διαδικτύου 37
5.2. Μοντέλο Συνθεσιμότητας για Υπηρεσίες Διαδικτύου 38
5.2.1. Τρόπος λειτουργίας και συνθεσιμότητα σύνδεσης 39
5.2.2. Συνθεσιμότητα μηνυμάτων 40
5.2.3. Σημασιολογική συνθεσιμότητα λειτουργιών 41
5.2.4. Ποιοτική Συνθεσιμότητα 41
5.2.5. Αρτιότητα σύνθεσης 42
5.3. Αυτόματη σύνθεση Υπηρεσιών Διαδικτύου 43
5.3.1. Φάση Προδιαγραφής 44
5.3.2. Φάση ταιριάσματος 45
5.3.3. Φάση επιλογής 47
5.3.4. Φάση Παραγωγής 48
6. Ο σχεδιασμός αλγορίθμων μεσολαβητών Ποιότητας Υπηρεσίας για Υπηρεσίες Διαδικτύου με QoS ιδιότητες 50
6.1. Η αρχιτεκτονική QCWS 50
6.1.1. Εξυπηρετητής 50
6.1.2. Ο μεσολαβητής QoS 51
6.1.3. Ο διαχειριστής πληροφοριών QoS 51
6.1.4. Διαχειριστή Διαπραγμάτευσης QoS 51
6.1.5. Αναλυτής QoS 52
6.1.6. Πελάτης 52
6.2. Αλγόριθμοι Allocation Πόρων Μεσολαβητή QoS 52
6.2.1. Η QCWS αρχιτεκτονική Εξυπηρετητή-Μεσολαβητή 52
6.2.2. Αλγόριθμοι QCWS Resource Allocation 53
6.3. Μελέτη Απόδοσης 56
6.3.1. Υποθέσεις 56
6.3.2. Απόδοση του HQ 56
6.3.3. Απόδοση του RQ 58
6.4. Συμπεράσματα για τον QCWS 60
7. Χειρισμός Εξαιρέσεων σε Συνεργατικές Διαδικασίες με βάση Υπηρεσίες Διαδικτύου 61
7.1. Αρχιτεκτονική του Web-Flow 61
7.2. Περιορισμοί Ποιότητας 63
7.3. Δυναμικός Χειρισμός Εξαιρέσεων 65
7.4. Συμπεράσματα για το δυναμικό χειρισμό εξαιρέσεων 66
8. Αποτελεσματική Πρόσβαση σε Υπηρεσίες Διαδικτύου 67
8.1. Σενάριο: Προγραμματισμός ενός ταξιδιού 67
8.2. Ερωτήματα σε Υπηρεσίες Διαδικτύου 67
8.2.1. Ένα μοντέλο ερωτημάτων για Υπηρεσίες Διαδικτύου 68
8.2.2. Υλοποίηση 75
8.3. Συμπεράσματα για το μοντέλο ερωτημάτων 76
9. Το πλαίσιο WS-Resource 77
9.1. WS-Resource 78
9.2. Κύκλος ζωής του WS-Resource 78
9.2.1. Το πρότυπο WS-Resource Factory 78
9.2.2. WS-Resource Ταυτότητα 78

9.2.3. Καταστροφή WS-Resource 79
9.3. WS-Resource Ιδιότητες 80
9.4. Renewable References 81
9.5. Ομάδες Υπηρεσιών 81
9.6. Base Faults 81
9.7. Ειδοποίηση 82
10. Περιγραφή του μεσολαβητή διαχείρισης υπηρεσίας με κριτήρια ποιότητας. 83
10.1. Υλοποίηση με χρήση του WSRF.NET 85
10.2. Πειραματικά αποτελέσματα 85
10.2.1. Γραφήματα πρώτου συνόλου πειραμάτων με δεδομένα εισόδου για εξυπηρέτηση πελατών με χαμηλές απαιτήσεις χωρητικότητας. 86
10.2.2. Γραφήματα του δεύτερου συνόλου πειραμάτων με δεδομένα εισόδου για εξυπηρέτηση πελατών με υψηλές απαιτήσεις χωρητικότητας. 90
11. Βιβλιογραφία 94
12. ΠΑΡΑΡΤΗΜΑ Α 96
13. ΠΑΡΑΡΤΗΜΑ Β 101
13.1. Δεδομένα πειράματος 101
13.2. C# κώδικας του WSBroker 102

Λίστα Πινάκων

Πίνακας 1:Παράδειγμα μηνύματος SOAP 4
Πίνακας 2: Παράδειγμα κλήσης SOAP RPC 5
Πίνακας 3: Παράδειγμα απάντησης SOAP RPC 6
Πίνακας 4: Παράδειγμα αφηρημένης περιγραφής WSDL 7
Πίνακας 5: Παράδειγμα συγκεκριμένης δεσμευτικής πληροφορίας WSDL 9
Πίνακας 6: Παράδειγμα δομής businessEntity UDDI 11
Πίνακας 7: Παράδειγμα δομής businessService UDDI 12
Πίνακας 8: Παράδειγμα δομής tModel UDDI 13
Πίνακας 9: Τύποι δομών δεδομένων των UDDI εγγραφών 25
Πίνακας 10: Ποσοστά υπηρεσιών G2 στα σύνολα αντιστοιχισμένων υπηρεσιών για ορισμένες υπηρεσίες G1 με τον αλγορίθμο Non-SVD και τον βασισμένο στον SVD αλγορίθμο 31
Πίνακας 11: Η CSSL προδιαγραφή για την σύνθετη υπηρεσία car broker 45
Πίνακας 12: Ο αλγόριθμος ταιριάσματος 46
Πίνακας 13: Συνθεσιμότητα μηνυμάτων και συναρτήσεις ελέγχου αρτιότητας 47
Πίνακας 14: WSFL περιγραφές που παρήχθησαν από την υπηρεσία car broker 48
Πίνακας 15: Οι επανακατανομές που συμβαίνουν και οι πελάτες που επαναδιαμορφώνονται. 54
Πίνακας 16: Παράδειγμα περιορισμού αποτελέσματος 65
Πίνακας 17: Παράδειγμα κανόνα χειρισμού εξαιρέσεων 66

Λίστα Σχημάτων

Σχήμα 1:Ρόλοι, διαδικασίες και artefacts Υπηρεσιών Διαδικτύου 13
Σχήμα 2: Τρέχον μοντέλο δέσμευσης Υπηρεσιών Διαδικτύου (push-bind) 23
Σχήμα 3:Ένα νέο μοντέλο εγγραφής και ανακάλυψης Υπηρεσιών Διαδικτύου 24
Σχήμα 4: tModel για πληροφορίες ποιότητας υπηρεσίας 26
Σχήμα 5: Αίτημα SOAP για ανακάλυψη υπηρεσίας 26
Σχήμα 6: Περιγραφή Υπηρεσιών Διαδικτύου με βάση Οντολογίες 33
Σχήμα 7: Εφαρμογή μεσίτευσης αυτοκινήτων 34
Σχήμα 8: Μοντέλο συνθεσιμότητας για Υπηρεσίες Διαδικτύου 38
Σχήμα 9: Αρτιότητα Σύνθεσης Σχήμα 9β Σχήμα 9γ 42
Σχήμα 10:Παρουσίαση της προτεινόμενης προσέγγισης για σύνθεση υπηρεσιών 43
Σχήμα 11: Αρχιτεκτονική του συστήματος 51
Σχήμα 12: Αρχιτεκτονική Server-Broker QCWS 52
Σχήμα 13: Ρυθμός επαναρυθμίσεων και μέση χρησιμότητα χρησιμοποιώντας τον HQ (N=60) 56
Σχήμα 14: Ρυθμός επαναδιαμόρφωση και μέση utility χρησιμοποιώντας τον HQ (N=40) 57
Σχήμα 15: Ρυθμός επαναδιαμόρφωση και μέση utility χρησιμοποιώντας τον HQ(N=80) 58
Σχήμα 16: Μέση utility/ επαναδιαμόρφωση vs βάρος 59
Σχήμα 17: Απόδοση με χρήση μεταβλητής και σταθεράς T_1 59
Σχήμα 18: Απόδιση του μεσολαβητή ως προς το βάρος 60
Σχήμα 19: Η Web Flow αρχιτεκτονική 62

Σχήμα 20: Σχήμα ερωτημάτων τριών επιπέδων. 69

Λίστα Γραφημάτων

Γράφημα 1:Πολιτική 1-Χαμηλό φορτίο 86
Γράφημα 2:Πολιτική 2-Χαμηλό φορτίο 87
Γράφημα 3: Πολιτική 3-Χαμηλό φορτίο 87
Γράφημα 4: Πολιτική 4-Χαμηλό φορτίο 88
Γράφημα 5: Σύγκριση μεθόδων –Χαμηλό φορτίο 89
Γράφημα 6:Σύγκριση μεθόδων-Χαμηλό φορτίο(μεγαλύτερη ανάλυση) 89
Γράφημα 7: Πολιτική 1-Υψηλό φορτίο 90
Γράφημα 8:Πολιτική 2-Υψηλό φορτίο 91
Γράφημα 9: Πολιτική 3-Υψηλό φορτίο 91
Γράφημα 10: Πολιτική 4-Υψηλό φορτίο 92
Γράφημα 11: Σύγκριση μεθόδων-Υψηλό φορτίο 93
Γράφημα 12: Σύγκριση μεθόδων-Υψηλό φορτίο(μεγαλύτερη ανάλυση) 93

Ευχαριστίες

Στο σημείο αυτό θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα μου Καθηγητή κ. Α. Τσακαλίδη για την καθοδήγηση του στην πραγματοποίηση αυτής της εργασίας καθώς και τα υπόλοιπα μέλη της τριμελούς επιτροπής, τον Καθηγητή κ. Σ. Λυκοθανάση και τον Αν. Καθηγητή κ. Ι. Γαροφαλάκη για την συνεργασία που είχαμε στην εκπόνηση αυτής της διπλωματικής.

Τέλος, θα ήθελα να ευχαριστήσω τον υποψήφιο διδάκτορα κ. Ε. Σακκόπουλο για την ουσιαστική υποστήριξη και την πολύτιμη βοήθειά του. Η καθοδήγηση του έπαιξε καθοριστικό ρόλο στην ολοκλήρωση αυτής της εργασίας.

Εισαγωγή

Οι Υπηρεσίες Διαδικτύου (Web Services) χρησιμοποιούνται ευρέως για να ενοποιήσουν ετερογενείς και αυτόνομες εφαρμογές σε διεπιχειρησιακές συνεργασίες. Υιοθετώντας πρόσφατα τεχνολογικά πρότυπα όπως το SOAP, το WSDL και το UDDI, υπηρεσίες από διαφορετικούς παρόχους μπορούν να συνεργαστούν αποτελεσματικά σε σύνθετες υπηρεσίες, ανεξάρτητα από πλατφόρμα φιλοξενείας και υλοποίησης, συνδυάζοντας τη λειτουργικότητά τους. Αυτή η διαλειτουργικότητα επιτρέπει στις επιχειρήσεις να δημοσιεύουν δυναμικά, να ανακαλύπτουν, και να διαχειρίζονται Υπηρεσίες Διαδικτύου ώστε να δημιουργούν ευκολότερα καινοτόμα προϊόντα. Οι εφαρμογές που αναμένεται να εκμεταλλευτούν έντονα τα πλεονεκτήματα των Υπηρεσιών Διαδικτύου περιλαμβάνουν το ηλεκτρονικό εμπόριο και την ηλεκτρονική διακυβέρνηση.

Στο πεδίο των Υπηρεσιών Διαδικτύου υπάρχουν αρκετά ανοικτά επιστημονικά πεδία που αφορούν σε τεχνολογικές λύσεις με τη χρήση αποδοτικών μεθόδων που θα αναπτύσσουν, θα ανακαλύπτουν και θα διαχειρίζονται Υπηρεσίες Διαδικτύου, ενώ σημαντική είναι και η διασφάλιση των ποιοτικών χαρακτηριστικών των παρεχόμενων υπηρεσιών. Η δημιουργία ενός περιβάλλοντος όπου θα αξιοποιούνται τα πλεονεκτήματα των Υπηρεσιών Διαδικτύου απαιτεί ολοκληρωμένες λύσεις για την περιγραφή, ανακάλυψη, καταλογογράφηση, και παρακολούθηση μιας σειράς από αυτόνομες και ετερογενείς Υπηρεσίες Διαδικτύου.Στόχος της διπλωματικής είναι η περιγραφή και πειραματική εξέταση προηγμένων διαδικασιών διαχείρισης Υπηρεσιών Διαδικτύου.

Στο πρώτο κεφάλαιο της διπλωματικής παρουσιάζονται οι εισαγωγικές έννοιες της τεχνολογίας των Υπηρεσιών Διαδικτύου, και συγκεκριμένα τα τεχνολογικά πρότυπα όπως το SOAP, το WSDL και το UDDI που έχουν υιοθετηθεί για την επικοινωνία, την περιγραφή και την ανακάλυψη των Υπηρεσιών Διαδικτύου.

Προκειμένου το UDDI να υποστηρίξει αποδοτικότερα την ανακάλυψη και τη διαμόρφωση των Υπηρεσιών Διαδικτύου, που είναι οι βασικοί τομείς εφαρμογής του, απαιτούνται επιπρόσθετες προδιαγραφές που επιτρέπουν την αξιολόγηση πιθανών (σημασιολογικών και πραγματικών) ετερογενειών μεταξύ Υπηρεσιών Διαδικτύου όπως επίσης και προδιαγραφές σχετικά με το κόστος. Στο κεφάλαιο 2 προτείνονται διάφορες βελτιώσεις του UDDI με σκοπό τη διευκόλυνση της ανακάλυψης και διαμόρφωσης Υπηρεσιών Διαδικτύου.

Καθώς οι Υπηρεσίες Διαδικτύου μπορούν να παρέχονται από τρίτα μέρη και να καλούνται δυναμικά μέσο του Διαδικτύου, η ποιότητα υπηρεσίας τους (QoS) μπορεί να ποικίλει σημαντικά. Επομένως είναι σημαντικό να υπάρξει ένα πλαίσιο εξασφάλισης της ποιότητας υπηρεσίας που παρέχεται από τον πάροχο, της ποιότητας υπηρεσίας που απαιτείται από τον πελάτη, και της αντιστοιχίας μεταξύ τους κατά την ανακάλυψη της Υπηρεσίας Διαδικτύου που ικανοποιεί την απαιτούμενη QoS. Στο κεφάλαιο 3 προτείνεται ένα νέο πρότυπο ανακάλυψης Υπηρεσιών Διαδικτύου που λαμβάνει υπόψη για την ανακάλυψη υπηρεσιών τις λειτουργικές και μη λειτουργικές απαιτήσεις. Ένας νέος ρόλος εισάγεται σε αυτό το πλαίσιο – ο εγγυητής, ο οποίος επιβεβαιώνει τις υποσχέσεις ποιότητας υπηρεσιών του προμηθευτή. Επίσης προτείνεται μια επέκταση των τύπων δομών δεδομένων τουUDDI που θα μπορούσε να χρησιμοποιηθεί για την υλοποίηση του προτεινόμενου εκτεταμένου UDDI προτύπου

Οι κατάλογοι UDDI παρέχουν αναζητήσεις λέξεων-κλειδιών για τις Υπηρεσίες Διαδικτύου. Η λειτουργία αναζήτησης είναι πολύ απλή και αποτυγχάνει να ανακαλύψει τις συσχετίσεις μεταξύ των Υπηρεσιών Διαδικτύου. Στο κεφάλαιο 4 προτείνεται ένας αλγόριθμος που ανακτά τις στενά συσχετιζόμενες Υπηρεσίες Διαδικτύου. Ο προτεινόμενος αλγόριθμος βασίζεται στη Singular Value Decomposition η οποία αποκαλύπτει σημασιολογικές σχέσεις μεταξύ των Υπηρεσιών Διαδικτύου.

Στη συνέχεια (κεφάλαιο 5) παρουσιάζεται ένα πλαίσιο με βάση οντολογίες για την αυτόματη σύνθεση Υπηρεσιών Διαδικτύου. Προτείνεται μια τεχνική από δηλωτικές περιγραφές υψηλού επιπέδου για την παραγωγή σύνθετων υπηρεσιών και ορίζονται κάποιες οδηγίες για αποτελεσματική σύνθεση με τη χρήση κανόνων. Οι κανόνες αυτοί συγκρίνουν τα

συντακτικά και σημασιολογικά χαρακτηριστικά των Υπηρεσιών Διαδικτύου για να καθορίζουν αν δύο Υπηρεσίες Διαδικτύου είναι συνθέσιμες.

Στο κεφάλαιο 6 παρουσιάζεται μια αρχιτεκτονική Υπηρεσιών Διαδικτύου με ιδιότητες Ποιότητας Υπηρεσίας η οποία χρησιμοποιεί ένα υποσύστημα μεσολαβητή QoS μεταξύ πελατών και παρόχων. Οι λειτουργίες του μεσολαβητή QoS περιλαμβάνουν τη συλλογή QoS πληροφοριών για τους παρόχους, τη λήψη αποφάσεων επιλογής για τους πελάτες, και τη διαπραγμάτευση με τους παρόχους για να λάβουν την ποιότητα υπηρεσίας που υποσχονται. Προτείνονται δύο αλγόριθμοι δέσμευσης πόρων που χρησιμοποιούνται από τον μεσολαβητή QoS όταν αυτός λειτουργεί ως front-end ενός εξυπηρετητή, με στόχο τη μεγιστοποίηση της χρησιμοποίηση των πόρων του εξυπηρετητή και την ελαχιστοποίηση της αστάθεια QoS.

Η υποστήριξη μιας υψηλής ποιότητας εκτέλεσης συνεργατικών διαδικασιών με βάση τις Υπηρεσίες Διαδικτύου απαιτεί το δυναμικό χειρισμό εξαιρέσεων. Η προσέγγιση που παρουσιάζεται στο κεφάλαιο 7 βασίζεται στον καθορισμό μιας ποικιλίας από ποιοτικούς περιορισμούς για ετερογενείς και αυτόνομες Υπηρεσίες Διαδικτύου. Ένα υποσύστημα κανόνων για την παρακολούθηση και το χειρισμό εξαιρέσεων αντιμετωπίζει αυτόματα πολλές εξαιρέσεις όπως η παραβίαση ποιοτικών περιορισμών ή τα λάθη υπηρεσιών.

Στο κεφάλαιο 8 παρουσιάζεται ένα πρότυπο βελτιστοποίησης ερωτημάτων βασισμένο στη συνάθροιση των παραμέτρων της ποιότητας υπηρεσίας διαφορετικών Υπηρεσιών Διαδικτύου. Το πρότυπο αυτό ρυθμίζει την ποιότητα υπηρεσίας μέσω ενός δυναμικού σχήματος αξιολόγησης και μιας πολυεπίπεδης αντιστοίχησης που επιτρέπει την εύρεση παρόμοιων και μερικών απαντήσεων σε κλήσεις Υπηρεσιών Διαδικτύου. Παρουσιάζεται μια νέα υποδομή που προσφέρει σύνθετες και βελτιστοποιημένες δυνατότητες ερωτημάτων για Υπηρεσίες Διαδικτύου όπου οι χρήστες υποβάλλουν δηλωτικά ερωτήματα και η υποδομή τα επιλύει μέσω συνδυασμένων κλήσεων διαφορετικών λειτουργιών Υπηρεσιών Διαδικτύου.

Στο κεφάλαιο 9 παρουσιάζονται οι προδιαγραφές του πλαισίου WS-Resource το οποίο παρέχει τυποποιημένες και διαλειτουργικές μεθόδους για την διαχείριση stateful πόρων. Η προσέγγιση WS-Resource μοντελοποιεί τις καταστάσεις σε ένα πλαίσιο Υπηρεσιών Διαδικτύου. Μια WS-Resource ορίζεται ως η σύνθεση μιας Υπηρεσίας Διαδικτύου και ενός stateful πόρου που 1) εκφράζεται ως μια συσχέτιση ενός XML εγγράφου ορισμένου τύπου με ένα port Type Υπηρεσιών Διαδικτύου, και 2) απευθύνεται και προσπελαύνεται σύμφωνα με ένα υπονοούμενο πρότυπο πόρων, μια συμβατική χρήση των WS-Addressing endpoint αναφορών.

Στα πλαίσια αυτής της διπλωματικής υλοποιήθηκε ένα σύστημα διαχείρισης Υπηρεσιών Διαδικτύου χρησιμοποιώντας την τεχνολογία των WS Resources. Ο μεσολαβητής (broker) διαχείρισης υπηρεσίας με κριτήρια ποιότητας συλλέγει πληροφορίες QoS για τους παρόχους υπηρεσιών (εξυπηρετητές) και όταν λάβει αιτήσεις υπηρεσιών από πελάτες επιλέγει τις υπηρεσίες που μπορούν να ικανοποιήσουν τόσο τις λειτουργικές απαιτήσεις όσο και τις απαιτήσεις Ποιότητας Υπηρεσίας των πελατών. Στο κεφάλαιο 10 παρουσιάζονται τα πειραματικά αποτελέσματα για τέσσερις διαφορετικές πολιτικές επιλογής υπηρεσίας.

1. Προαπαιτούμενες έννοιες

Το πλαίσιο λειτουργίας των Υπηρεσιών Διαδικτύου διαιρείται σε τρεις περιοχές- το πρωτόκολλο επικοινωνίας, την περιγραφή και την ανακάλυψη υπηρεσίας. Οι προδιαγραφές υπηρεσιών αναπτύσσονται για κάθε μια από αυτές ως εξής:

- Το SOAP [1], το οποίο επιτρέπει την επικοινωνία μεταξύ των υπηρεσιών Διαδικτύου
- Η "γλώσσα περιγραφής υπηρεσιών Διαδικτύου" (WSDL) [2], η οποία παρέχει μια επίσημη, αναγνώσιμη από τον υπολογιστή περιγραφή των Υπηρεσιών Διαδικτύου, και
- Ο κατάλογος από περιγραφές Υπηρεσιών Διαδικτύου (UDDI) [3].

1.1. Επικοινωνία: SOAP

Λαμβάνοντας υπόψη την κατανεμημένη και ετερογενή φύση του Διαδικτύου, οι μηχανισμοί επικοινωνίας πρέπει να είναι ανεξάρτητοι από πλατφόρμες, διεθνείς, και όσο το δυνατόν πιο συνεπείς. Το XML καθιερώνεται σήμερα σταθερά ως γλώσσα κωδικοποίησης των πληροφοριών και των δεδομένων με σκοπό την ανεξαρτησία των πλατφορμών και τη διεθνοποίηση. Η οικοδόμηση ενός πρωτοκόλλου επικοινωνίας που χρησιμοποιεί XML είναι έτσι μια φυσική απάντηση για τις υπηρεσίες Διαδικτύου.

Το SOAP είναι ένα βασισμένο στην XML πρωτόκολλο για κλήσεις διαδικασιών και μηνυμάτων (RPCs). Προκειμένου να καθοριστεί ένα νέο πρωτόκολλο μεταφοράς, το SOAP λειτουργεί με βάση τα υπάρχοντα πρωτόκολλα όπως το HTTP ή το SMTP.

Ουσιαστικά, ένα μήνυμα SOAP έχει μια πολύ απλή δομή: ένα στοιχείο XML (<Envelope>) με δύο στοιχεία-παιδιά, ένα από τα οποία περιέχει τον προαιρετικό <Header> και το άλλο το <Body>. Τα περιεχόμενα του <Header> και τα στοιχεία του <Body> είναι τα ίδια αυθαίρετα XML.

1.1.1. Μήνυμα SOAP

Στη βασική λειτουργία, το SOAP μπορεί να χρησιμοποιηθεί ως απλό πρωτόκολλο μηνύματος. Σε αυτό το τμήμα, οι προδιαγραφές Υπηρεσιών Διαδικτύου προσδιορίζονται χρησιμοποιώντας ένα απλό παράδειγμα που προέρχεται από το χώρο υπηρεσιών ταξιδιού. Έστω το παράδειγμα ένος ταξιδιώτης που σχεδιάζει μια πτήση και θέλει να καταχωρηθεί ηλεκτρονικά. Θεωρούμε ότι γνωρίζει μια Υπηρεσία Διαδικτύου που παρέχει μια ηλεκτρονική μέθοδο *CheckIn* και το σχήμα για την κωδικοποίηση του *eTicket*. Λαμβάνοντας υπόψη αυτό, θα μπορούσε απλά να δημιουργήσει και να στείλει ένα μήνυμα SOAP σε εκείνη την υπηρεσία για την επεξεργασία.

Ο πίνακας 1 παρουσιάζει ένα τέτοιο μήνυμα SOAP, που έχει μεταφερθεί από το HTTP. Οι επιγραφές HTTP είναι επάνω από στοιχείο SOAP <Envelope>. Η επιγραφή POST δείχνει ότι το μήνυμα χρησιμοποιεί το HTTP POST, το οποίο χρησιμοποιούν και οι διαφυλλιστές. Η επιγραφή Content-Type δηλώνει το text/xml ως τον τύπο του μηνύματος. Εάν υπάρχει μια απάντηση, η απάντηση HTTP θα είναι του ίδιου τύπου περιεχομένου και θα μπορεί να περιέχει ένα μήνυμα SOAP με τα δεδομένα απάντησης. Η επιγραφή SOAPAction δείχνει τον προορισμό του μηνύματος.

```
POST /travelservice HTTP/1.0
Host: xxx.xxx.xxx.xxx
Content-Type: text/xml; charset=utf-8
Content-Length: n
SOAPAction: "urn:travelservice-checkin"

<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
```

```
         xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
     <SOAP-ENV:Body>
        <et:eTicket
             xmlns:et="http://www.acmetravel.com/eticket/schema">
         <et:passengerName first="Joe" last="Smith"/>
         <et:flightInfo airlineName="AA"
      flightNumber="1111"
      departureDate="2002-01-01"
      departureTime="1905"/>
        </et:eTicket>
     </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Πίνακας 1:Παράδειγμα μηνύματος SOAP

Σημειώνεται ότι το μήνυμα στον πίνακα 1 δεν έχει κανένα στοιχείο SOAP <Header>. Το <Body> απλά περιέχει μια αντιπροσωπευτική XML ενός "e-ticket" με τις λεπτομέρειες ονόματος και πτήσης του προσώπου. Οποιοδήποτε ρεαλιστικό ενδοεπιχειρησιακό (B2B) σενάριο θα είχε ενδεχομένως μερικές καταχωρήσεις <Header> ως ένδειξη των συμπληρωματικών πληροφοριών.

1.1.1.1. Απομακρυσμένες κλήσεις διαδικασίας SOAP

Για να χρησιμοποιηθεί το SOAP για κλήσεις διαδικασιών (RPCs), πρέπει να καθοριστεί ένα πρωτόκολλο RPC. Αυτό περιλαμβάνει:

- Τον τρόπο που οι typped τιμές μπορούν να μεταφερθούν μεταξύ της SOAP (XML) αναπαράστασης και της αναπαράστασης της εφαρμογής (όπως μια κλάση της Java για ένα εισιτήριο), και
- που κρατούνται τα διάφορα μέρη RPC (ταυτότητα αντικειμένου, όνομα λειτουργίας και παράμετροι).

Το XML Schema παρέχει μια τυποποιημένη γλώσσα για τον καθορισμό της δομής των εγγράφων και τους τύπους δεδομένων των δομών XML. Δηλαδή δεδομένου ενός τύπου primitive όπως ένας ακεραίου αριθμού ή ενός σύνθετου τύπου, όπως ένα αρχείο με δύο πεδία (παραδείγματος χάριν ένας ακέραιος αριθμός και μια συμβολοσειρά), το XML Schema προσφέρει έναν τυποποιημένο τρόπο για την εγγραφή του τύπου σε XML. Για να επιτραπεί η μετάδοση από τις typped τιμές, το SOAP υποθέτει ένα σύστημα τύπων βασισμένο σε αυτό το XML Schema και καθορίζει την κανονική κωδικοποίησή του σε XML. Χρησιμοποιώντας αυτόν τον τρόπο κωδικοποίησης, μπορεί να παραχθεί μια κωδικοποίηση XML για οποιουσδήποτε τύπους δομημένων δεδομένων. Τα επιχειρήματα RPC καθώς και οι απαντήσεις αντιπροσωπεύονται επίσης κάνοντας χρήση αυτής της κωδικοποίησης.

Έστω τώρα ο ταξιδιώτης του παραδείγματος θέλει να μάθει εάν η πτήση του έχει καθυστέρηση. Γνωρίζει ότι η υπηρεσία Διαδικτύου έχει μια συνάρτηση *GetFlightInfo* που παίρνει δύο ορίσματα :

- μια συμβολοσειρά που περιέχει το όνομα αερογραμμών και
- έναν ακέραιο αριθμό με τον αριθμό πτήσης

και επιστρέφει μια δομημένη τιμή (ένα αρχείο) με δύο πεδία :

- τον αριθμό πύλης και
- την κατάσταση της πτήσης.

Σε αυτήν την περίπτωση, ο ταξιδιώτης μπορεί να ανακτήσει την κατάσταση της πτήσης με την αποστολή ενός HTTP POST στην υπηρεσία φέροντας ένα SOAP <Envelope> όπως φαίνεται στον πίνακα 2.

```
POST /travelservice HTTP/1.0
Host: xxx.xxx.xxx.xxx
Content-Type: text/xml; charset=utf-8
Content-Length: n
SOAPAction: "urn:travelservice-flightinfo"
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
        xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/1999/XMLSchema"
   xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance">
      <SOAP-ENV:Body>
           <m:GetFlightInfo
                xmlns:m="http://www.acme-travel.com/flightinfo">
                SOAP-ENV:encodingStyle=

"http://schemas.xmlsoap.org/soap/encoding/">
                <airlineName
xsi:type="xsd:string">UL</airlineName>
                <flightNumber
xsi:type="xsd:int">506</flightNumber>
           </m:GetFlightInfo>
       </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Πίνακας 2: Παράδειγμα κλήσης SOAP RPC

Σε αυτό το SOAP <Envelope>, η κλήση της *GetFlightInfo* είναι ένα στοιχείο XML με ιδιότητες που περιλαμβάνουν πληροφορίες για την κωδικοποίηση. Τα στοιχεία-παιδιά είναι τα ορίσματα της κλήσης μεθόδου: *airlineName* και *flightNumber*. Οι τύποι τους καθορίζονται στην ιδιότητα *type*, όπου το "xsd" αναφέρεται στους ορισμούς σχημάτων XML. Όταν η εφαρμογή SOAP λαμβάνει το μήνυμα, μετατρέπει το κείμενο XML για τα *airlineName* (UL) και *flightNumber* (506) στην κατάλληλη συμβολοσειρά και τον κατάλληλο ακέραιο αριθμό βασισμένα στην εφαρμογή της υπηρεσίας. Έπειτα καλείται η μέθοδος *GetFlightInfo* με εκείνες τις παραμέτρους.

Ο πίνακας 3 παρουσιάζει την απάντηση σε αυτό το αίτημα. Σε αυτήν την περίπτωση, η απάντηση περιέχει μια δομημένη τιμή με τις επιμέρους τιμές για τον αριθμό πύλης και την κατάσταση της πτήσης.

Οι εφαρμογές του SOAP υπάρχουν για διάφορες γλώσσες προγραμματισμού, συμπεριλαμβανομένης της Java και της C, οι όποιες παράγουν αυτόματα και επεξεργάζονται τα μηνύματα SOAP. Υποθέτοντας ότι τα μηνύματα προσαρμόζονται στην προδιαγραφή SOAP, τότε αυτά μπορούν να ανταλλαχθούν από τις υπηρεσίες που εφαρμόζονται μέσα σε διάφορες γλώσσες.

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: n

<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
                    xmlns:SOAP-
```

```
ENV="http://schemas.xmlsoap.org/soap/envelope/"
                    xmlns:xsd="http://www.w3.org/1999/XMLSchema"
                    xmlns:xsi="http://www.w3.org/1999/XMLSchema-
instance">
     <SOAP-ENV:Body>
         <m:GetFlightInfoResponse
               xmlns:m="http://www.acme-travel.com/flightinfo">
               SOAP-ENV:encodingStyle=

"http://schemas.xmlsoap.org/soap/encoding/">
             <fligh Info>
                <gate xsi:type="xsd:int">10</gate>
                <status xsi:type="xsd:string">ON TIME</status>
            </flightInfo>
         </m:GetFlightInfoResponse>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Πίνακας 3: Παράδειγμα απάντησης SOAP RPC

1.2. Περιγραφή: WSDL

Για τις υπηρεσίες Διαδικτύου, το SOAP προσφέρει τη βασική επικοινωνία, αλλά δεν ενημερώνει για τα στοιχεία που πρέπει να ανταλλάσσουν τα μηνύματα ώστε να αλληλεπιδρούν επιτυχώς με μια υπηρεσία. Αυτό το ρόλο έρχεται να εξυπηρετήσει η WSDL, ένα XML Schema για την περιγραφή των Υπηρεσιών Διαδικτύου ως συλλογές των endpoints επικοινωνίας που μπορούν να ανταλλάξουν ορισμένα μηνύματα.

Για τους προγραμματιστές και τους χρήστες, η WSDL παρέχει μια τυποποιημένη περιγραφή της αλληλεπίδρασης πελάτη-υπηρεσίας. Κατά τη διάρκεια της ανάπτυξης, οι υπεύθυνοι προγραμματιστές χρησιμοποιούν τα έγγραφα WSDL ως την είσοδο σε ένα εργαλείο δημιουργίας proxy που παράγει τον κώδικα πελατών σύμφωνα με τις απαιτήσεις των υπηρεσιών. Παραδείγματος χάριν, οι χρήστες υπηρεσιών ταξιδιού πρέπει να λαμβάνουν μόνο την περιγραφή WSDL εκείνης της υπηρεσίας και να την χρησιμοποιούν ως είσοδο στην υποδομή της σχεδίασης και του χρόνου εκτέλεσης ώστε να ανταλλάσσουν το σωστό τύπο μηνύματος SOAP με την υπηρεσία.

Αυτή η υποενότητα παρέχει τα παραδείγματα των εγγράφων WSDL που περιγράφουν την ταξιδιωτική υπηρεσία Διαδικτύου, η οποία μπορεί να επεξεργαστεί τους δύο τύπους αλληλεπιδράσεων από τα ανωτέρω παραδείγματα SOAP.

Η πρώτη αλληλεπίδραση, *GetFlightInfo*, προσεγγίζεται χρησιμοποιώντας το πρότυπο SOAP RPC. Παίρνει ένα όνομα αερογραμμών και έναν αριθμό πτήσης και επιστρέφει έναν σύνθετο (ή δομημένο) τύπο με πληροφορίες για την πτήση. Η δεύτερη αλληλεπίδραση, *CheckIn*, χρησιμοποιεί το καθαρό μήνυμα SOAP. Αναμένει να λάβει μια XML αντιπροσώπευση ενός ηλεκτρονικού εισιτηρίου και δεν επιστρέφει καμία πληροφορία.

Μια πλήρης περιγραφή υπηρεσιών WSDL παρέχει δύο τύπους πληροφοριών:

- μια περιγραφή υπηρεσιών επιπέδων εφαρμογής, ή αφηρημένη διεπαφή και
- συγκεκριμένες λεπτομέρειες που εξαρτώνται από πρωτόκολλα και που οι χρήστες πρέπει να ακολουθήσουν για να έχουν πρόσβαση στην υπηρεσία σε ένα συγκεκριμένο endpoint υπηρεσιών.

1.2.1. Αφηρημένη περιγραφή

Η WSDL καθορίζει την αφηρημένη περιγραφή υπηρεσιών από την άποψη των μηνυμάτων που ανταλλάσσονται σε μια αλληλεπίδραση υπηρεσιών. Υπάρχουν τρία κύρια συστατικά αυτής της αφηρημένης διεπαφής: το λεξιλόγιο, το μήνυμα και η αλληλεπίδραση. Η συμφωνία για ένα λεξιλόγιο είναι η βάση οποιουδήποτε τύπου επικοινωνίας. Η WSDL χρησιμοποιεί εξωτερικά συστήματα τύπων για να παρέχει τους ορισμούς τύπων δεδομένων για την ανταλλαγή πληροφοριών.

Για το παράδειγμα υπηρεσίας ταξιδιού, ο πίνακας 4 παρουσιάζει δύο τύπους δεδομένων (*string* και *int*) και άλλους δύο που καθορίζονται στο εξωτερικό σχήμα (*FlightInfoType* και *TicketType*). Η WSDL μπορεί να εισαγάγει τέτοιους εξωτερικούς ορισμούς χρησιμοποιώντας ένα στοιχείο *"import"* διευκρινίζοντας τη θέση τους.

```
<wsdl:message name="GetFlightInfoInput">
     <part name="airlineName" type="xsd:string"/>
     <part name="flightNumber" type="xsd:int"/>
</wsdl:message>

<wsdl:message name="GetFlightInfoOutput">
     <part name="flightInfo"
type="flightinfoxsd:FlightInfoType"/>
</wsdl:message>

<wsdl:message name="CheckInInput">
     <part name="eTicket" element="eticketxsd:TicketType"/>
</wsdl:message>

<wsdl:portType name="AirportService_PortType">
     <wsdl:operation name="GetFlightInfo">
          <wsdl:input message="tns:GetFlightInfoInput"/>
          <wsdl:output message="tns:GetFlightInfoOutput"/>
     </wsdl:operation>
     <wsdl:operation name="CheckIn">
          <wsdl:input message="tns:CheckInInput"/>
     </wsdl:operation>
</wsdl:portType>
```

Πίνακας 4: Παράδειγμα αφηρημένης περιγραφής WSDL

Η WSDL καθορίζει πολλαπλα στοιχεία <message>, κάθε ένα από τα οποία περιγράφεται από τύπους XSD ή από στοιχεία από ένα προκαθορισμένο λεξιλόγιο. Τα μηνύματα παρέχουν μια περίληψη και καθορισμό δεδομένων που στέλνεται από και προς τις υπηρεσίες.

Το παράδειγμα στον πίνακα 4 παρουσιάζει τρία μηνύματα που μπορεί να εμφανιστούν κατά τη διάρκεια μιας αλληλεπίδρασης Υπηρεσιών Διαδικτύου. Το μήνυμα GetFlightInfoInput έχει δύο μέρη: *airlineName*, το οποίο είναι μια συμβολοσειρά και *flightNumber*, το οποίο είναι ένας ακέραιος αριθμός. Τα άλλα δύο μηνύματα, GetFlightInfoOutput και CheckInInput, έχουν μόνο από ένα μέρος το καθένα.

Το στοιχείο <PortType> και τα υποστοιχεία του <operation> συνδυάζουν τα μηνύματα ώστε να καθορίζουν αλληλεπιδράσεις. Κάθε λειτουργία αντιπροσωπεύει ένα σχέδιο ανταλλαγής μηνυμάτων που υποστηρίζει η υπηρεσία Διαδικτύου, επιτρέποντας την πρόσβαση των χρηστών σε ένα ορισμένο βασικό κομμάτι της λειτουργίας υπηρεσιών. Μια λειτουργία είναι απλά ένας συνδυασμός μηνυμάτων με ετικέτες *input*, *output*, ή *fault* για να προσδιορίσει ποιο μέρος παίζει ένα ιδιαίτερο μήνυμα στην αλληλεπίδραση.

Ένα <portType> είναι μια συλλογή από διαδικασίες που υποστηρίζονται συλλογικά από ένα endpoint. Στο παράδειγμά μας, το AirportService_PortType περιγράφει δύο διαδικασίες: μια ενιαία αίτησης/απάντησης λειτουργία GetFlightInfo, η οποία αναμένει το μήνυμα GetFlightInfoInput ως είσοδο και επιστρέφει ένα μήνυμα GetFlightInfoOutput ως απάντηση καθώς και μια μονόδρομη λειτουργία CheckIn, η οποία παίρνει ακριβώς το μήνυμα CheckInInput.

1.2.2. Συγκεκριμένες δεσμευτικές πληροφορίες

Μέχρι τώρα, όλα τα στοιχεία στα οποία έχει γίνει αναφορά περιγράφουν τη λειτουργία επιπέδου εφαρμογών της υπηρεσίας. Για να ολοκληρωθεί η περιγραφή μιας αλληλεπίδρασης πελάτη-υπηρεσίας, απαιτούνται τρία ακόμη κομμάτια πληροφορίας:

- *ποιο* πρωτόκολλο επικοινωνίας πρέπει να χρησιμοποιηθεί (όπως SOAP πάνω από HTTP),
- *πώς* να ολοκληρωθούν οι μεμονωμένες αλληλεπιδράσεις υπηρεσιών πέρα από αυτό το πρωτόκολλο, και
- *πού* να ολοκληρωθεί η επικοινωνία (η διεύθυνση δικτύου).

Το στοιχείο <binding> της WSDL παρέχει το "ποιο" και "πώς" αυτών των πληροφοριών, συμπεριλαμβανομένου του πρωτοκόλλου επικοινωνίας και της προδιαγραφής της μορφής δεδομένων για ένα πλήρες <portType>. Το στοιχείο <binding> λέει πώς μια δεδομένη αλληλεπίδραση εμφανίζεται πάνω από το διευκρινισμένο πρωτόκολλο.

Ο πίνακας 5 παρουσιάζει ένα τμήμα WSDL για το παράδειγμα ταξιδιού. Η σύνδεση περιγράφει πώς να χρησιμοποιείται το SOAP για να έχει πρόσβαση στην υπηρεσία Διαδικτύου travelservice.

```
<wsdl:binding name="AirportService_SoapBinding"
                      type="tns:AirportService_PortType">
     <soap:binding
transport="http://schemas.xmlsoap.org/soap/http"/>
     <wsdl:operation name="GetFlightInfo">
        <soap:operation style="rpc"soapAction="urn:travelservice-
flightinfo"/>
        <wsdl:input> <soap:body encodingStyle=

"http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:travelservice-flightinfo"  use="encoded"/>
        </wsdl:input>
        <wsdl:output>
             <soap:body encodingStyle=

"http://schemas.xmlsoap.org/soap/encoding/"
  namespace="urn:travelservice-flightinfo" use="encoded"/>
        </wsdl:output>
     </wsdl:operation>
     <wsdl:operation name="CheckIn">
         <soap:operation style="document"
soapAction="urn:travelservice-checkin"/>
        <wsdl:input>
             <soap:body encodingStyle=

"http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:travelservice-checkin"   use="encoded"/>
        </wsdl:input>
     </wsdl:operation>
```

```
</wsdl:binding>

<wsdl:service name="travelservice">
     <wsdl:port name="travelservice_Port"
binding="tns:AirportService_SoapBinding">
<soap:address location=
"http://www.acmetravel.com/travelservice"/>
     </wsdl:port>
</wsdl:service>
```

Πίνακας 5: Παράδειγμα συγκεκριμένης δεσμευτικής πληροφορίας WSDL

Ειδικότερα, το έγγραφο WSDL δείχνει ότι:

- Η GetFlightInfo θα είναι μια αλληλεπίδραση του τύπου SOAP-RPC, στην οποία όλες οι ανταλλαγές μηνυμάτων κάνουν χρήση της τυποποιημένης κωδικοποίησης SOAP, και
- Η CheckIn είναι μια καθαρή αλληλεπίδραση μηνύματος (αποκαλούμενη "document-oriented" σε όρους WSDL), μέσα στην οποία το σώμα του μηνύματος SOAP περιέχει το κωδικοποιημένο μήνυμα χωρίς πρόσθετες κωδικοποιήσεις τύπων.

Αυτό που απομένει τώρα είναι να καθοριστεί το "που" για να προσπελαύνεται αυτός ο συνδυασμός της αφηρημένης διεπαφής, του πρωτοκόλλου και λεπτομερειών τακτοποίησης δεδομένων (το "binding"). Ένα στοιχείο WSDL <port> περιγράφει ένα ενιαίο endpoint ως τον συνδυασμό μιας σύνδεσης και μιας διεύθυνσης δικτύου. Συνεπώς, ένα στοιχείο <service> ομαδοποιεί ένα σύνολο σχετικών port.

Στο παράδειγμα υπηρεσιών ταξιδιού, ένα ενιαίο port (travelservice_Port) περιγράφει ένα endpoint που επεξεργάζεται τα αιτήματα SOAP για την υπηρεσία Διαδικτύου travelservice.

1.3. Κατάλογος: UDDI

Οι προδιαγραφές UDDI προσφέρουν στους χρήστες έναν ενοποιημένο και συστηματικό τρόπο για την εύρεση των φορέων παροχής υπηρεσιών μέσω ενός συγκεντρωμένου καταλόγου των υπηρεσιών που είναι κατά προσέγγιση ισοδύναμος με έναν αυτοματοποιημένο on-line "τηλεφωνικό κατάλογο" των Υπηρεσιών Διαδικτύου. Αφ' ενός, υπάρχουν προσιτοί σε διαφυλλιστές, δημόσιοι κατάλογοι UDDI και αφ' ετέρου, υπάρχουν επιχειρήσεις που αρχίζουν να χρησιμοποιούν τους "ιδιωτικούς" καταλόγους UDDI για να ενσωματώνουν και να προσπελαύνουν τις εσωτερικές υπηρεσίες τους.

Το UDDI παρέχει δύο βασικές προδιαγραφές που καθορίζουν τη δομή ενός καταλόγου υπηρεσιών και τη λειτουργία:

- έναν καθορισμό των πληροφοριών που παρέχουν για κάθε υπηρεσία και πώς αυτή κωδικοποιείται ("Δομή δεδομένων UDDI") και
- ένα API δημοσίευσης και ερωτημάτων για τον κατάλογο που περιγράφει πώς αυτές οι πληροφορίες μπορούν να δημοσιευθούν και να προσπελαστούν
- Η πρόσβαση καταλόγων μπορεί να ολοκληρωθεί χρησιμοποιώντας ένα τυποποιημένο SOAP API και για την έκδοση και για τη συζήτηση.

1.3.1. Οργάνωση της δομής

Το UDDI περιέχει αρχικά μερικές γενικές (μη λειτουργικές) προδιαγραφές, που αντιπροσωπεύουν τις «λευκές σελίδες» μιας Υπηρεσίας Διαδικτύου. Αποτελούνται από το όνομα της υπηρεσίας, μια μη επίσημη περιγραφή, όπως επίσης και ένα (UDDI-specific) μοναδικό κλειδί υπηρεσίας και κατηγοριοποιούνται ως διαχειριστικές πληροφορίες. Επιπλέον, επιτρέπει την ταξινόμηση του τομέα της εφαρμογής (domain) της Υπηρεσίας Διαδικτύου επισυνάπτοντας μια ή περισσότερες προδιαγραφές με βάση την ταξονομία (taxonomy-based). Αυτό θα μπορούσε να

αναφερθεί ως «κίτρινες σελίδες» μιας Υπηρεσίας Διαδικτύου. Δυστυχώς, δεν υπάρχουν συγκεκριμένες τυποποιημένες ταξονομίες για την ταξινόμηση των Υπηρεσιών Διαδικτύου. Αυτό σημαίνει ότι κάποιος πρέπει να χρησιμοποιήσει τις διαθέσιμες ταξονομίες για να κατατάξει τις επιχειρήσεις όταν προδιαγράφει Υπηρεσίες Διαδικτύου. Οι ταξινομήσεις Υπηρεσιών Διαδικτύου ομαδοποιούνται κάτω από την εννοιολογική ετικέτα "category bag".

Τελικά, το πλαίσιο προδιαγραφών Υπηρεσιών Διαδικτύου παρέχει πληροφορίες που μπορούν να χρησιμοποιηθούν για τη διαμόρφωση συγκεκριμένων Υπηρεσιών Διαδικτύου. Επειδή αυτά είναι μάλλον τεχνικές πληροφορίες, αυτές οι προδιαγραφές είναι χωρισμένες και ονομάζονται «πράσινες σελίδες». Μπορούν να βρεθούν κάτω από την εννοιολογική ετικέτα "binding templates"και επικεντρώνονται στην τεκμηρίωση των διεπαφών των Υπηρεσιών Διαδικτύου (παρέχοντας κατά συνέπεια πληροφορίες για τη θέση τους, τις μεθόδους που εκτίθενται από μια υπηρεσία, και τα αναμενόμενα δεδομένα εισόδου). Συνήθως, η τεκμηρίωση της διεπαφής επιτυγχάνεται με την εισαγωγή μιας WSDL προδιαγραφής.

Συνεπώς το UDDI κωδικοποιεί τρεις τύπους πληροφοριών για τις υπηρεσίες Διαδικτύου:

- πληροφορίες ***"λευκών σελίδων"*** που περιλαμβάνουν, παραδείγματος χάριν, λεπτομέρειες ονόματος και επαφών για την επιχείρηση ή την εταιρεία του φορέα παροχής υπηρεσιών,
- πληροφορίες ***"κίτρινων σελίδων"*** που παρέχουν μια κατηγοριοποίηση βασισμένη στους τύπους επιχειρήσεων και υπηρεσιών, και
- πληροφορίες ***"πράσινων σελίδων"*** που περιλαμβάνουν τα τεχνικά στοιχεία για την υπηρεσία.

Ο κατάλογος UDDI οργανώνεται γύρω από δύο θεμελιώδεις οντότητες που περιγράφουν τις επιχειρήσεις και τις υπηρεσίες που παρέχουν.

Το στοιχείο businessEntity που παρουσιάζεται στον πίνακα 6 παρέχει τις τυποποιημένες πληροφορίες άσπρων σελίδων, συμπεριλαμβανομένων των προσδιοριστικών, των στοιχείων επαφής και μιας απλής επιχειρησιακής περιγραφής. Κάθε επιχειρησιακή οντότητα θα μπορούσε να περιλαμβάνει ένα ή περισσότερα στοιχεία businessService, όπως φαίνονται στον πίνακα 6, ο οποίος αντιπροσωπεύει τις υπηρεσίες που παρέχει.

```
<businessEntity businessKey=
        "A687FG00-56NM-EFT1-3456-098765432124">
     <name>Acme Travel Incorporated</name>
     <description xml:lang="en">
         Acme is a world wide leader in online travel services
     </description>
     <contacts>
         <contact useType="US general">
             <personName>Acme Inc.</personName>
             <phone>1 800 CALL ACME</phone>
             <email useType="">acme@acme-travel.com</email>
             <address>
                 <addressLine>Acme</addressLine>
                 <addressLine>12 Maple Avenue</addressLine>
                 <addressLine>Springfield, CT 06785</addressLine>
             </address>
         </contact>
      </contacts>
      <businessServices>
         ...
      </businessSerivices>
```

```
      <identifierBag>
         ...
      </identifierBag>
      <categoryBag>
          <keyedReference tModelKey=
     "UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Electronic check-in"
                                 keyValue="84121801">
     </categoryBag>
</businessEntity>
```

Πίνακας 6: Παράδειγμα δομής businessEntity UDDI

Οι οντότητες επιχειρήσεων και υπηρεσιών μπορούν να διευκρινίσουν ένα categoryBag για να ταξινομήσουν την επιχείρηση ή την υπηρεσία.

Κάθε οντότητα δεδομένων (π.χ. επιχειρήσεις ή υπηρεσίες) προσδιορίζεται από ένα μοναδικό κλειδί. Αυτά τα ορισμένα κλειδιά παράγονται όταν καταχωρείται η οντότητα. Τα κλειδιά είναι εγγυημένα παγκοσμίως μοναδικά προσδιοριστικά (UUIDs). Παραδείγματος χάριν, μια ιδιότητα *businessKey* προσδιορίζει μεμονωμένα μια επιχειρησιακή οντότητα και μια ιδιότητα *serviceKey* προσδιορίζει μια υπηρεσία.

Σε αντίθεση με μια κατανοήσιμη από τον άνθρωπο περιγραφή, όνομα και κατηγοριοποίηση, η οντότητα υπηρεσιών περιέχει έναν κατάλογο από bindingTemplates που κωδικοποιούν τις τεχνικές πληροφορίες πρόσβασης σε υπηρεσία. Κάθε binding template αντιπροσωπεύει ένα σημείο πρόσβασης στην υπηρεσία. Η υπόθεση είναι ότι η ίδια υπηρεσία μπορεί να παρασχεθεί στα διαφορετικά endpoints, κάθε ένα από τα οποία μπορεί να έχει διαφορετικά τεχνικά χαρακτηριστικά.

Τα binding templates επιτρέπουν τις περιγραφές υπηρεσιών χρησιμοποιώντας τις αυθαίρετες εξωτερικές πληροφορίες (δηλ. πληροφορίες που δεν καθορίζονται από το ίδιο το UDDI). Η πληροφορία σε ένα binding template περιέχει το μοναδικό κλειδί του (*bindingKey*), μια παραπομπή στο κλειδί υπηρεσιών και το endpoint όπου μπορεί να προσεγγιστεί η υπηρεσία Διαδικτύου.

```
<businessService serviceKey=
               "894B5100-3AAF-11D5-80DC-002035229C64"
 businessKey="A687FG00-56NM-EFT1-3456-098765432124">
     <name>ElectronicTravelService</name>
     <description xml:lang="en">Electronic Travel
Service</description>
     <bindingTemplates>
        <bindingTemplate bindingKey=
"6D665B10-3AAF-115D-80DC-002035229C64"
        serviceKey= "894B5100-3AAF-11D5-80DC-002035229C64">
           <description>
               SOAP-based e-checkin and flight info
           </description>
           <accessPoint URLType="http">
                http://www.acme-travel.com/travelservice
           </accessPoint>
           <tModelInstanceDetails>
               <tModelInstanceInfo tModelKey=
                  "D2033110-3BGF-1KJH-234C-09873909802">
```

```
                ...
          </tModelInstanceDetails>
        </bindingTemplate>
     </bindingTemplates>
     <categoryBag>
          ...
     </categoryBag>
</businessService>
```

Πίνακας 7: Παράδειγμα δομής businessService UDDI

1.3.2. Τεχνικές περιγραφές και tModels

Μια πολύ ενδιαφέρουσα πληροφορία που καλύπτεται από ένα binding template είναι η tModelInstanceDetails, η οποία παρέχει την τεχνική περιγραφή των υπηρεσιών. Περιέχει έναν κατάλογο αναφορών με τεχνικές προδιαγραφές τις οποίες αλολουθεί η υπηρεσία. Οι φορείς παροχής υπηρεσιών καταγράφουν πρώτα την απαραίτητη τεχνική προδιαγραφή στον κατάλογο UDDI, ο οποίος ορίζει ένα μοναδικό κλειδί προσδιοριστικών (*tModelKey*).

Το UDDI αντιπροσωπεύει κάθε καταχωρημένη τεχνική προδιαγραφή χρησιμοποιώντας την οντότητα πληροφοριών tModel. Τα endpoints υπηρεσιών που υποστηρίζουν την προδιαγραφή μπορούν έπειτα απλά να προσθέσουν την αντίστοιχη αναφορά στον κατάλογο tModelInstanceDetails τους.

Για παράδειγμα, ο κατάλογος UDDI ενός εγγράφου WSDL εξετάζεται τώρα ως tModel. Υποθέτοντας ότι η βιομηχανία ταξιδιού έχει καθορίσει τις τυποποιημένες διεπαφές WSDL καθώς και τις συνδέσεις για ηλεκτρονική καταχώρηση και ανάκτηση των πληροφοριών πτήσης, δημιουργείται πρώτα ένα tModel όπως φαίνεται στον πίνακα 8, ώστε να αντιπροσωπεύσει αυτούς τους ορισμούς WSDL. Τα endpoints υπηρεσιών που εφαρμόζουν αυτές τις διεπαφές μπορούν έπειτα να περιλάβουν το αντίστοιχο tModel στον κατάλογο tModelInstanceDetails.

Η ιδέα πίσω από τον μηχανισμό tModel είναι απλή και ισχυρή: Για να περιγράψει επαρκώς μια υπηρεσία, πρέπει συχνά να παρέχονται πληροφορίες των οποίων ο τύπος ή το σχήμα δεν μπορεί (ή δεν πρέπει) να αναμένεται. Αντικαθιστώντας την πληροφορία με ένα μοναδικό κλειδί παρέχεται μια αναφορά με αυθαίρετους τύπους πληροφοριών.

```
<tModel tModelKey="D2033110-3BGF-1KJH-234C-09873909802">
     <name>http://www.travel.org/e-checkin-interface</name>
     <description xml:lang="en">
   Standard service interface definition for travel services
     </description>
     <overviewDoc>
         <description xml:lang="en">
             WSDL Service Interface Document
         </description>
         <overviewURL>
               http://www.travel.org/services/e-checkin.wsdl
         </overviewURL>
     </overviewDoc>
     <identifierBag>
        ...
     </identifierBag>
     <categoryBag>
        ...
     </categoryBag>
</tModel>
```

Πίνακας 8: Παράδειγμα δομής tModel UDDI

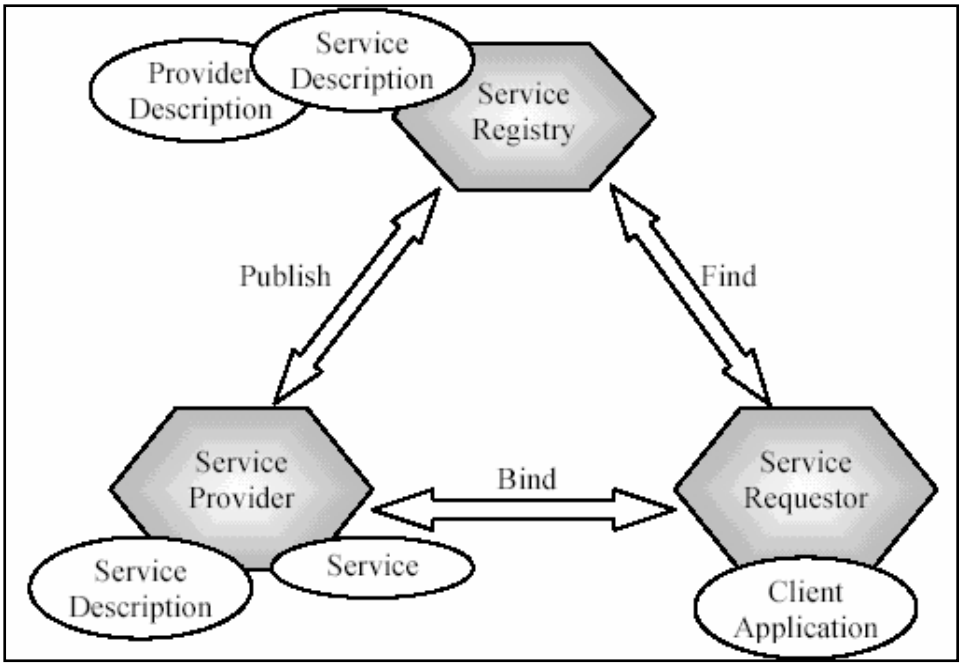
1.3.3. Κατηγοριοποίηση

Ο αποτελεσματικός εντοπισμός των ιδιαίτερων τύπων επιχειρήσεων και υπηρεσιών εξαρτάται από τη δυνατότητα περιγραφής των καταχωρήσεων επιχειρήσεων και υπηρεσιών του καταλόγου UDDI σύμφωνα με ένα σχέδιο κατηγοριοποίησης ή την ταξονομία. Στην πραγματικότητα, σε οποιαδήποτε ρεαλιστική κατάσταση, απαιτούνται πολλαπλά δυαδικά ψηφία πληροφοριών, όπως η γεωγραφική θέση ή ο τύπος βιομηχανίας ή προϊόντος, για τον χαρακτηρισμό των υπηρεσιών που επιδιώκονται.

Για τον προσδιορισμό των συστημάτων ταξονομίας, κάθε σύστημα ταξινόμησης καταχωρείται από μόνο του ως tModel μέσα στον κατάλογο UDDI. Οι πληροφορίες ταξονομίας κωδικοποιούνται έπειτα ανά τα ζευγάρια τιμής ονόματος, περιορισμένα από μια βασική αναφορά tModel που προσδιορίζει σε ποια ταξονομία ανήκει το κάθε ζεύγος.Χρησιμοποιώντας την κατηγοριοποίηση, οι ερωτήσεις στον κατάλογο UDDI μπορούν να εκτελεσθούν για να εντοπίσουν πολύ συγκεκριμένους τύπους υπηρεσιών.

1.4. Ορισμοί Υπηρεσιών Διαδικτύου

Η αρχιτεκτονική Υπηρεσιών Διαδικτύου καλύπτει τρία στοιχεία: τους ρόλους, τις διαδικασίες και τα artefacts. Το σχήμα 1 επεξηγεί αυτά τα στοιχεία. Η αρχιτεκτονική βασίζεται στην αλληλεπίδραση τριών ρόλων: **πάροχος υπηρεσιών** (Service Provider), **κατάλογος υπηρεσιών** (Service Registry) και **πελάτης υπηρεσιών** (Service Requester/Client). Οι αλληλεπιδράσεις περιλαμβάνουν τις διαδικασίες **Δημοσίευσης** (Publish), **Ανακάλυψης** (Find) και **Δέσμευσης** (Bind).Οι ρόλοι αυτοί μαζί με τις διαδικασίες ενεργούν πάνω στην υπομονάδα λογισμικού της Υπηρεσίας Διαδικτύου, στην περιγραφή της (συμπεριλαμβανομένης και της περιγραφής του πάροχου υπηρεσιών) και στην εφαρμογή πελάτη.



Σχήμα 1:Ρόλοι, διαδικασίες και artefacts Υπηρεσιών Διαδικτύου

1.4.1. Ρόλοι στο πρότυπο Υπηρεσιών Διαδικτύου

Οι συμμετέχοντες στο πρότυπο Υπηρεσιών Διαδικτύου είναι οι ακόλουθοι:

Πάροχος Υπηρεσιών (Service Provider):

- από την επιχειρησιακή άποψη, αυτός είναι ο ιδιοκτήτης της υπηρεσίας.
- από την αρχιτεκτονική άποψη, είναι η πλατφόρμα που παρέχει την πρόσβαση στην υπηρεσία.

Πελάτης Υπηρεσιών (Service Requester):

- από την επιχειρησιακή άποψη, είναι η επιχείρηση που απαιτεί ορισμένες λειτουργίες που είναι καλυμμένες από την αντίστοιχη υπηρεσία.
- από την αρχιτεκτονική άποψη, είναι η εφαρμογή πελατών που ψάχνει και κατόπιν επιδρά στην υπηρεσία.

Κατάλογος Υπηρεσιών (Service Registry):

- από την επιχειρησιακή άποψη, είναι ο ιδιοκτήτης μιας υπηρεσίας καταλόγων.
- από την αρχιτεκτονική άποψη, είναι μια πλατφόρμα που παρέχει την πρόσβαση σε καταχωρημένες πληροφορίες υπηρεσιών.

Ο κατάλογος υπηρεσιών είναι μια εξερευνήσιμη αποθήκη περιγραφών υπηρεσιών. Οι πάροχοι υπηρεσιών μπορούν να δημοσιεύσουν τις περιγραφές υπηρεσιών τους, συμπεριλαμβανομένης μιας περιγραφής της επιχείρησης τους. Οι πελάτες υπηρεσιών μπορούν να βρουν τις υπηρεσίες και να λάβουν τις binding πληροφορίες (από τις περιγραφές της υπηρεσίας). Αυτές οι πληροφορίες χρησιμοποιούνται κατά τη διάρκεια της ανάπτυξης για στατική σύνδεση ή κατά τη διάρκεια εκτέλεσης για δυναμική σύνδεση. Οι δυναμικά συνδεδεμένοι πελάτες υπηρεσιών έχουν πρόσβαση στον κατάλογο υπηρεσιών σε κάθε εκτέλεση. Για τους στατικά συνδεδεμένους πελάτες υπηρεσιών, η πρόσβαση στον κατάλογο υπηρεσιών εκτελείται μόνο μία φορά ώστε να ανακτηθούν οι πληροφορίες. Στη στατική σύνδεση, ο ρόλος του καταλόγου υπηρεσιών είναι προαιρετικός. Ο πελάτης δηλαδή θα μπορούσε να λάβει την περιγραφή υπηρεσιών άμεσα από τον πάροχο ή από άλλες πηγές εκτός του καταλόγου υπηρεσιών (π.χ. περιοχή WWW ή περιοχή FTP).

1.4.2. Διαδικασίες στο πρότυπο υπηρεσιών Διαδικτύου

Το πρότυπο Υπηρεσιών Διαδικτύου περιλαμβάνει τις ακόλουθες διαδικασίες:

- **Δημοσίευση** (Publish): Για να έχει τη δυνατότητα ένας πελάτης να βρει μια Υπηρεσία Διαδικτύου και να έχει πρόσβαση σε αυτήν, ο πάροχος θα πρέπει να τη δημοσιεύσει σε έναν κατάλογο υπηρεσιών.
- **Εύρεση** (Find): Στη λειτουργία εύρεσης, ο πελάτης ανακτά μια περιγραφή υπηρεσιών εξετάζοντας τον κατάλογο υπηρεσιών. Η λειτουργία εύρεσης μπορεί να εμφανιστεί σε δύο διαφορετικές φάσεις του κύκλου ζωής για τον πελάτη: στο χρόνο σχεδιασμού, για να ανακτηθεί η περιγραφή διεπαφής υπηρεσιών για την ανάπτυξη εφαρμογής πελατών (στατική σύνδεση), ή σε χρόνο εκτέλεσης, που παίρνει τη σύνδεση υπηρεσιών και την περιγραφή της θέσης για την κλήση (δυναμική σύνδεση).
- **Δέσμευση** (Binding): Στη λειτουργία δέσμευσης, ο πελάτης επικαλείται την υπηρεσία σε χρόνο εκτέλεσης χρησιμοποιώντας τις δεσμευτικές λεπτομέρειες της περιγραφή υπηρεσιών ώστε να εντοπίσει την υπηρεσία, να έρθει σε επαφή μαζί της και την καλέσει.

1.4.3. Artefacts μιας Υπηρεσίας Διαδικτύου

Τα artefacts που παράγονται και εξετάζονται μέσα στο πλαίσιο των Υπηρεσιών Διαδικτύου είναι τα ακόλουθα:

- **Υπηρεσία** (Service): Είναι η υλοποίηση ενός υποσυστήματος λογισμικού που αναπτύσσεται σε μια πλατφόρμα προσπελάσιμη από το διαδίκτυο, η οποία παρέχεται από τον πάροχο και που καλείται από έναν πελάτη.

- **Περιγραφή Υπηρεσιών** (Service Description): Η περιγραφή υπηρεσιών περιέχει τις λεπτομέρειες της διεπαφής και την υλοποίηση της υπηρεσίας. Η περιγραφή διεπαφών υπηρεσιών περιλαμβάνει τις πληροφορίες για διαδικασίες που παρέχονται από μια υπηρεσία, καθώς επίσης και τις παραμέτρους τους. Μπορεί να συγκριθεί με την υπογραφή μιας μεθόδου. Επιπλέον, περιγράφεται το πρωτόκολλο που χρησιμοποιείται για την επικοινωνία με την Υπηρεσία Διαδικτύου. Η περιγραφή εφαρμογής υπηρεσιών περιέχει πληροφορίες για τη θέση όπου εκτίθεται η υπηρεσία, δηλ. τη δικτυακή διεύθυνση του endpoint που παρέχει την υπηρεσία. Η πλήρης περιγραφή υπηρεσιών δημοσιεύεται σε έναν κατάλογο υπηρεσιών από τον πάροχο για να καταστήσει την υπηρεσία προσιτή στους πελάτες. Περιλαμβάνει τους τύπους δεδομένων της υπηρεσίας, τις διαδικασίες, τις δεσμευτικές πληροφορίες και την τοποθεσία δικτύου, όπως προβλέπεται από τις διεπαφές υπηρεσιών και τις περιγραφές της εφαρμογής. Θα μπορούσε επίσης να περιλαμβάνει πληροφορίες για τον πάροχο, την κατηγοριοποίηση του φορέα και της υπηρεσίας και άλλα μετα-δεδομένα που διευκολύνουν την ανακάλυψη και τη χρήση από τον πελάτη.
- **Εφαρμογή Πελάτη** (Client Application): Είναι η εφαρμογή που εκτελείται από τον πελάτη ο οποίος χρησιμοποιεί τη λειτουργία της Υπηρεσίας Διαδικτύου και την καλεί σε χρόνο εκτέλεσης.

2. Προδιαγραφές Υπηρεσιών Διαδικτύου με βελτιώσεις του UDDI

Το UDDI περιέχει δύο τυποποιημένα πλαίσια προδιαγραφών [5] που διακρίνονται μεταξύ τους ως εξής: ένα πλαίσιο για τις προδιαγραφές των επιχειρήσεων (που μπορεί να χρησιμοποιηθεί για να χτίσει έναν επιχειρηματικό κατάλογο) και ένα πλαίσιο για τον καθορισμό των Υπηρεσιών Διαδικτύου (που είναι η βάση για την υλοποίηση των καταλόγων Υπηρεσιών Διαδικτύου). Τα δύο πλαίσια συνδέονται μεταξύ τους σε ποσοστό N:M, δηλαδή μια επιχείρηση μπορεί να προσφέρει πολλαπλές Υπηρεσίες Διαδικτύου και μια Υπηρεσία Διαδικτύου μπορεί πιθανότατα να παρέχεται από διαφορετικές επιχειρήσεις (συμπεριλαμβανομένης της πιθανότητας αναφοράς σε Υπηρεσίες Διαδικτύου που παρέχονται από άλλες επιχειρήσεις).

Το πλαίσιο επιχειρησιακών προδιαγραφών, που είναι ένα υψηλού επιπέδου στοιχείο στο αντίστοιχο UDDI μοντέλο δεδομένων, περιέχει πληροφορίες σχετικά με επιχειρήσεις που μπορούν να διαιρεθούν σε διαφορετικές κατηγορίες. Αναφέρεται με το εννοιολογικό όνομα «επιχειρησιακή οντότητα» (business entity). Καθώς το UDDI εξελίχθηκε στην έκδοση 3.0, το πλαίσιο επιχειρησιακών προδιαγραφών βελτιώθηκε για να υποστηρίξει πολύπλοκους οργανισμούς (που αποτελούνται από επιχειρησιακές μονάδες, τμήματα κλπ).

Κατά την ανάλυση της δυνατότητας εφαρμογής του UDDI σαν metaschema για την ανάπτυξη καταλόγων Υπηρεσιών Διαδικτύου ο κακώς καθορισμένος διαχωρισμός του επιχειρησιακού και του πλαισίου προδιαγραφών Υπηρεσιών Διαδικτύου αποδεικνύεται μια αδυναμία στο εννοιολογικό σχέδιο. Ειδικότερα, θα έπρεπε να αναθεωρηθεί η επίσημη θεματική ομαδοποίηση των προδιαγραφών, που ενώνει και τα δύο πλαίσια και αγνοεί το γεγονός ότι η προδιαγραφή των Υπηρεσιών Διαδικτύου χρησιμοποιεί επίσης τις λευκές και τις κίτρινες σελίδες.

Επιπλέον, το UDDI πλαίσιο προδιαγραφών Υπηρεσιών Διαδικτύου έχει αδυναμίες σχετικά με τις γενικές απαιτήσεις των πλαισίων προδιαγραφών. Αυτό οδηγεί στην πρόταση ορισμένων βελτιώσεων. Παρόλο που δεν υλοποιεί ρητά μια αντικειμενοστραφή δομή είναι δυνατό να εισάγει ορισμένες πτυχές των προδιαγραφών με βάση την προηγούμενη περιγραφή τους, δεδομένου ότι επικεντρώνεται σε διαχειριστικές προδιαγραφές, ταξινομήσεις της περιοχής, και πληροφορίες για τη διεπαφή της Υπηρεσίας. Επιπλέον, η θεματική ομαδοποίηση σε άσπρες, κίτρινες, και πράσινες σελίδες θα έπρεπε να εφαρμοστεί στο πλαίσιο προδιαγραφών Υπηρεσιών Διαδικτύου για να βελτιώσει την αναγνωσιμότητα. Λαμβάνοντας και τα δύο ως δεδομένα, το πλαίσιο προδιαγραφών Υπηρεσιών Διαδικτύου εκπληρώνει αυτές τις απαιτήσεις υλοποίησης μιας αντικειμενοστραφούς δομής.

Το πλαίσιο UDDI επιτρέπει κυρίως τον καθορισμό διαφορετικών (εννοιολογικών) τύπων Υπηρεσιών Διαδικτύου και επιπλέον υποστηρίζει την ταξινόμησή τους προκειμένου να τις διαχωρίζει. Εντούτοις, αυτό απαιτεί την εισαγωγή μιας εξειδικευμένης ταξονομίας που βασίζεται σε περιοχές (domains), σε αντίθεση με τις υπάρχουσες ταξονομίες που έχουν μάλλον ως σκοπό να ταξινομήσουν τους κλάδους της βιομηχανίας. Επιτρέπει την κατηγοριοποίηση των Υπηρεσιών Διαδικτύου που προσφέρουν γενικές υπηρεσίες (π.χ. κρυπτογράφηση δεδομένων ή αποστολή ηλεκτρονικών μηνυμάτων) και αυτών που προσφέρουν εξειδικευμένες υπηρεσίες (π.χ. ηλεκτρονική διακυβέρνηση).

Αν και το UDDI πλαίσιο προδιαγραφών Υπηρεσιών Διαδικτύου καλύπτει διαφορετικές πτυχές, δύσκολα χαρακτηρίζεται ως μια πλήρη προδιαγραφή Υπηρεσιών Διαδικτύου. Προκειμένου να υποστηρίξει την ανακάλυψη και τη διαμόρφωση των Υπηρεσιών Διαδικτύου, που είναι οι βασικοί τομείς εφαρμογής του, απαιτούνται επιπρόσθετες προδιαγραφές που επιτρέπουν την αξιολόγηση πιθανών (σημασιολογικών και πραγματικών) ετερογενειών μεταξύ Υπηρεσιών Διαδικτύου όπως επίσης και προδιαγραφές σχετικά με το κόστος. Για αυτό, το πλαίσιο UDDI θα έπρεπε να επεκταθεί ώστε να περιέχει πιο πλήρεις προδιαγραφές των Υπηρεσιών Διαδικτύου. Σε αυτό το κεφάλαιο παρουσιάζονται διάφορες βελτιώσεις του πλαισίου UDDI με βάση την εργασία [6].

2.1. E-UDDI: Βελτιώσεις προς μια πλήρη Προδιαγραφή

Στη συνέχεια χρησιμοποιείται ένα πρότυπο πλαίσιο προδιαγραφών που σχεδιάστηκε από μια διεπιστημονική ομάδα τυποποίησης της Γερμανικής Κοινωνίας της Πληροφορικής για να επεκτείνει το UDDI πλαίσιο προδιαγραφών Υπηρεσιών Διαδικτύου, διατηρώντας προς τα πίσω συμβατότητα με το πρότυπο UDDI. Το πλαίσιο που προκύπτει καλείται «Επεκταμένο UDDI» (Extended UDDI-E-UDDI), επειδή οι υλοποιήσεις με βάση το βελτιωμένο πλαίσιο προδιαγραφών υποστηρίζουν τη διαλειτουργικότητα με τις υλοποιήσεις που βασίζονται στο τυποποιημένο UDDI.

2.2. Ένα πρότυπο Πλαίσιο Προδιαγραφών

Το προτεινόμενο πλαίσιο προδιαγραφών αρχικά στόχευε να καθιερώσει ένα συστηματικό πρότυπο για την περιγραφή της εξωτερικής μορφής των επιχειρησιακών τμημάτων που μπορούν να υιοθετηθούν για την καθορισμό των Υπηρεσιών Διαδικτύου. Τεκμηριώνει ικανοποιητικά την εξωτερική μορφή, που σημαίνει ότι περιγράφει τις υπηρεσίες που παρέχονται από components όπως επίσης και τις συνθήκες (γενικές, τεχνικές, σημασιολογικές και πραγματολογικές) που εφαρμόζονται όταν καλείται μια component υπηρεσία. Ο όρος «συστηματικό» υπονοεί ότι το πρότυπο όχι μόνο περιγράφει τι πρέπει να καθοριστεί άλλα επίσης τον τρόπο που θα καθοριστεί, συμπεριλαμβανομένων των γραφών (notations) που πρέπει να χρησιμοποιηθούν κατά τη διαδικασίας της προδιαγραφής. Εκτός από τις γενικές και μη λειτουργικές προδιαγραφές, με αυτόν τον τρόπο προτιμά τη χρήση τυποποιημένων γραφών (π.χ. προγραμματιστικών ή μαθηματικών γλωσσών) παρά τη χρήση κοινών είτε γραφικών γλωσσών. Παρόλα αυτά, το σχεδιασμένο πλαίσιο εστιάζει ιδιαίτερα στην υποστήριξη αυτοματοποιημένης ανακάλυψης και διαμόρφωσης component, δίνοντας επομένως έμφαση σε επίσημες γραφές.

Το προτεινόμενο πλαίσιο προδιαγραφών υλοποιεί μια αντικειμενοστραφή δομή με σκοπό να βελτιώσει την αναγνωσιμότητα και να συντονίσει καλύτερα την κατάλληλη χρήση (εξειδικευμένων) γραφών. Αποτελείται από εννιά πλευρές: «διαχειριστικές πληροφορίες», «απόδοση», «ασφάλεια», «περιοχή», «συντονισμός», «συμπεριφορά», «διεπαφή», «ορολογία», και «στόχος».

Οι πρώτες τέσσερις πλευρές περιέχουν γενικές, μη λειτουργικές προδιαγραφές είτε κατηγοριοποιήσεις αντίστοιχα. Οι τρεις πτυχές που εμφανίζονται στη μέση εστιάζουν στον καθορισμό τεχνικών πληροφοριών που απαιτούνται για τη διαδικασία της διαμόρφωσης, και ειδικά για την αξιολόγηση και τη διαχείριση ετερογενειών. Επομένως, προδιαγραφές που αναφέρονται στη συμπεριφορά των components και της διεπαφής εστιάζουν στην τεκμηρίωση συντακτικών συνθηκών. Οι προδιαγραφές για το συντονισμό τεκμηριώνουν την διατεταγμένη κλήση μεθόδων των component. Τέλος, οι δυο τελευταίες πτυχές στρέφονται σε προδιαγραφές σχετικά με την υλοποίηση της σημασιολογίας και των διαδικασιών (pragmatics). Οι σημασιολογικές περιγραφές τεκμηριώνουν όρους, έννοιες και σχέσεις μεταξύ τους και αποτελούν τη βάση για την ανάπτυξη μεταφραστών εάν εμφανιστούν σημασιολογικές ανομοιομορφίες κατά τη διαμόρφωση. Οι προδιαγραφές που αναφέρονται στην υλοποιημένη διαδικασία τεκμηριώνουν τους εννοιολογικούς στόχους που υποστηρίζονται από τα αντίστοιχα component. Μπορούν να χρησιμοποιηθούν για να καθιερώσουν κοινά workflows μεταξύ components, μετριάζοντας κατά συνέπεια τις πραγματολογικές ετερογένειες.

Η αντικειμενοστραφής δομή του πλαισίου μπορεί να αντιστοιχιστεί στη θεματική ομαδοποίηση του UDDI που είναι αρκετά συμβατή. Εφόσον οι Υπηρεσίες Διαδικτύου μπορούν να χαρακτηριστούν ως ειδικά (διαλειτουργικά) components, τα επιτεύγματα του πλαισίου προδιαγραφών που παρουσιάζεται εδώ συμβάλλουν άμεσα στον καθορισμό των Υπηρεσιών Διαδικτύου.

2.2.1. Εφαρμογή των Βελτιώσεων σε Λευκές και Κίτρινες Σελίδες

Το πλαίσιο UDDI περιέχει ήδη μερικές γενικές προδιαγραφές που καλούνται «διαχειριστικές πληροφορίες» και σχετίζονται με τις λευκές σελίδες. Προκειμένου να επιτευχθεί μια πληρέστερη προδιαγραφή, οι πληροφορίες που απαιτούνται για την προμήθεια των Υπηρεσιών Διαδικτύου θα έπρεπε να προστεθούν στις διαχειριστικές πληροφορίες. Αυτές οι πληροφορίες αποτελούνται από τις προδιαγραφές σχετικά με την άδεια χρήσης και τις μορφές διανομής (που αποτελείται από τις αντίστοιχες μεθόδους τιμολόγησης και υποστηριζόμενης πληρωμής). Δεδομένου ότι αυτές οι προδιαγραφές δεν είναι λειτουργικές εφαρμόζεται η χρήση κοινών γλωσσών, καθοδηγημένες από μια ειδική ταξονομία μεθόδων πληρωμής (που παρέχει διάφορες εναλλακτικές λύσεις).

Εκτός από αυτές τις διαχειριστικές πληροφορίες, η προδιαγραφή Υπηρεσιών Διαδικτύου προσαρμόζεται από προδιαγραφές που αναφέρονται στην απόδοσή της και την ασφάλεια. Οι πληροφορίες για την απόδοση μιας Υπηρεσίας Διαδικτύου περιλαμβάνουν λεπτομέρειες για την ποιότητα υπηρεσίας (όπως ο αναμενόμενος χρόνος μεταξύ αποτυχιών, ο μέγιστος χρόνος απόκρισης, η μέγιστη ρυθμαπόδοση δεδομένων, κλπ) που είναι κρίσιμες για την αξιολόγηση της εφαρμοσιμότητάς της. Οι πληροφορίες για την ασφάλεια μιας Υπηρεσίας Διαδικτύου πρέπει να περιλαμβάνουν λεπτομέρειες για την κρυπτογράφηση δεδομένων, την πολιτική ασφαλείας κλπ. Δυστυχώς, μέχρι τώρα είναι διαθέσιμες μόνο μερικές επίσημες notations για τον καθορισμό της απόδοσης και της ασφάλειας, για τις οποίες επιβάλλεται η χρήση των κοινών γλωσσών. Εκτός από την έλλειψη επίσημων γραφών απαραίτητες είναι και οι τυποποιημένες ταξονομίες που περιέχουν πλήρεις καταλόγους των πιθανών προδιαγραφών για τον καθορισμό της απόδοσης και της ασφάλειας.

Οι κατηγοριοποιήσεις Υπηρεσιών Διαδικτύου που έχουν συνοψιστεί στις κίτρινες σελίδες επιτρέπουν την ταξινόμηση εμπορικών υπηρεσιών χρησιμοποιώντας τις αντίστοιχες ταξονομίες για τον κλάδο της βιομηχανίας. Εντούτοις, θα ήταν καταλληλότερη η χρήση μιας εξειδικευμένης ταξονομίας για να κατηγοριοποιήσει την περιοχή. Εκτός από αυτό, δεν υπάρχει ανάγκη να επεκταθεί το UDDI πλαίσιο προδιαγραφών προκειμένου να καθορίσει ταξινομήσεις.

2.2.2. Εφαρμογή των βελτιώσεων στις Πράσινες Σελίδες

Οι πράσινες σελίδες του UDDI, που περιέχουν τεχνικές πληροφορίες, επικεντρώνονται στον καθορισμό της τοποθεσίας μιας Υπηρεσίας Διαδικτύου και της διεπαφής της. Αυτό είναι χρήσιμο για τη διαμόρφωση και τέλος την κλήση μιας Υπηρεσίας Διαδικτύου. Τυπικά μια διεπαφή περιέχει τις ονομασμένες υπηρεσίες που προσφέρονται, τις ονομασμένες δημόσιες ιδιότητες, τις μεταβλητές και τις σταθερές καθώς και τις δηλώσεις των συγκεκριμένων τύπων δεδομένων. Επιπλέον διευκρινίζει τις υπογραφές των υπηρεσιών και τη δήλωση εξαιρέσεων (τα οποία στην WSDL ονομάζονται “faults”). Αυτές οι πληροφορίες βρίσκονται στο binding template που βρίσκεται στην περιγραφή μιας Υπηρεσίας Διαδικτύου.

Παρόλα αυτά, η προδιαγραφή διεπαφών (που εστιάζει στην τεκμηρίωση της σύνταξης) δεν είναι συνήθως ικανοποιητική ώστε να καλέσει σωστά μια Υπηρεσία Διαδικτύου [7]. Αφ' ενός, για να κληθούν σωστά οι μέθοδοι μιας Υπηρεσία Διαδικτύου απαιτούνται πληροφορίες για τη συμπεριφορά της. Αφ' ετέρου, οι μέθοδοι των Υπηρεσιών Διαδικτύου μπορούν τυπικά να μην καλούνται αυθαίρετα, αλλά μόνο από καλά ορισμένες εντολές, με άλλα λόγια υπάρχουν εξαρτήσεις μεταξύ των μεθόδων Υπηρεσιών Διαδικτύου.

Η συμπεριφορά των Υπηρεσιών Διαδικτύου οφείλει να είναι τεκμηριωμένη καθορίζοντας τις pre- and post-συνθήκες των μεθόδων που δημοσιεύονται στις διεπαφές τους. Μια pre-συνθήκη εκφράζει τους περιορισμούς κάτω από τους οποίους μια μέθοδος που έχει κληθεί επιστρέφει σωστά αποτελέσματα. Μια post- συνθήκη περιγράφει την κατάσταση που προκύπτει από την εκτέλεση μιας μεθόδου και επομένως εγγυάται ότι θα ικανοποιήσει συγκεκριμένα κριτήρια. (δεδομένου ότι κλήθηκε με ικανοποιημένες pre-συνθήκες). Επιπλέον των pre- and post-συνθηκών οι προδιαγραφές των σταθερών είναι χρήσιμες στην περιγραφή των παγκόσμιων

ιδιοτήτων που διατηρούνται από όλες τις μεθόδους. Τόσο οι pre- και post- συνθήκες καθώς και οι σταθερές μπορούν να καθοριστούν με την Object Constraint Language (OCL), ένα επίσημο notation που παρέχεται ως τμήμα της Unified Modelling Language (UML). Επιπλέον, επίσημες προδιαγραφές μπορούν προαιρετικά να συνοδεύονται από περιγραφές με χρήση της κοινής γλώσσας.

Περιορισμοί που αφορούν την διατεταγμένη κλήση μεθόδων Υπηρεσιών Διαδικτύου μπορεί να συμβούν μέσα σε μια σύνθετη Υπηρεσία Διαδικτύου (π.χ. ο χρήστης πρέπει πρώτα να εγγραφεί σε μια υπηρεσία e-shop πριν παραγγείλει ένα προϊόν) αλλά και μεταξύ Υπηρεσιών Διαδικτύου (αυτό συμβαίνει ήδη όταν η υπηρεσία εγγραφής παρέχεται ως αυτόματη διαδικασία από τρίτο μέρος, όπως το. Microsoft .NET Passport). Αυτά καλούνται περιορισμοί συντονισμού και βοηθούν στη διαμόρφωση μιας Υπηρεσίας Διαδικτύου βάσει των διαδικασιών εφαρμογής. Οι αντίστοιχες προδιαγραφές μπορούν για παράδειγμα να διατυπωθούν χρησιμοποιώντας ένα εκτεταμένο OCL notation, που περιλαμβάνει τους χρονικούς τελεστές *before, after, sometime, always, until, sometimes_before, sometimes_past, always_past,* επιτρέποντας την προσαρμογή των υπαρχόντων εργαλείων που είναι ευρέως διαθέσιμα. Οι περιορισμοί συντονισμού μπορούν προαιρετικά να συνοδεύονται από περιγραφές χρησιμοποιώντας κοινή γλώσσα είτε ένα διάγραμμα UML ακολουθίας.

2.2.3. Εισαγωγή των Μπλε Σελίδων για Σημασιολογικές Προδιαγραφές

Δυστυχώς, το UDDI πλαίσιο προδιαγραφών δεν καθορίζει τη domain-specific σημασιολογία, που παρέχει πληροφορίες για την υλοποιημένη ορολογία, ούτε για την πραγματολογία των Υπηρεσιών Διαδικτύου, που παρέχουν πληροφορίες για τις υλοποιημένες διαδικασίες που υποστηρίζουν έναν εννοιολογικό στόχο. Προκειμένου να καλυφθεί αυτό το κενό, προστίθενται σε αυτό προδιαγραφές σχετικά με την ορολογία και το (εννοιολογικό) στόχο. Δε θα έπρεπε να προστεθούν στις άσπρες σελίδες (δεδομένου ότι αυτές οι προδιαγραφές δεν μπορούν να χαρακτηριστούν ως γενικές πληροφορίες) ούτε στις πράσινες σελίδες που εστιάζουν στις τεχνικές προδιαγραφές. Επομένως, εισάγονται οι **«μπλε σελίδες»** προκειμένου να συνοψίσουν τις εννοιολογικές πληροφορίες για μια Υπηρεσία Διαδικτύου.

Καταρχήν, οι μπλε σελίδες περιέχουν πληροφορίες για τη σημασιολογία μιας Υπηρεσίας Διαδικτύου, με επίκεντρο τον καθορισμό της υλοποιημένης ορολογίας (όροι, έννοιες, και σχέσεις μεταξύ τους). Αυτό μπορεί να γίνει παρέχοντας ένα λεξικό που να περιέχει τους όρους και τους αντίστοιχους ορισμούς τους. Έτσι, η χρησιμοποιούμενη γλώσσα πρέπει να τυποποιηθεί για να επιτρέψει την αυτοματοποιημένη επεξεργασία. Αυτό γίνεται με την παροχή σχεδίων για τη συντακτική κατασκευή προτάσεων (και καλείται τυποποιημένη κοινή γλώσσα). Εναλλακτικά, η σημασιολογία μπορεί να καθοριστεί με τον ορισμό μιας οντολογίας που παρέχει πληροφορίες σχετικά με όρους και έννοιες χρησιμοποιώντας μια αναγνώσιμη από μηχανές μορφή. Η DAML-S παρέχει μια εφαρμόσιμη επίσημη γραφή που μπορεί να χρησιμοποιηθεί στα πλαίσια του E-UDDI.

Οι εννοιολογικοί όροι μπορούν συνήθως να αντιστοιχιστούν σε τύπους δεδομένων που χρησιμοποιούνται στη διεπαφή (π.χ. ο εννοιολογικός όρος «πελάτης» χαρτογραφείται σε έναν αντίστοιχο τύπο δεδομένων που ονομάζεται “Customer”). Η ρητή τεκμηρίωση αυτών των αντιστοιχήσεων διευκολύνει το σχεδιασμό των σημασιολογικών μεταφραστών με βάση μια δεδομένη εννοιολογική σημασιολογία.

Εκτός από τη σημασιολογία, οι μπλε σελίδες περιέχουν επίσης εννοιολογικές πληροφορίες σχετικά με την πραγματολογία μιας Υπηρεσίας Διαδικτύου, σχετικά δηλαδή με τις domain-specific διαδικασίες που υλοποιήθηκαν για να υποστηρίξουν την αυτοματοποίηση ενός εννοιολογικού στόχου. Οι αντίστοιχες προδιαγραφές εστιάζουν στην τεκμηρίωση των υποστηριζόμενων στόχων και τον τρόπο που μπορούν να διασπαστούν σε (μια διαταγμένη λίστα) υποστόχων. Αυτό καθορίζει ένα ή περισσότερα work-flows που μπορεί να καθοριστούν με τη χρήση μιας επίσημης γλώσσας ορισμού work-flows ή μιας τυποποιημένης κοινής γλώσσας.

Εννοιολογικοί (υπο-) στόχοι μπορούν να αντιστοιχηθούν σε ένα σύνολο από μεθόδους (που παρέχονται στη διεπαφή της Υπηρεσίας Διαδικτύου). Αυτές οι αντιστοιχήσεις χρησιμεύουν ως οδηγίες λειτουργίας για τη λογική χρήση των μεθόδων της διεπαφής με σκοπό την εκτέλεση ενός στόχου (για παράδειγμα περιγράφοντας ποιες μέθοδοι θα κληθούν για να παραγγείλουν αυτόματα προϊόντα χρησιμοποιώντας μια Υπηρεσίας Διαδικτύου μέσα σε μια αλυσίδα ανεφοδιασμού).

2.2.4. Ένα Πρότυπο Δεδομένων Κατάλληλο για το E-UDDI

Η διατήρηση προς τα πίσω συμβατότητας του προτύπου UDDI οδηγεί σε ένα ευμετάβλητο XML μοντέλο δεδομένων. Όπως είδαμε προηγούμενα, πρόσφατα εισαχθείσες προδιαγραφές περιέχονται συνήθως σε ομάδες που τιτλοφορούνται με ένα αυτό-περιγραφικό όνομα (όπως «συμπεριφορά», «συντονισμός», «ορολογία», κλπ). Παρόλα αυτά, αυτό δεν ισχύει για προδιαγραφές που έχουν ληφθεί από το αρχικό πρότυπο δεδομένων. Κάποια από αυτά δεν ομαδοποιούνται καθόλου (πράγμα που ισχύει για διαχειριστικές πληροφορίες) και κάποια τιτλοφορούνται με ένα μάλλον τεχνικό όνομα.

Επιπλέον το μοντέλο δεδομένων που προκύπτει δεν περιέχει ρητά παρουσιασμένη θεματική ομαδοποίηση σε λευκές, κίτρινες, μπλε, και πράσινες σελίδες. Όπως είδαμε πριν, η εφαρμογή της ομαδοποίησης στις πολύπλοκες προδιαγραφές μιας Υπηρεσίας Διαδικτύου συμβάλλει σε μια αναγνώσιμη (από άνθρωπο) δομή και επομένως θα έπρεπε να χρησιμοποιηθεί για να ταξινομήσει το αντίστοιχο πρότυπο δεδομένων (ειδικά αν χρησιμοποιείται πρώτιστα για την ανταλλαγή πληροφοριών, πράγμα που υποδηλώνεται χρησιμοποιώντας XML για το σχεδιασμό της).

2.3. Συμπεράσματα

Στη μελέτη αυτή προτάθηκαν διάφορες βελτιώσεις του πλαισίου UDDI με σκοπό τη διευκόλυνση της ανακάλυψης και της διαμόρφωσης Υπηρεσιών Διαδικτύου. Αυτό οδηγεί σε ένα λεπτομερές πλαίσιο προδιαγραφών που διαιρείται σε εννιά πλευρές τεκμηρίωσης και τέσσερις θεματικές ενότητες. Παρέχει ποικίλες πληροφορίες που υποστηρίζουν την επαναχρησιμοποίηση των Υπηρεσιών Διαδικτύου. Επιπλέον, οι επίσημες γραφές επιτρέπουν το σχεδιασμό CASE tools για την υποστήριξη των διαδικασιών προδιαγραφής, αξιολόγησης, και διαμόρφωσης.

Το πρότυπο δεδομένων που προτείνεται εφαρμόζει μια θεωρητική εναλλακτική λύση στις υπάρχουσες λειτουργικές υλοποιήσεις του UDDI (www.uddi.org). Η δομή που προκύπτει δεν εφαρμόζεται μόνο για να καθορίσει Υπηρεσίες Διαδικτύου, αλλά μπορεί γενικά να υλοποιηθεί και για να καθορίσει components. Αυτό θα απαιτούσε την εισαγωγή κάποιων επιπλέον προδιαγραφών σχετικά με την πλατφόρμα, τις απαιτήσεις του συστήματος, τον τύπο επαναχρησιμοποίησης (λογικός ή φυσικός) και τον τύπο κώδικα. Αυτές οι προδιαγραφές θα μπορούσαν να προστεθούν στις διαχειριστικές πληροφορίες που ανήκουν στις λευκές σελίδες. Επιπλέον, η χρήση μιας τυποποιημένης Γλώσσας Περιγραφής Διεπαφών (Interface Description Language) θα έπρεπε να έχει τη δυνατότητα να καθορίσει διεπαφές επιπλέον της WSDL, που εξειδικεύεται στη διευκρίνιση Υπηρεσιών Διαδικτύου.

Συνεπώς, θα μπορούσε να επιτευχθεί ακόμη και μια ενοποιημένη προδιαγραφή των components λογισμικού (που θα μπορούσε να είναι η βάση για τους μελλοντικούς component καταλόγους και τα CASE tools). Εξετάζοντας αυτές τις πιθανές εξελίξεις, οι Υπηρεσίες Διαδικτύου προχωρούν ένα βήμα στην νέα εποχή της ενοποίησης εφαρμογών και την ανάπτυξη component-based εφαρμογών.

3. Ένα πρότυπο για Ανακάλυψη Υπηρεσιών Διαδικτύου με Ποιότητα Υπηρεσίας

Καθώς οι Υπηρεσίες Διαδικτύου μπορούν να παρέχονται από τρίτα μέρη και να καλούνται δυναμικά μέσο του Διαδικτύου, η ποιότητα υπηρεσίας τους (QoS) μπορεί να ποικίλει σημαντικά. Επομένως είναι σημαντικό να υπάρξει ένα πλαίσιο εξασφάλισης της ποιότητας υπηρεσίας που παρέχεται από τον πάροχο, της ποιότητας υπηρεσίας που απαιτείται από τον πελάτη, και της αντιστοιχίας μεταξύ τους κατά την ανακάλυψη της Υπηρεσίας Διαδικτύου. Το διεθνές πρότυπο ποιότητας ISO 8402 περιγράφει την ποιότητα ως: *«το σύνολο των ιδιοτήτων και χαρακτηριστικών ενός προϊόντος ή μιας υπηρεσίαςπου φέρουν τη δυνατότητα να ικανοποιήσουν τις ζητούμενες ανάγκες».*

Η ποιότητα υπηρεσίας θεωρείται εδώ ως ένα σύνολο από μη λειτουργικά χαρακτηριστικά που μπορεί να επηρεάσουν την ποιότητα υπηρεσίας που παρέχεται από μια Υπηρεσία Διαδικτύου. Υπάρχουν πολλές πτυχές της ποιότητα υπηρεσίας που είναι κρίσιμες για τις Υπηρεσίες Διαδικτύου. Οι QoS μπορούν να οργανωθούν σε κατηγορίες καθεμία από τις οποίες περιλαμβάνει ένα σύνολο από ποσοτικά προσδιορίσιμες παραμέτρους. Στη συνέχεια παρουσιάζονται συνοπτικά αυτές οι κατηγορίες [8]. Για τη διευκόλυνση της περιγραφής οι κατηγορίες ομαδοποιούνται σε διαφορετικούς τύπους (π.χ. QoS σχετικές με το κόστος) και χρησιμοποιούνται παραδείγματα που επιδεικνύουν τα είδη των ερωτημάτων που μπορεί να θέσουν συγκεκριμένες QoS.

3.1. Κατηγορίες ποιότητας υπηρεσίας (QoS)

3.1.1. Ποιότητα Υπηρεσίας σε χρόνο εκτέλεσης

1. ***Εξελιξιμότητα*** – Η ικανότητα της αύξησης της υπολογιστικής δυνατότητας του παρόχου υπηρεσιών και της δυνατότητας του συστήματος να επεξεργαστεί περισσότερες λειτουργίες ή συνδιαλλαγές σε ένα δεδομένο χρονικό διάστημα.
   Παράδειγμα: Θα κλιμακωθεί το σύστημα για να διαχειριστεί Χ συνδιαλλαγές ανά δευτερόλεπτο;

2. ***Χωρητικότητα*** – Το όριο των ταυτόχρονων αιτημάτων για εγγυημένη απόδοση.
   Παράδειγμα: Πόσες ταυτόχρονες συνδέσεις υποστηρίζει η υπηρεσία;

3. ***Απόδοση*** – Το μέτρο της ταχύτητας ολοκλήρωσης μιας αίτησης υπηρεσίας. Μετράται με:
   - *Χρόνο απόκρισης* – ο εγγυημένος ανώτατος (είτε μέσος είτε ελάχιστος) χρόνος που απαιτείται για την ολοκλήρωση μιας αίτησης υπηρεσίας.
   - *Χρόνος λανθάνουσας κατάστασης (Latency)* – Ο χρόνος που απαιτείται μεταξύ της στιγμής που θα φτάσει ένα αίτημα έως ότου ικανοποιηθεί.

   Παράδειγμα: Ποια είναι η μέση καθυστέρηση στην εξυπηρέτηση ενός αιτήματος;

4. ***Ρυθμοαπόδοση*** – Ο αριθμός των ολοκληρωμένων αιτημάτων υπηρεσιών σε μια συγκεκριμένη χρονική περίοδο.

5. ***Αξιοπιστία*** – Η δυνατότητα μιας υπηρεσίας να εκτελέσει τις απαιτούμενες λειτουργίες σύμφωνα με καθορισμένους όρους για ένα συγκεκριμένο χρονικό διάστημα.

6. ***Διαθεσιμότητα*** – Η πιθανότητα το σύστημα να είναι σε λειτουργία όταν κάποιος χρήστης καλέσει την υπηρεσία.

7. ***Ευρωστία/ Ευελιξία*** **–** Ο βαθμός στον οποίο μια υπηρεσία μπορεί να λειτουργήσει σωστά παρουσία άκυρων, ελλιπών ή αντιφατικών εισόδων.
   <u>Παράδειγμα</u>: Η υπηρεσία θα λειτουργήσει ακόμα και εάν παρέχονται ελλιπείς παράμετροι για να εξυπηρετήσει την κλήση υπηρεσίας;

8. ***Χειρισμός εξαιρέσεων*** – Δεδομένου ότι δεν είναι δυνατό για το σχεδιαστή υπηρεσιών να καθορίσει όλες τις πιθανές εξόδους και εναλλακτικές λύσεις, μπορούν να αναμένονται εξαιρέσεις. Ο χειρισμός εξαιρέσεων είναι ο τρόπος που η υπηρεσία χειρίζεται τις εξαιρέσεις.
   <u>Παράδειγμα</u>: Η υπηρεσία θα δουλεύει σωστά ακόμα και αν δώσω λιγότερες παραμέτρους απ' ότι απαιτεί;

9. ***Ακρίβεια*** – Καθορίζει το ποσοστό λάθους που παράγεται από την υπηρεσία.
   <u>Παράδειγμα</u>: Πόσα λάθη παράγει η υπηρεσία σε μια χρονική περίοδο;

3.1.2. Διαχείριση Προδιαγραφών και Ποιότητα Υπηρεσίας Κόστους

- ***Υποστηριζόμενα πρότυπα*** – Ένα μέτρο του εάν η υπηρεσία είναι συμβατή με τα πρότυπα (π.χ βιομηχανικά πρότυπα). Αυτό μπορεί να επηρεάσει τη φορητότητα της υπηρεσίας και την διαλειτουργικότητα της με άλλες.
  <u>Παράδειγμα</u>: Πόσο θα τηρήσει η υπηρεσία τα εφαρμοζόμενα πρότυπα;

- ***Σταθερότητα /κύκλος αλλαγής*** – Ένα μέτρο της συχνότητας αλλαγής της υπηρεσίας όσον αφορά στη διεπαφή και την υλοποίηση της.
  <u>Παράδειγμα</u>: Πόσο σταθερή είναι η υπηρεσία, πόσο συχνά αλλάζει(διεπαφή και υλοποίηση);
- ***Εγγυημένες απαιτήσεις μηνύματος*** – Εξασφαλίζει τη σειρά και την διατήρηση των μηνυμάτων;
- ***Κόστος*** – Είναι ένα μέτρο του κόστους που σχετίζεται με την αίτηση της υπηρεσίας.
  <u>Παράδειγμα</u>: Από τι εξαρτάται το κόστος (ανά αίτηση ή ανά μέγεθος δεδομένων);

- ***Πληρότητα*** **–** Ένα μέτρο της διαφοράς μεταξύ του καθορισμένου συνόλου χαρακτηριστικών και του υλοποιημένου συνόλου χαρακτηριστικών.
  <u>Παράδειγμα</u>: Πόσα από τα καθορισμένα χαρακτηριστικά είναι διαθέσιμα;

3.1.3. Ποιότητα Υπηρεσίας Ασφάλειας

Μετρά την εμπιστοσύνη και τους μηχανισμούς ασφάλειας που υλοποιούνται.

1. ***Πιστοποίηση χρήστη*** – Ο τρόπος που η υπηρεσία πιστοποιεί τους χρήστες ή άλλες υπηρεσίες που μπορούν να προσπελάσουν τα δεδομένα της;

2. ***Εξουσιοδότηση*** – Ο τρόπος που η υπηρεσία εξουσιοδοτεί τους κατάλληλους χρήστες έτσι ώστε μόνο αυτοί να μπορούν να προσπελάσουν τις προστατευμένες υπηρεσίες;

3. ***Εμπιστευτικότητα*** – Ο τρόπος που διαχειρίζεται η υπηρεσία τα δεδομένα, έτσι ώστε μόνο εξουσιοδοτημένοι χρήστες να μπορούν να προσπελάσουν ή να τροποποιήσουν τα δεδομένα.

4. ***Ανιχνευσιμότητα*** – Η πιθανότητα να ανιχνευθεί η ιστορία μιας υπηρεσίας όταν μια αίτηση έχει εξυπηρετηθεί.

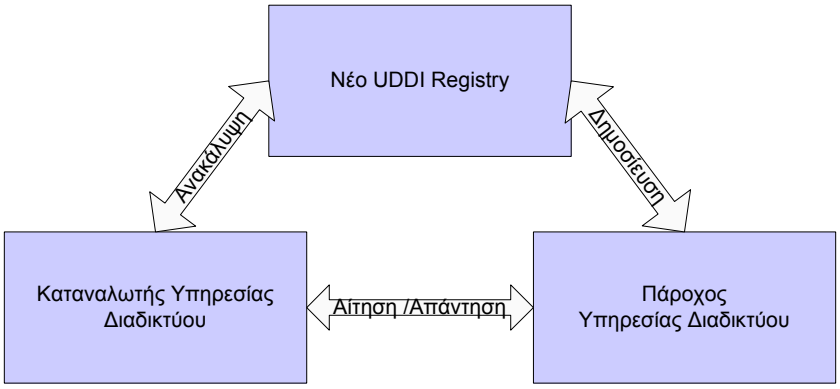
5. ***Κρυπτογράφηση δεδομένων*** – Ο τρόπος που η υπηρεσία κρυπτογραφεί τα δεδομένα.

Μέχρι να εξετασθούν τα παραπάνω θέματα ποιότητας υπηρεσίας, δεν είναι ρεαλιστικό να αναμένεται ότι μια επιχείρηση θα ήθελε να αναζητήσει μια Υπηρεσία Διαδικτύου με βάση τις αναμενόμενες λειτουργικές απαιτήσεις σε ένα UDDI κατάλογο και να καλέσει αυτήν την υπηρεσία χωρίς τη διαβεβαίωση ότι θα επιτευχθεί η αναμενόμενη ποιότητα υπηρεσίας.

Στη συνέχεια παρουσιάζεται ένα νέο πρότυπο που έχει προταθεί για την ανακάλυψη υπηρεσιών όπου η ποιότητα της υπηρεσίας λαμβάνεται ως περιορισμός κατά την αναζήτηση. Το πρότυπο αυτό αποτελεί επέκταση του υπάρχοντος UDDI μοντέλου και η υιοθέτηση του μπορεί να παρέχει στους καταναλωτές Υπηρεσιών Διαδικτύου την ποιότητα της υπηρεσίας που πρόκειται να καλέσουν.

3.2. Επέκταση του προτύπου UDDI

Το υπάρχον πρότυπο για δημοσίευση και ανακάλυψη Υπηρεσιών Διαδικτύου με βάση τους UDDI καταλόγους είναι αρκετά ανεξέλεγκτο. Το 48% των εγγραφών του UDDI καταλόγου (δοκιμές σε tModels μόνο) περιέχει αχρησιμοποίητους συνδέσμους. Αυτοί οι σύνδεσμοι περιέχουν ελλιπείς είτε ανακριβείς πληροφορίες [9]. Αυτό είναι ένα παράδειγμα που επιδεικνύει την σπουδαιότητα της αντιμετώπισης των θεμάτων ποιότητας υπηρεσίας (QoS). Μια άλλη ανεπάρκεια του υπάρχοντος UDDI προτύπου είναι ότι περιορίζει την αναζήτηση υπηρεσιών σε λειτουργικές απαιτήσεις μόνο. Πιθανότατα όμως υπάρχουν περισσότερες από μια διαθέσιμες Υπηρεσίες Διαδικτύου που ικανοποιούν τις ίδιες λειτουργικές απαιτήσεις, με διαφορετικά χαρακτηριστικά ποιότητας υπηρεσίας. Επομένως, η ικανότητα ενσωμάτωσης της ποιότητας υπηρεσίας στη διαδικασία ανακάλυψης γίνεται ιδιαίτερα σημαντική. Για να προσπεραστούν αυτές οι ανεπάρκειες προτείνεται ένα νέο μοντέλο.



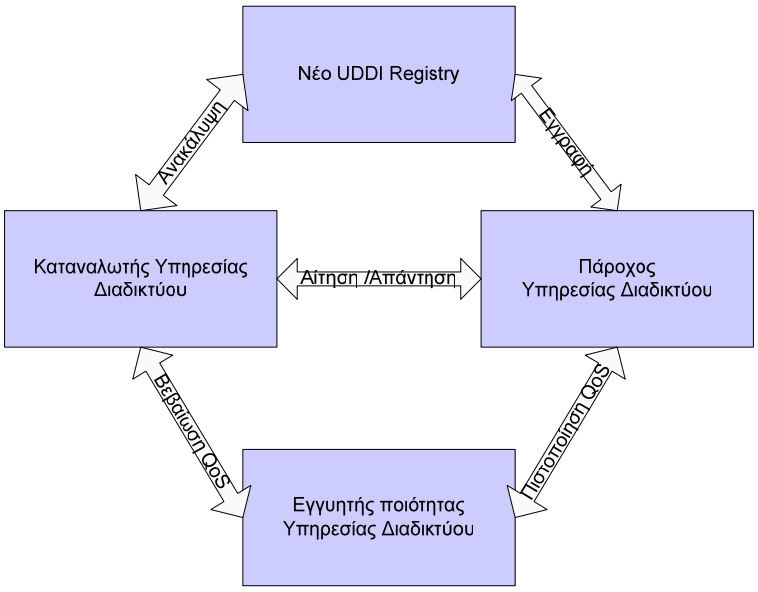
Σχήμα 2: Τρέχον μοντέλο δέσμευσης Υπηρεσιών Διαδικτύου (push-bind)

Το προτεινόμενο πλαίσιο είναι ένα πρότυπο που μπορεί να συνυπάρξει με τους υπάρχοντες UDDI καταλόγους. Οι κατάλογοι αυτοί μπορούν να προσφέρουν υπηρεσίες σε χρήστες για τους οποίους η ποιότητα υπηρεσίας δεν είναι σημαντική. Οι κατάλογοι που βασίζονται στο νέο, προτεινόμενο μοντέλο μπορούν να εξυπηρετήσουν εφαρμογές που χρειάζονται διαβεβαίωση της ποιότητας υπηρεσίας. Υπάρχουν τέσσερις ρόλοι στο προτεινόμενο πρότυπο: α)ο Πάροχος Υπηρεσίας Διαδικτύου, β) ο Πελάτης Υπηρεσίας Διαδικτύου, γ) ο Εγγυητής QoS Υπηρεσίας Διαδικτύου (certifier), και δ) ο νέος UDDI κατάλογος. Όπως πριν, ο πάροχος Υπηρεσίας Διαδικτύου παρέχει μια Υπηρεσία Διαδικτύου με την έκδοση της στον κατάλογο, ο πελάτης

Υπηρεσίας Διαδικτύου χρειάζεται την Υπηρεσία Διαδικτύου που παρέχεται από τον πάροχο, ο νέος UDDI κατάλογος είναι ένας κατάλογοςς από καταχωρημένες Υπηρεσίες Διαδικτύου με ευχέρεια αναζήτησης, ενώ ο ρόλος του εγγυητή είναι να ελέγξει τις υποσχέσεις QoS του προμηθευτής υπηρεσίας.

Ο προτεινόμενος νέος κατάλογος διαφέρει από το υπάρχον UDDI πρότυπο καθώς έχει πληροφορίες για τη λειτουργική περιγραφή της Υπηρεσίας Διαδικτύου όπως επίσης και για την αντίστοιχη ποιότητα υπηρεσίας που είναι καταχωρημένη στον κατάλογο. Η αναζήτηση θα μπορούσε να έχει ως περιορισμούς τη λειτουργική περιγραφή της επιθυμητής Υπηρεσίας Διαδικτύου καθώς και τα απαιτούμενα χαρακτηριστικά ποιότητας υπηρεσίας. Ο νέος ρόλος σε αυτό το μοντέλο είναι ο εγγυητής QoS Υπηρεσίας Διαδικτύου που δεν υπάρχει στο τρέχον UDDI πρότυπο. Ο εγγυητής πιστοποιεί τους ισχυρισμούς της ποιότητας υπηρεσίας μιας Υπηρεσίας Διαδικτύου πριν την εγγραφή της.

Στο προτεινόμενο πρότυπο, ένας πάροχος Υπηρεσίας Διαδικτύου πρέπει να παρέχει πληροφορίες σχετικά με την εταιρία, τα λειτουργικά χαρακτηριστικά της παρεχόμενης υπηρεσίας οπως απαιτείται από τον υπάρχον UDDI κατάλογο, όπως επίσης και πληροφορίες ποιότητας υπηρεσίας σχετικές με την προτεινόμενη Υπηρεσία Διαδικτύου. Η υποσχόμενη ποιότητα υπηρεσίας πρέπει να πιστοποιηθεί και να καταχωρηθεί στον κατάλογο.Ο πάροχος Υπηρεσίας Διαδικτύου πρέπει πρώτα να επικοινωνήσει με τον εγγυητή QoS Υπηρεσίας Διαδικτύου διαβιβάζοντας τις QoS υποσχέσεις του. Ο εγγυητής ελέγχει τους ισχυρισμούς του προμηθευτή και είτε επιβεβαιώνει την ποιότητα είτε την υποβιβάζει. Το αποτέλεσμα στέλνεται πίσω στον πάροχο με πληροφορίες αναγνώρισης πιστοποίησης. Αυτές οι πληροφορίες επίσης καταχωρούνται στον κατάλογο του εγγυητή και προσδιορίζονται από ένα certification Id. Ο εγγυητής παρέχει ένα σύνολο από Υπηρεσίες Διαδικτύου ώστε οποιαδήποτε ενδιαφερόμενα μέρη να μπορούν να προσπελάσουν τον κατάλογό του για επαληθευτικούς σκοπούς. Αφού εκδοθεί η πιστοποίηση από τον εγγυητή, ο πάροχος καταχωρείται στον UDDI κατάλογο, τόσο με την λειτουργική περιγραφή της υπηρεσίας όσο και με τις σχετικές πληροφορίες πιστοποίησης της ποιότητας υπηρεσίας. Ο UDDI κατάλογος επικοινωνεί με τον εγγυητή για να ελέγξει την ύπαρξη πιστοποίησης. Αφού ελεγχθεί επιτυχημένα, ο κατάλογος καταχωρεί την υπηρεσία στην αποθήκη του.



Σχήμα 3:Ένα νέο μοντέλο εγγραφής και ανακάλυψης Υπηρεσιών Διαδικτύου

Ένας καταναλωτής μιας Υπηρεσίας Διαδικτύου έχει συγκεκριμένες λειτουργικές και ποιοτικές απαιτήσεις, όπως για παράδειγμα "χρόνος απόκρισης μικρότερος από 2 ms και κόστος λιγότερο από 100€ ανά κλήση". Ο καταναλωτής ψάχνει τον UDDI κατάλογο για μια Υπηρεσία Διαδικτύου με την απαιτούμενη λειτουργία αλλά μπορεί επίσης να προσθέσει ποιοτικούς περιορισμούς στην λειτουργία αναζήτησης. Εάν υπάρχουν πολλές Υπηρεσίες Διαδικτύου στο UDDI κατάλογο με παρόμοιες λειτουργίες, τότε οι απαιτήσεις ποιότητας υπηρεσίας επιβάλλουν μια λεπτομερέστερη αναζήτηση. Η αναζήτηση θα επιστρέψει μια Υπηρεσία Διαδικτύου που προσφέρει την απαιτούμενη λειτουργικότητα με το επιθυμητό σύνολο ποιότητας υπηρεσίας. Εάν δεν υπάρχει Υπηρεσία Διαδικτύου με αυτές τις ιδιότητες, ο καταναλωτής ενημερώνεται ώστε να μειώσει τους περιορισμούς της ποιότητα ή να εξετάσει trade-offs μεταξύ των επιθυμητών ιδιοτήτων της υπηρεσίας. Μόλις βρεθεί μια Υπηρεσία Διαδικτύου, η WSDL και οι πιστοποιημένες QoS πληροφορίες ανακτώνται από τον καταναλωτή. Ο καταναλωτής μπορεί να διασταυρώσει τις QoS πιστοποιήσεις με τον εγγυητή χρησιμοποιώντας το certification Id. Μόλις ο καταναλωτής είναι ικανοποιημένος με τα αποτελέσματα, η Υπηρεσία Διαδικτύου καλείται όπως στο τρέχον πρότυπο.

3.3. Εκτεταμένες UDDI δομές δεδομένων

Οι πληροφορίες που συνθέτουν μια υπάρχουσα UDDI εγγραφή αποτελούνται από πέντε τύπους δομών δεδομένων [10]: το businessEntity, το businessService, το bindingTemplate, το publisherAssertion και το tModel. Αυτοί οι πέντε τύποι αποτελούν το πλήρες ποσό πληροφοριών που παρέχεται από το υπάρχον UDDI πλαίσιο περιγραφής υπηρεσιών. Καθεμία από αυτές τις XML δομές περιέχει ένα πλήθος από πεδία δεδομένων που εξυπηρετούν είτε επιχειρηματικούς είτε τεχνικούς περιγραφικούς σκοπούς.

tModel	Περιγραφή των προδιαγραφών για υπηρεσίες ή ταξονομίες. Βάση για τεχνικά fingerprints
businessService	Περιγραφικές πληροφορίες μιας συγκεκριμένης υπηρεσίας
businessEntity	Πληροφορίες για το συμβαλλόμενο μέρος που δημοσιεύει τις πληροφορίες μιας οικογένειας υπηρεσιών
bindingTemplate	Τεχνικές πληροφορίες σχετικά με το σημείο εισόδου μιας υπηρεσία και την κατασκευή specs fingerprints
publisherAssertion	Πληροφορίες για μια σχέση μεταξύ δύο συμβαλλόμενων μερών, επιβεβαιωμένη από το ένα ή και τα δύο fingerprints
qualityInformation	Πληροφορίες ποιότητας υπηρεσιας

Πίνακας 9: Τύποι δομών δεδομένων των UDDI εγγραφών

Προκειμένου να πραγματοποιηθεί η προτεινόμενη επέκταση του UDDI προτύπου προτείνεται η προσθήκη ενός νέου τύπου δομής δεδομένων. Αυτή η δομή που πρέπει να προστεθεί αντιπροσωπεύει την περιγραφή της ποιότητας υπηρεσίας μιας συγκεκριμένης πληροφορίας. Διαφορετικές κατηγορίες πληροφοριών ποιότητας υπηρεσίας μπορούν να παρέχονται μέσο της δομής **qualityInformation**, όπως η διαθεσιμότητα, η αξιοπιστία κλπ. Αυτή η προτεινόμενη δομή δεδομένων είναι του τύπου businessService, και επιπλέον του τύπου bindingTemplate, που παρέχει δεσμευτικές πληροφορίες μιας συγκεκριμένης υπηρεσίας.

Όπως το bindingTemplate, αυτή η νέα δομή δεδομένων αναφέρεται επίσης στα tModels που καθορίζονται στον UDDI κατάλογο. Αντίθετα με το bindingTemplate που αναφέρεται στα tModels ως αναφορά στις προδιαγραφές διεπαφών των υπηρεσιών, το qualityInformation αναφέρεται στα tModels ως αναφορά στις ταξονομίες ποιότητας υπηρεσίας που πρέπει επίσης να καθοριστούν στον επεκταμένο UDDI κατάλογο. Αυτές οι ταξονομίες καθορίζουν νέες ορολογίες

ή έννοιες σχετικά με τις προτεινόμενες QoS πληροφορίες, όποιοι δεν υπάρχουν στους υπάρχοντες UDDI καταλόγους.

```
<tModel tModelKey="uuid:0e727db0-3e14-11d5-98bf-002035229c64">
    <name>uudi-org:qualityInformation</name>
    <description xml:lang="en">Quality of Service Information</description>
    <overviewDoc>
       <description xml:lang="en"></description>
       <overviewURL> http://www.uddi.org/specification.html
       </overviewURL>
    </overviewDoc>
    <categoryBag>
        <keyedReference
           keyName="uddi-org:types" keyValue="categorization"
           tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"/>
        <keyedReference
           keyName="uddi-org:types" keyValue="checked"
           tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"/>
       <keyedReference
           keyName="uddi-org:types" keyValue="specification"
           tModelKey="uuid:c1acf26d-9672-4404-9d70 39b756e62ab4"/>
    </categoryBag>
</tModel>
```

Σχήμα 4: tModel για πληροφορίες ποιότητας υπηρεσίας

```
<?xml version="1.0" encoding="UTF-8" ?>
  <envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <body>
      <find_service businessKey="*" generic="1.0"
         xmlns="urn:uddi-org:api" maxRows="100">
         <findQualifiers></findQualifiers>
        <name>Stock quote</name>
        <qualityInformation>
          <availability> 0.9 </availability>
        </qualityInformation>
      </find_service>
    </body>
  </envelope>
```

Σχήμα 5: Αίτημα SOAP για ανακάλυψη υπηρεσίας

3.4. Συμπεράσματα για το προτεινόμενο πρότυπο

Συμπερασματικά στο νέο πρότυπο ανακάλυψης Υπηρεσιών Διαδικτύου που προτείνεται πρέπει να ληφθούν υπόψη για την ανακάλυψη υπηρεσιών οι λειτουργικές και μη λειτουργικές απαιτήσεις (δηλ. ποιότητα της υπηρεσίας). Ένας νέος ρόλος εισάγεται σε αυτό το πλαίσιο – ο εγγυητής, ο οποίος επιβεβαιώνει τις υποσχέσεις ποιότητας υπηρεσιών του προμηθευτή.

Ο ρόλος τους είναι παρόμοιος με τους πράκτορες αξιολόγησης σε άλλες περιοχές όπως ο οικονομικός τομέας, η βιομηχανία κ.λπ. Επίσης προτείνεται μια επέκταση των τύπων δομών δεδομένων του UDDI που θα μπορούσε να χρησιμοποιηθεί για την υλοποίηση του προτεινόμενου εκτεταμένου UDDI προτύπου.

Για να υλοποιηθεί το προτεινόμενο πλαίσιο, πρέπει να ορισθεί ένα σύνολο από μετρικές που ποσοτικοποιούν κάθε προτεινόμενη κατηγορία QoS και τα σχετικά πρότυπα για την

αναπαράστασή τους. Περαιτέρω έρευνα απαιτείται για την εύρεση αλγορίθμων ταιριάσματος μεταξύ επιθυμητής και παρεχόμενης ποιότητας υπηρεσίας. Τέλος, για την πλήρη αξιοποίηση του προτεινόμενου πλαισίου είναι απαραίτητο να ενσωματωθεί η σημασιολογική μοντελοποίηση των QoS κατηγοριών.

4. Αλγόριθμος για αντιστοίχηση Υπηρεσιών Διαδικτύου

Η λειτουργία αναζήτησης με βάση λέξεις κλειδιά για Υπηρεσίες Διαδικτύου που παρέχουν οι κατάλογοι UDDI είναι πολύ απλή και αποτυγχάνει να ανακαλύψει τις συσχετίσεις μεταξύ των Υπηρεσιών Διαδικτύου. Στο κεφάλαιο αυτό, προτείνεται ένας αλγόριθμος που ανακτά τις στενά συσχετιζόμενες Υπηρεσίες Διαδικτύου [11]. Ο προτεινόμενος αλγόριθμος βασίζεται στη Singular Value Decomposition (SVD) της γραμμικής άλγεβρας, η οποία αποκαλύπτει σημασιολογικές σχέσεις μεταξύ των Υπηρεσιών Διαδικτύου.

Διάφορες τεχνικές έχουν προταθεί για την αντιστοίχηση των Υπηρεσιών Διαδικτύου οι οποίες απαιτούν τον καθορισμό ενός tModel (τεχνικό μοντέλο). Το tModel αντιπροσωπεύει μια αφηρημένη διεπαφή που εφαρμόζεται από μια Υπηρεσία Διαδικτύου και πρέπει να υιοθετηθεί από τις Υπηρεσίες Διαδικτύου ενώ δημοσιεύονται στο UDDI. Αυτό επιτρέπει την αντιστοίχηση των Υπηρεσιών Διαδικτύου. Το μειονέκτημα αυτής της τεχνικής είναι ότι οι μόνες Υπηρεσίες Διαδικτύου που μπορούν να συμμετέχουν στην ενισχυμένη διαδικασία αντιστοίχησης είναι εκείνες που ακολουθούν το tModel που καθορίζεται από το συντάκτη. Οι Υπηρεσίες Διαδικτύου που δεν υιοθετούν το tModel που προτείνεται από τους συντάκτες δεν θα είναι σε θέση να λάβουν καλύτερη αντιστοίχηση. Σε αυτό το κεφάλαιο, προτείνεται μια προσέγγιση της αντιστοίχησης μεταξύ των Υπηρεσιών Διαδικτύου που βασίζεται στο κείμενο που χρησιμοποιείται για να τις περιγράψει στο UDDI. Αυτή η προσέγγιση εκμεταλλεύεται τη SVD για να αποκαλύψει σημασιολογικές σχέσεις μεταξύ των Υπηρεσιών Διαδικτύου, οι οποίες επιτρέπουν πιο έγκυρα αποτελέσματα. Ο προτεινόμενος αλγόριθμος μπορεί να χρησιμοποιηθεί στη επεξεργασία των επιστρεφόμενων αποτελεσμάτων από το UDDI και να παρέχει καλύτερες υπηρεσίες. Το πλεονέκτημα της προτεινόμενης μεθόδου είναι ότι μπορεί να εφαρμοστεί σε οποιαδήποτε Υπηρεσία Διαδικτύου που δημοσιεύεται στο UDDI. Η τεχνική δεν περιορίζεται από το tModel που χρησιμοποιείται από μια Υπηρεσία Διαδικτύου κατά τη διάρκεια της έκδοσης.

Στην συνέχεια, περιγράφεται η συμβατική διαδικασία ανακάλυψης Υπηρεσιών Διαδικτύου από τους καταλόγους UDDI και έπειτα προτείνεται ο αλγόριθμος για την αντιστοίχιση των Υπηρεσιών Διαδικτύου.

4.1. Ανακάλυψη Υπηρεσιών Διαδικτύου

Η ανακάλυψη Υπηρεσιών Διαδικτύου είναι η διαδικασία εύρεσης ενός κατάλληλου παρόχου υπηρεσιών για μια αίτηση υπηρεσίσς μέσω ενός matchmaker υπηρεσιών. Τα βασικά βήματα αυτής της διαδικασίας είναι: περιγραφή υπηρεσιών, δημοσίευση υπηρεσιών, περιγραφή των καταναλωτικών αναγκών, αντιστοιχία υπηρεσιών.

Είναι δυνατό να ανακαλυφθούν οι μεμονωμένες Υπηρεσίες Διαδικτύου με βάση τις παραμέτρους που εισάγονται στην εγγραφή UDDI κατά τη διάρκεια της δημοσίευσης. Εντούτοις, ο κατάλογος UDDI δεν διευκολύνει την αντιστοίχιση των Υπηρεσιών Διαδικτύου που παρέχουν παρόμοιες λειτουργίες. Προκειμένου να ανακτηθούν παρόμοιες υπηρεσίες, εξάγονται οι λέξεις από τις περιγραφές των Υπηρεσιών Διαδικτύου όπως έχουν εισαχθεί στον κατάλογο UDDI. Αυτές οι λέξεις προεπεξεργάζονται και τους ανατίθενται βάρη. Τα βάρη που ανατίθενται στις λέξεις καθορίζουν τη σημασία των λέξεων μέσα στην UDDI αποθήκη. Οι ζυγισμένες λέξεις χρησιμοποιούνται για να χτίσουν μια μήτρα που περιέχει τις πληροφορίες για όλες τις Υπηρεσίες Διαδικτύου που έχουν κοινές λέξεις στην περιγραφή τους. Η Singular Value Decomposition εφαρμόζεται στις μήτρες για να λάβει τις σημασιολογικές σχέσεις μεταξύ των Υπηρεσιών Διαδικτύου.

4.2. Ο αλγόριθμος και η εφαρμογή

Οι λέξεις που εξάγονται από το στοιχείο επιχειρησιακής περιγραφής του καταλόγου UDDI προεπεξεργάζονται πριν τους αναθέτουν τα βάρη. Στο στάδιο προεπεξεργασίας αφαιρούνται τα stopwords. Τα Stopwords είναι συχνά εμφανιζόμενες λέξεις που μπορούν να αγνοηθούν με

σκοπό την αντιστοίχηση των Υπηρεσιών Διαδικτύου. Το σύνολο των λέξεων που λαμβάνονται μετά από την αφαίρεση των stopwords περνά μέσα από ένα stemming αλγορίθμο. Αυτό μας επιτρέπει να λάβουμε τη ρίζα της λέξης. Χρησιμοποιείται ο αλγόριθμος του Porter για την αφαίρεση του επιθέματος.

Στις προκύπτουσες λέξεις ανατίθεται ένα βάρος. Το βάρος υπολογίζεται με βάση την αντίστροφη συχνότητα εγγράφων (Inverse Document Frequency, IDF). Το IDF είναι ένα μέτρο σπουδαιότητας μιας λέξης. Η βάση για τη στάθμιση IDF είναι η παρατήρηση ότι οι άνθρωποι τείνουν να εκφράσουν τις ανάγκες πληροφοριών τους χρησιμοποιώντας μάλλον γενικούς, συχνά εμφανιζόμενους όρους, ενώ οι χαμηλής συχνότητας όροι είναι πιθανό να είναι ιδιαίτερα σημαντικοί στον προσδιορισμό του σχετικού υλικού. Αυτό συμβαίνει επειδή ο αριθμός των εγγράφων που είναι σχετικά με μια ερώτηση είναι γενικά μικρός, και έτσι οποιοιδήποτε συχνά εμφανιζόμενοι όροι πρέπει απαραιτήτως να εμφανιστούν σε πολλά άσχετα έγγραφα. Οι σπάνια εμφανιζόμενοι όροι έχουν μεγαλύτερη πιθανότητα εμφάνισης στα σχετικά έγγραφα και πρέπει έτσι να εξεταστούν σαν να είχαν μεγαλύτερη δυνατότητα κατά την αναζήτηση μιας βάσης δεδομένων. Το IDF ορίζεται ως ο λογάριθμος της αναλογίας του αριθμού εγγράφων σε μια συλλογή ανά τον αριθμό εγγράφων που περιέχουν τη δεδομένη λέξη όπως εμφανίζεται παρακάτω.

$$\mathbf{IDF = \log [\ N/d\ ]}$$

όπου,

N	ο αριθμός εγγράφων στη συλλογή , και
d	ο αριθμός εγγράφων που περιέχουν τη δεδομένη λέξη.

Το βάρος μιας λέξης ορίζεται ως η τιμή του IDF. Μετά από τον υπολογισμό των βαρών για τις λέξεις, κατασκευάζονται τα διανύσματα για κάθε υπηρεσία. Το διάνυσμα για κάθε υπηρεσία Si κατασκευάζεται όπως φαίνεται παρακάτω.

$$\mathbf{Si = (Pi,1*W1,\ Pi,2*W2,\ Pi,3*W3,\ ...,\ Pi,n*Wn)}$$

όπου,

Si	το διάνυσμα για την υπηρεσία i
Wj	το βάρος που υπολογίστηκε για τη λέξη j
n	ο αριθμός μοναδικών λέξεων (μετά από την αφαίρεση των stopwords) για όλες τις υπηρεσίες στη συλλογή
Pi,j	παίρνει τιμή 1 ή 0 ανάλογα με το εάν η λέξη j υπάρχει ή όχι στην περιγραφή της υπηρεσίας i

Τα διανύσματα που υπολογίζονται για τις υπηρεσίες χρησιμοποιούνται για να διαμορφώσουν μια μήτρα υπηρεσιών S. Έστω ο αριθμός επιστρεφόμενων υπηρεσιών είναι m, η μήτρα υπηρεσιών S κατασκευάζεται ως $\mathbf{S = [S\ 1, S2, ..., Sm]^T}$ ή

$$\mathbf{S = (si,j\ )_{mxn},\ όπου\ si,j = Pi,j * Wj,\ i \in [\ 1, m\ ],\ j \in [\ 1, n\ ]}$$

Με βάση αυτήν την μήτρα υπηρεσιών S, η Singular Value Decomposition της μήτρας χρησιμοποιείται για να εξαγάγει το πρότυπο σχέσεων μεταξύ των υπηρεσιών και να καθορίσει τα κατώφλια για να βρει τις αντιστοιχημένες υπηρεσίες. Παρακάτω δίνεται η περιγραφή του αλγορίθμου.

Δεδομένου ότι το S είναι μια πραγματική μήτρα, υπάρχει SVD του S: $S=UmxmΣmxnV^Tnxn$, όπου U και V είναι ορθογώνιες μήτρες. Τα U και V μπορούν να γραφούν αντίστοιχα ως $Umxm = [u1, u2, ..., um]_{mxm}$ και $Vnxn = [v1, v2, ..., vn]_{nxn}$, όπου ui (i = 1... , m) είναι το m-διάστατο διάνυσμα $ui = (u1,i, u2,i, ..., um,i)^T$ και vi (i = 1... , n) είναι ένα n-διάστατο διάνυσμα $vi = (v1,i, v2,i, ..., vn,i)^T$. Έστω rank(S)=r και οι μοναδικές τιμές της μήτρας S είναι:

$$σ1 \geq σ2 \geq ... \geq σr > σr+1 = ... = σn = 0$$

Για ένα δεδομένο κατώφλι **ε** ($0 < \mathbf{ε} \leq 1$), επιλέγουμε μια παράμετρο K έτσι ώστε

$(σk - σk+1) / σk \geq \mathbf{ε}$. Κατόπιν δηλώνεται ότι $Uk = [u1,u2, ..., uk\ ]_{mxk}$, $Vk = [v1, v2, ..., vk]_{nxk}$, $Σk = diag\ (σ1, σ2, …,σk)$ και $Sk = UkΣkVk^T$.

Από το θεώρημα της άλγεβρας, η Sk είναι η καλύτερη μήτρα προσέγγισης στο S και περιέχει τις βασικές πληροφορίες μεταξύ των υπηρεσιών. Αυτή η ιδιοκτησία το καθιστά πιθανό να φιλτράρει τις άσχετες υπηρεσίες. Σε αυτόν τον αλγόριθμο, οι αντιστοιχημένες υπηρεσίες μετρώνται από την ομοιότητα μεταξύ τους. Για τη μέτρηση της ομοιότητας υπηρεσιών βασισμένης στο Sk, επιλέγεται η i-οστή σειρά Ri της μήτρας UkΣk ως ισότιμο διάνυσμα της υπηρεσίας i σε ένα k-διάστατο υποδιάνυσμα:

$$\mathbf{Ri = (ui1\sigma1, ui2\sigma2, ..., uik\sigma k), i = 1, 2, ..., m} \quad (1)$$

Οι εξισώσεις (3) χαρτογραφούν τις υπηρεσίες στα διανύσματα στο ίδιο k-διάστατο υποδιάστημα, στο οποίο είναι δυνατό να μετρηθεί η ομοιότητα (βαθμός σχετικότητας) μεταξύ των υπηρεσιών. Γι'αυτό το σκοπό χρησιμοποιείται η μέτρηση ομοιότητας συνημιτόνου, δηλ. για δύο διανύσματα χ = (χ1, χ2, …, χk) και y = (y1, y2, ..., yk) σε ένα k-διάστατο διάστημα, η ομοιότητα μεταξύ τους ορίζεται ως:

$$\text{sim}(\chi, y) = |\chi * y| / (\| \chi\|2 * \|y\|2), \text{ όπου } \chi * y = \Sigma^{k}_{i=1} \chi i * yi, \| \chi\|2 = \sqrt{(\chi * \chi)}$$

Κατ' αυτό τον τρόπο, η ομοιότητα μεταξύ οποιωνδήποτε δύο υπηρεσιών i και j ορίζονται ως:

$$\mathbf{sim'i,j = sim\ (Ri, Rj) = |Ri*Rj| / (||Ri||2*||Rj||2), i,j = 1,2, ..., m} \quad (2)$$

Σύμφωνα με την ομοιότητα μεταξύ οποιωνδήποτε δύο υπηρεσιών (2), για μια επιστρεφόμενη υπηρεσία i, μπορούν να επιλεγούν οι N υπηρεσίες με τις πρώτες N υψηλότερες ομοιότητες για την υπηρεσία i δεδομένου ότι οι αντιστοιχημένες υπηρεσίες για αυτήν την υπηρεσία είναι ένα σύνολο υπηρεσιών MSi όπου MSi = {υπηρεσία j | η υπηρεσία j έχει μια από τις N υψηλότερες ομοιότητες για την υπηρεσία i }.Εδώ ο αριθμός N είναι μια παράμετρος. Η τιμή του θα μπορούσε να επιλεγεί σύμφωνα με τις απαιτήσεις εφαρμογής.

4.3. Αξιολόγηση

Για να γίνει η αξιολόγηση αναζητούνται Υπηρεσίες Διαδικτύου σχετικές με τον όρο "stock" στον κατάλογο UDDI της IBM. Ο προτεινόμενος αλγόριθμος εφαρμόζεται στο σύνολο των Υπηρεσιών Διαδικτύου για να βρει τις αντιστοιχισμένες υπηρεσίες για κάθε επιστρεφόμενη υπηρεσία. Για να πραγματοποιηθεί η σύγκριση, υπολογίζεται η ομοιότητα μεταξύ των υπηρεσιών με βάση τεχνικές αντιστοίχησης με λέξεις-κλειδιά και αυτό δηλώνεται ως αλγόριθμος Non-SVD. Ως εκ τούτου, για τον αλγόριθμο Non-SVD, η ομοιότητα μεταξύ οποιωνδήποτε δύο υπηρεσιών i και j ορίζεται ως:

$$\mathbf{sim'i,j = sim'\ (Si, Sj) = |Si*Sj| / (||Si||2*||Sj||2), i,j = 1, 2, ..., m} \ (3)$$

Ο προτεινόμενος αλγόριθμος δηλώνεται ως βασισμένος στον SVD αλγόριθμος. Για το θέμα "stock", η εγγραφή UDDI της IBM επιστρέφει 47 υπηρεσίες. Μετά από την επεξεργασία των λέξεων που εξάγονται από τις περιγραφές υπηρεσιών, λαμβάνεται ένα σύνολο λέξεων-κλειδιών με 66 μοναδικές λέξεις. Για τις υπηρεσίες που δεν έχουν καμία περιγραφή, χρησιμοποιείται το θέμα ως προκαθορισμένη περιγραφή. Η παράμετρος N τίθεται ίση με 20.

Δεδομένου ότι οι Υπηρεσίες Διαδικτύου είναι μια νέα τεχνολογία, πολλές υπηρεσίες εγγράφονται για δοκιμαστικούς λόγους. Αυτές οι Υπηρεσίες Διαδικτύου συχνά δεν έχουν περιγραφές. Στην προκαταρκτική αξιολόγησή του προτεινόμενου αλγορίθμου, 22 από τις 47 υπηρεσίες δεν είχαν περιγραφή. Οι αναζητούμενες Υπηρεσίες Διαδικτύου ταξινομούνται σε δύο ομάδες: την ομάδα 1 (G1) που περιέχει τις υπηρεσίες με περιγραφή, και την ομάδα 2 (G2) που περιέχει τις υπηρεσίες χωρίς καμία περιγραφή. Στις υπηρεσίες G2 έχουν ανατεθεί οι προκαθορισμένες περιγραφές βασισμένες στο θέμα.

Για τις υπηρεσίες G2, χρησιμοποιείται το "stock" ως περιγραφή τους. Οι αντιστοιχημένες υπηρεσίες που επιστράφηκαν από τον Non-SVD αλγόριθμο και τον βασισμένο στον SVD

αλγόριθμο, για οποιαδήποτε υπηρεσία μέσα στο G2, βρίσκονται επίσης μέσα σε αυτήν την ομάδα. Έπειτα μελετάται η εύρεση αντιστοιχισμένων υπηρεσιών για κάθε υπηρεσία μέσα στο G1. Για μια υπηρεσία στο G1 και το αντιστοιχημένο σύνολο υπηρεσιών, εξετάζονται τα ποσοστά των υπηρεσιών G2 μέσα στα αντιστοιχημένα σύνολα υπηρεσιών. Τα πειραματικά αποτελέσματα για μερικές επιλεγμένες υπηρεσίες στο G1 παρατίθενται στον πίνακα 10.

#	Υπηρεσία	Περιγραφή Υπηρεσίας	Non-SVD (%)	SVDBased (%)
1	http://12.232.105.153/stockquote/client/form.htm	getting stock quote values from yahoo.	35	95
2	http://webservices.garsterworks.com/stocks/stocks.asmx	Retrieve the latest stock quotes. Quotes are delayed 20 minutes.	35	95
3	http://www.webservicex.net/stockquote.asmx?WSDL	Get Stock quote for a company symbol by using this web service	15	90
4	http://gotnet.net.innerhost.com/stock/stock.asmx	Stock Quote Service	20	95
5	http://localhost:8080/soap/servlet/ rpcrouter	Live Quotes From Net-Kraft	0	0
6	http://services.xmethods.net/soap/urn:xmethodsdelayedquotes.wsdl	Obtain stock quotes.	40	100

Πίνακας 10: Ποσοστά υπηρεσιών G2 στα σύνολα αντιστοιχισμένων υπηρεσιών για ορισμένες υπηρεσίες G1 με τον αλγορίθμο Non-SVD και τον βασισμένο στον SVD αλγορίθμο.

Από αυτά τα αποτελέσματα αξιολόγησης, είναι σαφές ότι ο βασισμένος στον SVD αλγόριθμος αθροίζει τις υπηρεσίες που περιέχουν τη θεματική λέξη "stock" μέσα στο σύνολο αντιστοιχημένων υπηρεσιών εάν η δεδομένη υπηρεσία περιέχει αυτήν τη θεματική λέξη, αλλά ο αλγόριθμος Non-SVD δεν το κάνει αυτό. Παραδείγματος χάριν, οι υπηρεσίες στον πίνακα 10 εκτός από την υπηρεσία 5. Για την υπηρεσία 5, δεδομένου ότι η περιγραφή της περιέχει τη θεματική λέξη "stock", δεν υπάρχει καμία υπηρεσία G2 στο αντιστοιχημένο σύνολο υπηρεσιών της. Αυτό το φαινόμενο δείχνει ότι ο βασισμένος στον SVD αλγόριθμος αξιολογεί τη σημασία της λέξης-κλειδιού ή το σφαιρικό βάρος μέσα σε ολόκληρες τις επιστρεφόμενες υπηρεσίες και επιστρέφει τις υπηρεσίες που περιέχουν τις υψηλότερες λέξεις σφαιρικού βάρους εφ' όσον περιέχει επίσης η δεδομένη υπηρεσία τις υψηλότερες λέξεις σφαιρικού βάρους. Σε αυτήν την αξιολόγηση, υπάρχει μόνο μια θεματική λέξη "stock" και περιλαμβάνεται σχεδόν σε κάθε αναζητούμενη υπηρεσία. Επομένως, πρέπει να έχει το μεγαλύτερο σφαιρικό βάρος.

Η αξιολόγηση δείχνει επίσης ότι ο βασισμένος στον SVD αλγόριθμος δεν μετρά την ομοιότητα μεταξύ των υπηρεσιών που βασίζονται μόνο στον αριθμό αντιστοιχισμένων λέξεων-κλειδιών όπως ο αλγόριθμος Non-SVD και το σφαιρικό βάρος της λέξης αξιολογείται μέσα στη διαδικασία SVD. Επομένως, δεν υπάρχει καμία ανάγκη να υπολογιστεί ρητά το σφαιρικό βάρος για κάθε λέξη-κλειδί όπως στη συμβατική ανάκτηση πληροφορίας. Με άλλα λόγια, ο βασισμένος στον SVD αλγόριθμος αποκαλύπτει βαθύτερες σχέσεις μεταξύ των υπηρεσιών και επιστρέφει περισσότερες σημασιολογικά αντιστοιχημένες υπηρεσίες.

5. Σύνθεση Υπηρεσιών Διαδικτύου στον Σημασιολογικό Ιστό

Στη συνέχεια παρουσιάζεται ένα πλαίσιο με βάση οντολογίες για την αυτόματη σύνθεση Υπηρεσιών Διαδικτύου [12]. Προτείνεται μια τεχνική από δηλωτικές περιγραφές υψηλού επιπέδου για την παραγωγή σύνθετων υπηρεσιών και ορίζονται κάποιες οδηγίες για αποτελεσματική σύνθεση με τη χρήση κανόνων. Οι κανόνες αυτοί συγκρίνουν τα συντακτικά και σημασιολογικά χαρακτηριστικά των Υπηρεσιών Διαδικτύου για να καθορίζουν αν δύο Υπηρεσίες Διαδικτύου είναι συνθέσιμες.

Ο Σημασιολογικός Ιστός έχει στόχο να μετατρέψει το Διαδίκτυο σε ένα μέσο στο οποίο τα δεδομένα θα μπορούν να διαμοιραστούν, να κατανοηθούν και να επεξεργαστούν από αυτόματα εργαλεία. Η ανάπτυξη τεχνολογιών που να οδηγούν στην υλοποίηση του Σημασιολογικού Ιστού περιλαμβάνει και τη δημιουργία σημασιολογικά ενεργοποιημένων Υπηρεσιών Διαδικτύου. Στη συνέχεια προσδιορίζονται δύο τύποι υπηρεσιών: απλές και σύνθετες. Οι απλές υπηρεσίες είναι εφαρμογές διαδικτύου που δε βασίζονται σε άλλες Υπηρεσίες Διαδικτύου για να ικανοποιήσουν τις απαιτήσεις του πελάτη. Μια σύνθετη (composite) υπηρεσία ορίζεται ως μια συνεργασία υπηρεσιών (απλών ή/και σύνθετων) που εργάζονται από κοινού για να παρέχουν μια υπηρεσία προστιθέμενης αξίας. Για να πραγματοποιηθεί η σύνθεση υπηρεσιών πρέπει να αναγνωριστεί ο τρόπος που οι υπηρεσίες διασυνδέονται, καλούνται και αντιστοιχίζονται μεταξύ τους.

Η αυτόματη σύνθεση Υπηρεσιών Διαδικτύου, η οποία αναμένεται να παίξει σημαντικό ρόλο στην πραγματοποίηση του Σημασιολογικού Ιστού, περιλαμβάνει την αυτόματη επιλογή και διαλειτουργικότητα των Υπηρεσιών Διαδικτύου. Η διαδικασία της σύνθεσης Υπηρεσιών Διαδικτύου πρέπει να είναι διαφανής στους χρήστες και οι λεπτομερείς περιγραφές των σύνθετων υπηρεσιών θα πρέπει να δημιουργούνται αυτόματα από τις προδιαγραφές του συνθέτη. Η σημασιολογία των Υπηρεσιών Διαδικτύου είναι σημαντική για την αυτόματη σύνθεση υπηρεσιών, καθώς είναι σημαντικό να διασφαλιστεί ότι οι επιλεγμένες υπηρεσίες για τη σύνθεση παρέχουν τα σωστά χαρακτηριστικά. Τέτοια χαρακτηριστικά μπορεί να είναι συντακτικά είτε σημασιολογικά. Για να συλληφθούν τα σημασιολογικά χαρακτηριστικά των Υπηρεσιών Διαδικτύου χρησιμοποιείται στη συνέχεια η έννοια της οντολογίας, η χρήση της οποίας επεκτείνει τη συντακτική συνδεσιμότητα των υπηρεσιών σε σημασιολογική. Τα θέματα που παρουσιάζονται στη συνέχεια είναι τα εξής

– **Ένα μοντέλο συνθεσιμότητας Υπηρεσιών Διαδικτύου**: Ένα καίριο θέμα στην αυτόματη σύνθετη Υπηρεσιών Διαδικτύου είναι εάν αυτές οι υπηρεσίες είναι *συνθέσιμες* (composable) [14]. Η συνθεσιμότητα (*Composability)* αναφέρεται στο εάν οι Υπηρεσίες Διαδικτύου που θα συντεθούν μπορούν πράγματι να αλληλεπιδράσουν μεταξύ τους. Στη συνέχεια προτείνεται ένα μοντέλο συνθεσιμότητας που συγκρίνει συντακτικά και σημασιολογικά χαρακτηριστικά Υπηρεσιών Διαδικτύου.

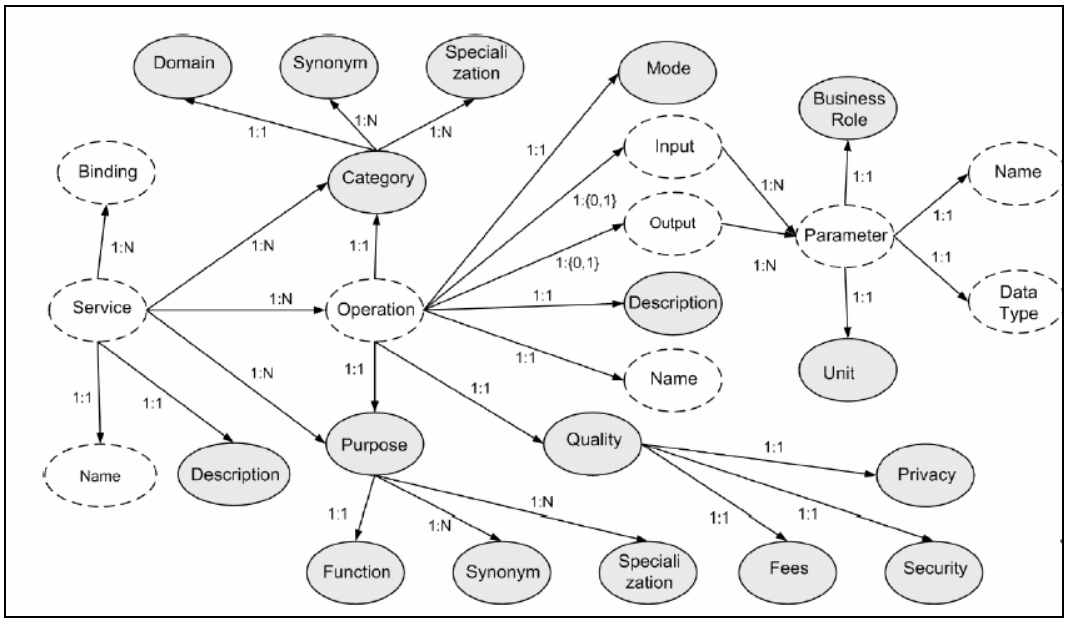
– **Αυτόματη δημιουργία σύνθετων υπηρεσιών**: Προτείνεται μια τεχνική για τη δημιουργία περιγραφών σύνθετων υπηρεσιών, ενώ διατηρούνται οι προαναφερθέντες κανόνες συνθεσιμότητας. Η προτεινόμενη τεχνική χρησιμοποιεί ως είσοδο μια υψηλού επιπέδου προδιαγραφή της επιθυμητής σύνθεσης. Η προδιαγραφή περιέχει τη λίστα λειτουργιών που εκτελούνται μέσω της σύνθεσης χωρίς να αναφέρεται σε κάποια component υπηρεσία.

5.1. Σημασιολογική περιγραφή Υπηρεσιών Διαδικτύου

Η WSDL παρέχει λίγη αν όχι καθόλου υποστήριξη για σημασιολογική περιγραφή των Υπηρεσιών Διαδικτύου. Περιλαμβάνει κυρίως constructs που περιγράφουν τις Υπηρεσίες Διαδικτύου από τη συντακτική πλευρά. Για να ανταποκριθεί σε σημασιολογικά ενεργοποιημένες Υπηρεσίες Διαδικτύου, η WSDL θα πρέπει να επεκταθεί με σημασιολογικές δυνατότητες. Αυτό θα επιτρέψει την αυτόματη επιλογή και σύνθεση Υπηρεσιών Διαδικτύου. Στη συνέχεια ορίζεται μια οντολογία για Υπηρεσίες Διαδικτύου και προδιαγράφεται χρησιμοποιώντας τη νέα γλώσσα DAML+OIL.

Η DAML+OIL υιοθετεί μια αντικειμενοστραφή προσέγγιση, περιγράφοντας ορολογίες αναφορικά με κλάσεις, ιδιότητες και αξιώματα. Βασίζεται σε νεότερα πρότυπα οντολογιών διαδικτύου όπως το RDF και το RDF Schema και επεκτείνει αυτές τις γλώσσες με πλουσιότερα primitives μοντελοποίησης. Άλλες γλώσσες διαδικτυακών οντολογιών, όπως η OWL, μπορεί επίσης να χρησιμοποιηθούν για να καθορίσουν την προτεινόμενη οντολογία. Στην μελέτη που εξετάζεται εδώ μοντελοποιείται η προτεινόμενη οντολογία χρησιμοποιώντας ένα κατευθυνόμενο γράφο όπως φαλινεται στο σχήμα 6. Οι κόμβοι αναπαριστούν τις έννοιες της οντολογίας. Οι μη γεμισμένοι κόμβοι αναφέρονται σε έννοιες της WSDL (όπως όνομα, είσοδος). Οι γκρι κόμβοι αναφέρονται σε επιπλέον χαρακτηριστικά που εισάγονται για να ενισχύσουν τις WSDL περιγραφές με σημασιολογικές δυνατότητες. Οι ακμές αναπαριστούν τις σχέσεις μεταξύ των οντολογικών εννοιών. Για παράδειγμα, η ακμή service→ operation δηλώνει ότι μια υπηρεσία έχει μια ή περισσότερες λειτουργίες. Η ακμή operation→input δηλώνει ότι μια λειτουργία έχει το πολύ ένα μήνυμα εισόδου.

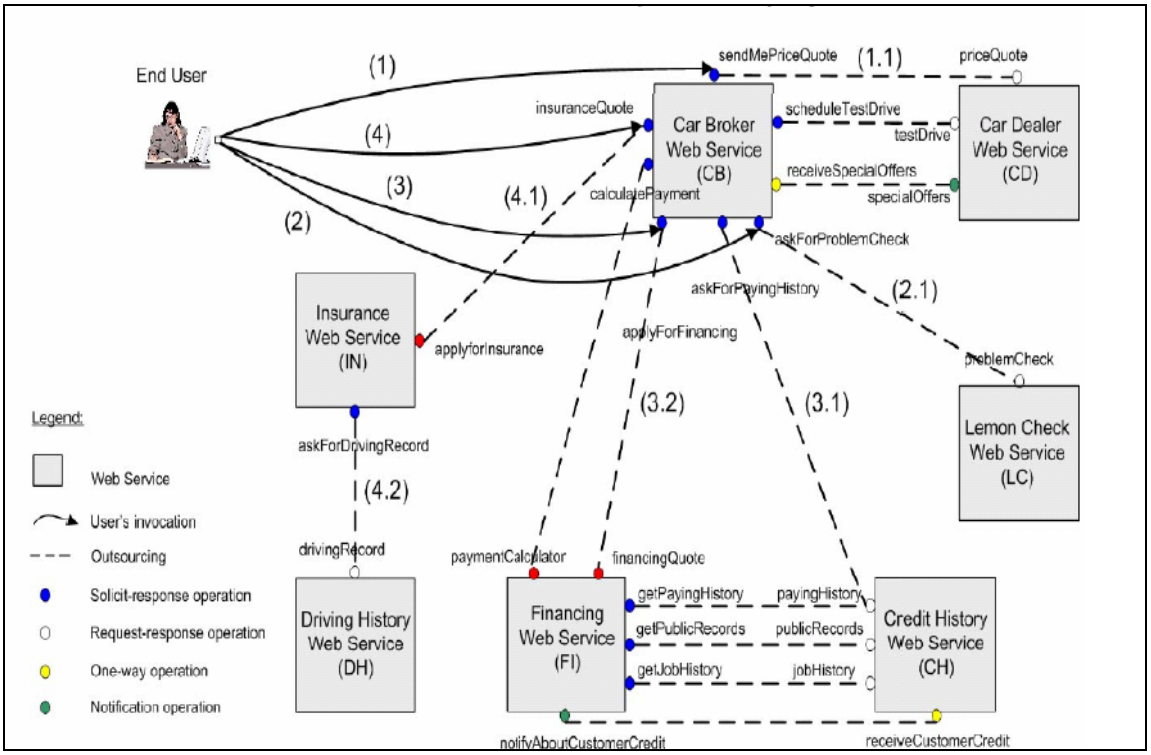
Μια Υπηρεσία Διαδικτύου ορίζεται αρχικοποιώντας κάθε έννοια οντολογίας. Θεωρούνται τρεις τύποι συμμετεχόντων στην προσέγγιση αυτή: οι πάροχοι, οι δημιουργοί και οι καταναλωτές. Οι πάροχοι (providers) είναι οι οντότητες που παρέχουν απλές Υπηρεσίες Διαδικτύου. Ο πάροχος είναι υπεύθυνος για την περιγραφή της Υπηρεσίας Διαδικτύου του αναθέτοντας μια τιμή σε κάθε έννοια οντολογίας. Οι δημιουργοί ευθύνονται για τον καθορισμό των σύνθετων υπηρεσιών. Μόλις δημιουργηθούν, οι περιγραφές σύνθετων υπηρεσιών διαφημίζονται σε ένα κατάλογο έτσι ώστε να μπορούν να ανακαλυφθούν. Οι καταναλωτές μπορεί να είναι οι τελικοί χρήστες ή άλλες Υπηρεσίες Διαδικτύου που καλούν μια Υπηρεσία Διαδικτύου (απλή ή σύνθετη).



Σχήμα 6: Περιγραφή Υπηρεσιών Διαδικτύου με βάση Οντολογίες

Παράδειγμα 1. Ως παράδειγμα θεωρείται μια εφαρμογή μεσίτευσης αυτοκινήτων (Σχήμα 7). Η εταιρία παρέχει μια σύνθετη υπηρεσία car broker (CB) που παρέχει πακέτα πώλησης αυτοκινήτων. Οι πελάτες της εταιρίας υποβάλλουν τις αιτήσεις τους στην CB. Η CB παίρνει αποτελέσματα από άλλες Υπηρεσίες Διαδικτύου για να χειριστεί κάθε αίτημα. Παραδείγματα τέτοιων υπηρεσιών περιλαμβάνουν την insurance(IN), την car dealer (CD), την lemon check (LC), την financing (FI), και την credit history (CH). Ένα κλασσικό σενάριο θα ήταν αυτό στο οποίο ένας πελάτης χρησιμοποιεί την υπηρεσία CB για να αγοράσει ένα αυτοκίνητο ενός συγκεκριμένου μοντέλου, κατασκευής και μιλίων. Ο πελάτης θα άρχιζε καλώντας την λειτουργία sendMePriceQuote της υπηρεσίας CB για να πάρει μια προσφορά τιμής (βήμα 1). Η CB τότε θα

αλληλεπιδρούσε διαφανώς με την car dealer μέσω της λειτουργίας priceQuote της CD (βήμα 1.1). Εάν ενδιαφερόταν για ένα μεταχειρισμένο αυτοκίνητο, ο πελάτης θα έλεγχε το ιστορικό αναφορών του καλώντας την λειτουργία askForProblemCheck της CB (βήμα 2). Αυτή η λειτουργία επεξεργάζεται χρησιμοποιώντας τη λειτουργία problemCheck της υπηρεσίας LC (βήμα 2.1). Ο πελάτης θα έκανε αίτηση για επιδότηση καλώντας τη λειτουργία applyForFinancing που παρέχεται από την CB (βήμα 3). Πριν δεχτεί το οικονομικό σχέδιο, η CB θα έλεγχε το πιστωτικό υπόλοιπο του πελάτη καλώντας τη λειτουργία payingHistory της CH (βήμα 3.1). Ένα το πιστωτικό υπόλοιπο ήταν θετικό, η CB θα καλούσε την λειτουργία financingQuote που παρέχεται από την υπηρεσία financing (βήμα 3.2). Τέλος, ο πελάτης θα ζητούσε μια αναφορά ασφάλισης μέσο της λειτουργίας insuranceQuote της CB (βήμα 4). Η CB θα καλούσε διαφανώς την λειτουργία applyforInsurance που παρέχεται από την υπηρεσία insurance (βήμα 4.1). Η υπηρεσία θα outsource από την λειτουργία drivingRecord της DH πριν εκδόσει αναφορές ασφάλισης (βήμα 4.2).



Σχήμα 7: Εφαρμογή μεσίτευσης αυτοκινήτων

5.1.1. Τρόποι λειτουργίας και μηνύματα

Οι λειτουργικότητες που παρέχονται από την Υπηρεσία Διαδικτύου είναι προσπελάσιμες μέσω της κλήσης λειτουργιών. Θεωρούνται τέσσερις τρόποι λειτουργίας: Ανακοίνωση (notification), Μονόδρομη (one-way), Απαίτηση-απάντηση (solicit-response), και Αίτημα-απάντηση (request-response). Μια λειτουργία ανακοίνωσης στέλνει ένα μήνυμα εξόδου αλλά δεν περιμένει να λάβει κάποιο μήνυμα απάντησης. Στην μονόδρομη λειτουργία, η υπηρεσία λαμβάνει ένα μήνυμα εισόδου, το καταναλώνει αλλά δεν παράγει κάποιο μήνυμα εξόδου. Σε μια λειτουργία απαίτησης-απάντησης, η υπηρεσία δημιουργεί ένα μήνυμα εξόδου και λαμβάνει σε απάντηση ένα μήνυμα εισόδου. Σε μια αιτήματος-απάντησης λειτουργία, η υπηρεσία λαμβάνει ένα μήνυμα εισόδου, το επεξεργάζεται και στέλνει ένα σχετικό μήνυμα εξόδου. Η CB::sendMePriceQuote είναι ένα παράδειγμα λειτουργίας απαίτησης-απάντησης. Οι είσοδοί αυτής της λειτουργίας περιλαμβάνουν τις παραμέτρους VIN (αριθμός κυκλοφορίας αυτοκινήτου)

και price (τιμή). Το μήνυμα εισόδου της περιέχει τέσσερις παραμέτρους: make (κατασκευή), model (μοντέλο), year (έτος), και mileage(μίλια). Η FI::paymentCalculator είναι ένα παράδειγμα μιας λειτουργίας αιτήματος-απάντησης. Οι είσοδοί της περιλαμβάνουν τις παραμέτρους purchasePrice, downPayment, και loanTerm. Το μήνυμα εξόδου αυτής της λειτουργίας περιλαμβάνει τις παραμέτρους interestRateand και monthlyPayment. Η CD::specialOffers είναι μια λειτουργία ανακοίνωσης της οποίας οι έξοδοι περιλαμβάνουν τις τιμές make (κατασκευή), model (μοντέλο), year (έτος), και mileage(μίλια) και price (τιμή). Για παράδειγμα, η λειτουργία CH::receiveCustomerCredit είναι μια μονόδρομη λειτουργία η οποία παίρνει σαν είσοδο τις παραμέτρους firstName (Όνομα), lastName(Επώνυμο), SSN(Αριθμός Κοινωνικής Ασφάλισης), dateOfBirth (Ημερομηνία γέννησης), και creditInformation (Πληροφορίες πιστωτικού ορίου).

Μια λειτουργία έχει ένα μήνυμα εισόδου και/ή εξόδου ανάλογα με τον τρόπο λειτουργίας της. Οι λειτουργίες αίτησης-απάντησης και απαίτησης-απάντησης έχουν και μηνύματα εισόδου και εξόδου. Οι λειτουργίες ανακοίνωσης (αντίστοιχα οι μονόδρομες) έχουν μόνο μηνύματα εξόδου (αντίστοιχα εισόδου). Κάθε μήνυμα περιέχει μια ή περισσότερες παραμέτρους (που καλούνται parts στην WSDL). Μια παράμετρος έχει ένα όνομα και ένα τύπο δεδομένων. Ο τύπος δεδομένων δίνει το πεδίο τιμών που μπορούν να ανατεθούν στην παράμετρο. Χρησιμοποιούνται οι built-in τύποι δεδομένων του XML Schema ως το typing σύστημα. Οι ενσωματωμένοι τύποι είναι προκαθορισμένοι στις προδιαγραφές του XML Schema. Μπορούν να είναι είτε primitive είτε derived. Αντίθετα με τους primitive τύπους (π.χ. αλφαριθμητικο, δεκαδικός), οι derived τύποι ορίζονται με βάση άλλους τύπους. Για παράδειγμα, ένας ακέραιος προκύπτει από τον δεκαδικό primitive τύπο. Παρόλο που οι τύποι δεδομένων είναι σημαντικοί για το αυτόματο ταίριασμα παραμέτρων των μηνυμάτων, δεν συλλαμβάνουν τη σημασιολογία αυτών των παραμέτρων. Για παράδειγμα, η παράμετρος τιμή μπορεί να μετράται σε δολάρια, ευρώ ή λίρες. Επιπλέον μπορεί να αναπαριστά μια συνολική τιμή ή τιμή χωρίς φόρους.

Για να μοντελοποιηθούν τέτοιοι περιορισμοί, σε κάθε παράμετρο συσχετίζεται μια μονάδα και ένας επιχειρησιακός ρόλος. Χρησιμοποιούνται πρότυπες μονάδες μέτρησης (μήκος, εμβαδόν, βάρος κλπ) για να ανατεθούν τιμές στις μονάδες των παραμέτρων. Εάν μια παράμετρος δεν έχει μονάδα (π.χ. firstName), η μονάδα μέτρησής της είναι ίση με «καμία» (none). Ο επιχειρησιακός ρόλος δίνει τη σημασιολογία της αντίστοιχης παραμέτρου. Παίρνει την τιμή του από μια προκαθορισμένη ταξονομία για επιχειρησιακούς ρόλους. Κάθε παράμετρος θα έχει μια καλά ορισμένη (well-defined) σημασιολογία σύμφωνα με αυτήν την ταξονομία. Ένα παράδειγμα τέτοιας ταξονομίας είναι το επιχειρησιακό λεξικό της RosettaNet. Περιέχει ένα κοινό λεξιλόγιο που μπορεί να χρησιμοποιηθεί για να περιγράψει επιχειρησιακές ιδιότητες. Για παράδειγμα, εάν η παράμετρος price έχει ένα ρόλο “extendedPrice” (που ορίζεται στο RosettaNet), τότε αναπαριστά μια «συνολική τιμή για μια ποσότητα προϊόντος». Για σκοπούς ευελιξίας, διαφορετικές Υπηρεσίες Διαδικτύου μπορεί να υιοθετούν διαφορετικές ταξονομίες για να καθορίσουν τους επιχειρησιακούς ρόλους των παραμέτρων τους. Εδώ χρησιμοποιούνται τα XML namespaces ως προθέματα για τους επιχειρησιακούς ρόλους με την ταξονομία στην οποία ορίζονται

Ορισμός 1

Μήνυμα. Ένα μήνυμα M ορίζεται ως μια πλειάδα (P,T ,U,R) όπου:

- Το **P** είναι ένα σύνολο από ονόματα παραμέτρων.
- **T: P → DataTypes**, είναι μια συνάρτηση που αναθέτει σε κάθε παράμετρο ένα τύπο δεδομένων. Οι DataTypes είναι ένα σύνολο από XML τύπους δεδομένων.
- **U: P → Units**, είναι μια συνάρτηση που δίνει τη μονάδα μέτρησης που χρησιμοποιείται για κάθε παράμετρο. Το Units είναι μια ταξονομία για μονάδες μέτρησης.
- **R: P → Roles**, είναι μια συνάρτηση που αναθέτει έναν επιχειρησιακό ρόλο σε κάθε παράμετρο. Το Roles είναι μια ταξονομία για επιχειρησιακούς ρόλους.

5.1.2. Σκοπός και κατηγορία

Κάθε λειτουργία περιγράφεται σημασιολογικά μέσο του σκοπού (purpose) και της κατηγορίας (category). Ο σκοπός περιλαμβάνει τρεις ιδιότητες: τη συνάρτηση (function), τα συνώνυμα (synonyms), και την ειδίκευση (specialization). Η συνάρτηση δίνει τη λειτουργικότητα που παρέχεται από τη λειτουργία. Παραδείγματα συναρτήσεων περιλαμβάνουν τις "request_for_quotation", "purchase_order", και "delivery_order". Όσο για τους επιχειρησιακούς ρόλους των παραμέτρων, μπορεί να χρησιμοποιηθεί ένα ευρύ σύνολο από ταξονομίες για να ορίσει την ιδιότητα συνάρτησης. Παραδείγματα τέτοιων ταξονομιών περιλαμβάνουν το RosettaNet, το cXML (commerceXML), και το EDI (electronic data interchange) [13]. Κάθε σκοπός έχει πρόθεμα ένα XML namespace που δείχνει στην αντίστοιχη ταξονομία. Για παράδειγμα, ο σκοπός μιας λειτουργίας μπορεί να επεξεργασθεί από το namespace της cXML ταξονομίας. Η ιδιότητα *συνώνυμα* περιέχει το σύνολο των εναλλακτικών ονομάτων συναρτήσεων για τη λειτουργία. Για παράδειγμα, "quotation" είναι συνώνυμο του "request for quotation". Η ιδιότητα *ειδίκευση* ορίζει ένα σύνολο από χαρακτηριστικά της τρέχουσας συνάρτησης. Για παράδειγμα, μια "request for quotation" μπορεί να είναι "for your information" ή "legally binding". Μπορεί επίσης να περιλαμβάνει "destination charge" είτε όχι, να είναι "valid until a specific date", κλπ. Παρακάτω συνοψίζεται η έννοια του σκοπού λειτουργίας :

Ορισμός 2

Σκοπός. Ο σκοπός μιας λειτουργίας op_{ik} ορίζεται από την πλειάδα (Συνάρτηση, Συνώνυμα, Ειδίκευση) όπου *Συνάρτηση* είναι η επιχειρησιακή λειτουργικότητα της op_{ik} που ορίζεται σε μια ταξονομία, *Συνώνυμα* είναι ένα σύνολο από εναλλακτικά ονόματα συνάρτησης και *Ειδίκευση* είναι ένα σύνολο από ιδιότητες της συνάρτησης op_{ik}.

Κατηγορία. Η κατηγορία μιας λειτουργίας ορίζεται με τον ίδιο τρόπο όπως ο σκοπός. Κάθε κατηγορία περιέχει τρεις ιδιότητες: περιοχή, συνώνυμα, και ειδίκευση. Η *Περιοχή* δίνει την περιοχή ενδιαφέροντος για την τρέχουσα λειτουργία. Η ταξονομίες όπως η NAICS (North American Industry Classification System) και η UNSPSC (Universal Standard Products and Services Classification) μπορούν να χρησιμοποιηθούν για να ορίσουν την ιδιότητα περιοχής. Οι ιδιότητες *Συνώνυμα* και *Ειδίκευση* λειτουργούν όπως στον σκοπό της λειτουργίας. Συνώνυμα της περιοχής "automobile dealers" περιλαμβάνουν τα "car dealers" και "car sellers".Ένα παράδειγμα ιδιότητας ειδίκευσης που σχετίζεται με "insurance" είναι {"car","home"}. Αυτό σημαίνει ότι η αντίστοιχη λειτουργία παρέχει τόσο ασφάλιση αυτοκινήτων όσο και σπιτιών. Παρακάτω ορίζεται η έννοια της κατηγορίας:

Ορισμός 3

Κατηγορία. Η κατηγορία μιας λειτουργίας opik ορίζεται με μια πλειάδα (Περιοχή, Συνώνυμα, Ειδίκευση) όπου η Περιοχή είναι η περιοχή ενδιαφέροντος της opik που ορίζεται σε μια ταξονομία. Συνώνυμα είναι ένα σύνολο από εναλλακτικές περιοχές, και Ειδίκευση είναι ένα σύνολο χαρακτηριστικών της περιοχής της opik.

5.1.3. Ποιότητα Λειτουργίας

Αρκετές Υπηρεσίες Διαδικτύου μπορεί να παρέχουν «παρόμοιες» λειτουργίες όσον αφορά τον τρόπο λειτουργίας, το μήνυμα, τον σκοπό και την κατηγορία. Είναι επομένως σημαντικό να οριστούν οι ποιοτικές ιδιότητες που βοηθούν το δημιουργό να επιλέξει την καλύτερη Υπηρεσία Διαδικτύου. Ορίζονται τρεις ποιοτικές ιδιότητες για λειτουργίες: το *κόστος* (fee), η *ασφάλεια* (security), και η *μυστικότητα* (privacy). Άλλες ιδιότητες όπως ο *χρόνος* και η *διαθεσιμότητα* μπορούν επίσης να προστεθούν. Η ιδιότητα κόστους δείχνει το χρηματικό ποσό που απαιτείται για την εκτέλεση της λειτουργίας. Η ασφάλεια και η μυστικότητα είναι δύο σημαντικές απαιτήσεις των Υπηρεσιών Διαδικτύου. Οι επιχειρήσεις συλλέγουν, αποθηκεύουν, επεξεργάζονται και μοιράζονται πληροφορίες σχετικά με εκατομμύρια χρήστες που έχουν

διαφορετικές προτιμήσεις σχετικά με την μυστικότητα και την ασφάλεια των πληροφοριών τους. Ειδικά η μυστικότητα και η ασφάλεια είναι ιδιαίτερα σημαντικές για εφαρμογές ηλεκτρονικής διακυβέρνησης όπου οι πολίτες είναι ευαίσθητοι με τα προσωπικά δεδομένα τους. Η ιδιότητα ασφάλειας είναι μια boolean που δείχνει εάν τα μηνύματα της λειτουργίας ανταλλάσσονται με ασφάλεια μεταξύ εξυπηρετητών και πελατών. Η ιδιότητα μυστικότητα περιέχει τις παραμέτρους εισόδου και εξόδου που δεν θα πρέπει να κοινοποιηθούν σε εξωτερικές οντότητες (εκτός από τον πάροχο υπηρεσίας). Εάν η παράμετρος δεν ανήκει στο σύνολο privacy, τότε δεν καθορίζεται περιορισμός μυστικότητας σε αυτήν την παράμετρο. Έστω ότι privacy = {SSN, Credit Card Number} όπου SSN και Credit Card Number είναι δύο παράμετροι εισόδου. Αυτή η ιδιότητα δηλώνει ότι αυτές οι παράμετροι κρατούνται μυστικές από τον πάροχο υπηρεσίας. Με βάση τις προηγούμενες ιδιότητες ορίζεται παρακάτω η έννοια της ποιότητας λειτουργίας:

Ορισμός 4

Ποιότητα. Η ποιότητα μιας λειτουργίας op_{ik} ορίζεται από μια πλειάδα ($Fees_{ik}$, $Security_{ik}$, $Privacy_{ik}$). Η $Fees_{ik}$ είναι το ποσό που απαιτείται για να εκτελεστεί η op_{ik}. Η $Security_{ik}$ επιστρέφει σωστό ή λάθος ανάλογα με το εάν τα μηνύματα της op_{ik} ανταλλάσσονται με ασφάλεια. Η $Privacy_{ik}$ είναι ένα σύνολο παραμέτρων εισόδου και εξόδου που δεν κοινοποιούνται σε εξωτερικές οντότητες.

5.1.4. Καθορίζοντας λειτουργίες και Υπηρεσίες Διαδικτύου

Οι Υπηρεσίες Διαδικτύου προσπελαύνονται μέσω λειτουργιών. Κάθε λειτουργία ορίζεται από ένα όνομα και ένα κείμενο περιγραφής που συγκεντρώνει τα χαρακτηριστικά των λειτουργιών. Επίσης έχει ένα τρόπο λειτουργίας, μηνύματα εισόδου και/ή εξόδου, σκοπό και κατηγορία. Παρακάτω παρουσιάζεται ο ορισμός μιας λειτουργίας υπηρεσίας:

Ορισμός 5

Λειτουργία. Μια λειτουργία op_{ik} ορίζεται από μια πλειάδα ($Description_{ik}$, $Mode_{ik}$, In_{ik}, Out_{ik}, $Purpose_{ik}$,$Category_{ik}$, $Quality_{ik}$) όπου:

- $Description_{ik}$ είναι το αθροιστικό κείμενο σχετικά με τα χαρακτηριστικά της λειτουργίας.
- $Mode_{ik} \in$ {“one-way”, “notification”, “solicit-response”, “request-response”}.
- In_{ik} και Out_{ik} είναι τα μηνύματα εισόδου και εξόδου αντίστοιχα
- $In_{ik}= (\emptyset,T_{ik})$ και $Out_{ik} = (\emptyset,T_{ik})$ για λειτουργίες ανακοίνωσης και μονόδρομες αντίστοιχα.
- $Purpose_{ik}$ περιγράφει την επιχειρησιακή λειτουργία που παρέχεται από την λειτουργία (Ορισμός 2).
- $Category_{ik}$ περιγράφει την περιοχή ενδιαφέροντος της λειτουργίας (Ορισμός 3)
- Qualityik δίνει τις ποιοτικές ιδιότητες της λειτουργίας (Ορισμός 4).

Μια Υπηρεσία Διαδικτύου ορίζεται από ένα *όνομα* και ένα κείμενο *περιγραφής* που συνοψίζει *τα* χαρακτηριστικά της υπηρεσίας. Οι αλληλεπιδράσεις με την υπηρεσία εκτελούνται σύμφωνα με μια συγκεκριμένη *σύμβαση* (binding). Η σύμβαση ορίζει τη μορφή των μηνυμάτων και τις λεπτομέρειες του πρωτοκόλλου για τις λειτουργίες της υπηρεσίας και τα μηνύματα. Παραδείγματα συμβάσεων περιλαμβάνουν το *SOAP*, το *HTTP Get/Post* και το MIME. Μια υπηρεσία μπορεί να έχει αρκετές συμβάσεις που να σχετίζονται με αυτή. Σε κάθε Υπηρεσία Διαδικτύου, επίσης συσχετίζεται ένας *σκοπός* και μια *κατηγορία*. *Ο σκοπός* περιγράφει τις επιχειρησιακές λειτουργικότητες που παρέχονται από τις λειτουργίες της υπηρεσίας. Κάθε στοιχείο στο *σκοπό* της υπηρεσίας αναφέρεται στην επιχειρησιακή λειτουργικότητα που παρέχεται από μια συγκεκριμένη λειτουργία.

Η κατηγορία περιγράφει την περιοχή ενδιαφέροντος της υπηρεσίας. Περιλαμβάνει τις κατηγορίες όλων των λειτουργιών που παρέχονται από την υπηρεσία. Επίσης περιέχει ένα στοιχείο που αντιστοιχεί στην κατηγορία της υπηρεσίας. Πράγματι, η περιοχή ενδιαφέροντος μιας σύνθετης υπηρεσίας μπορεί να είναι διαφορετική από τις περιοχές ενδιαφέροντος των

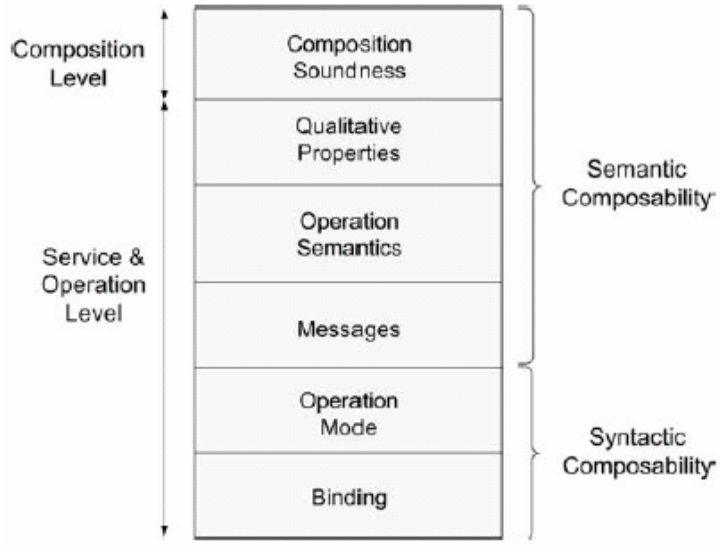
λειτουργιών της. Για παράδειγμα, η σύνθετη υπηρεσία car dealer σχετίζεται με την βιομηχανία "automobile dealers". Ακόμα περιλαμβάνει λειτουργίες σχετικές και με την «insurance» (π.χ. insuranceQuote). Παρακάτω δίνεται ένας επίσημος ορισμός για μια Υπηρεσία Διαδικτύου (σύνθετη ή απλή):

Ορισμός 6

Υπηρεσία Διαδικτύου. Μια Υπηρεσία Διαδικτύου WS_i. ορίζεται από μια πλειάδα ($Description_i$, OP_i , $Bindings_i$, $Purpose_i$, $Category_i$) όπου:

- $Description_i$ είναι ένα συνοπτικό κείμενο για τα χαρακτηριστικά της υπηρεσίας.
- OP_i είναι ένα σύνολο από λειτουργίες που παρέχονται WS_i.
- $Bindings_i$ είναι ένα σύνολο από binding πρωτόκολλα που υποστηρίζονται από την WS_i.
- $Purpose_i = \{Purpose_{ik} (op_{ik}) \mid op_{ik} \in OP_i\}$ είναι ένα σύνολο από σκοπούς των λειτουργιών της WS.
- $Category_i = \{Category_{ik} (op_{ik}) \mid op_{ik} \in OP_i\} \cup \{Categoryi (WS_i)\}$ είναι ένα σύνολο από κατηγορίες των λειτουργιών της WS_i.

Παράδειγμα. Έστω η υπηρεσία car dealer (CD) που απεικονίζεται στο Σχήμα 8. Αυτή η υπηρεσία ορίζεται από μια πλειάδα (Desc, OP, B, P, C), όπου OP = {priceQuote, testDrive, specialOffers}, και B = {"SOAP"}. P είναι ο σκοπός της υπηρεσίας που ορίζεται από το σύνολο {purpose(priceQuote), purpose(testDrive), purpose(specialOffers)}. C είναι η κατηγορίας της υπηρεσίας που ορίζεται από το σύνολο {category(priceQuote), category(testDrive), category(specialOffers)} $\cup$ {category(CD)}. Το στοιχείο category(CD) ορίζεται ως εξής: category(CD).domain = "automobile dealers", category(CD).synonyms = {"car dealers", "car sellers"}, and category(CD).specialization = {"used cars"}.



Σχήμα 8: Μοντέλο συνθεσιμότητας για Υπηρεσίες Διαδικτύου

5.2. Μοντέλο Συνθεσιμότητας για Υπηρεσίες Διαδικτύου

Ένα σημαντικό θέμα όταν ορίζεται μια σύνθετη υπηρεσία είναι εάν οι υπηρεσίες που την αποτελούν έχουν δυνατότητα σύνθεσης. Για παράδειγμα, θα ήταν δύσκολο να κληθεί μια υπηρεσία εάν δεν υπήρχε αντιστοιχία μεταξύ των παραμέτρων που ζητούνται από την λειτουργία (π.χ. τύποι δεδομένων, αριθμός παραμέτρων) και αυτών που μεταδίδονται από την υπηρεσία πελάτη. Στη συνέχεια προσδιορίζονται δύο σύνολα κανόνων συνθεσιμότητας για να συγκρίνουν τις συντακτικές και σημασιολογικές ιδιότητες των Υπηρεσιών Διαδικτύου. Οι συντακτικοί

κανόνες περιλαμβάνουν: (1)συνθεσιμότητα τρόπου λειτουργίας (mode composability), που συγκρίνει τους τρόπους λειτουργίας και (2) συνθεσιμότητα σύνδεσης (binding composability), που συγκρίνει τα πρωτόκολλα σύμβασης των υπηρεσιών που αλληλεπιδρούν. Οι σημασιολογικοί κανόνες περιλαμβάνουν (1) συνθεσιμότητα μηνύματος (message composability), που συγκρίνει τον αριθμό των παραμέτρων των μηνυμάτων, τους τύπους δεδομένων τους, τους επιχειρησιακούς ρόλους και τις μονάδες (2) σημασιολογική συνθεσιμότητα υπηρεσίας (operation semantics composability), που συγκρίνει τη σημασιολογία των λειτουργιών των υπηρεσιών, (3) ποιοτική συνθεσιμότητα (qualitative composability), που συγκρίνει τις ποιοτικές ιδιότητες των Υπηρεσιών Διαδικτύου, και (4) αρτιότητα σύνθεσης (composition soundness), που ελέγχει εάν αξίζει ο συνδυασμός Υπηρεσιών Διαδικτύου με έναν συγκεκριμένο τρόπο. Η αρτιότητα σύνθεσης (composition soundness) ελέγχει την συνθεσιμότητα στο επίπεδο δημιουργίας, αντίθετα με άλλους κανόνες που αντιμετωπίζουν την συνθεσιμότητα στα επίπεδα υπηρεσίας και λειτουργίας. Στο μοντέλο που παρουσιάζεται εδώ οι πελάτες και οι εξυπηρετητές αναφέρονται στις outsourcing και outsourced υπηρεσίες αντίστοιχα. Υιοθετείται η προσέγγιση που ορίζεται στα πρότυπα XLANG, WSFL,και BPEL4WS, δηλαδή οι Υπηρεσίες Διαδικτύου τόσο σε πελάτες όσο και σε εξυπηρετητές ορίζονται σε WSDL επαυξημένη με σημασιολογικές ιδιότητες. Όπως και σε αυτά τα πρότυπα, κάθε λειτουργία στην πλευρά του εξυπηρετητή έχει το πολύ μια ταιριαστή λειτουργία στην πλευρά του πελάτη και αντίστροφα.

5.2.1. Τρόπος λειτουργίας και συνθεσιμότητα σύνδεσης

Για να πραγματοποιηθούν οι αλληλεπιδράσεις των Υπηρεσιών Διαδικτύου, οι υπηρεσίες πελάτη και εξυπηρετητή πρέπει να έχουν «διπλούς» τρόπους λειτουργίας. Μια λειτουργία ανακοίνωσης σε μια υπηρεσία πρέπει να είναι συνδεδεμένη με μια μονόδρομη λειτουργία στην συνεργατική υπηρεσία. Παρόμοια, μια λειτουργία απαίτησης-απάντησης αντιστοιχεί σε μια λειτουργία αίτησης-απάντησης σε μια συνεργατική υπηρεσία. Για παράδειγμα, η λειτουργία CB::sendMePriceQuote (απαίτησης-απάντησης) αντιστοιχεί στην CD::priceQuote (αίτησης-απάντησης), και η CB::receiveSpecialOffers (μονόδρομη) αντιστοιχεί στην CD::specialOffers (ανακοίνωσης). Ο ακόλουθος κανόνας που καλείται συνθεσιμότητα τρόπου λειτουργίας, ελέγχει εάν δύο λειτουργίες έχουν «διπλούς» τρόπους λειτουργίας.

Ορισμός 7

Συνθεσιμότητα τρόπου λειτουργίας. Οι λειτουργίες op_{ik}=(Dik, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}, Q_{ik}) και op_{jl}=(D_{jl}, M_{jl}, In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl}) έχουν συνθεσιμότητα τρόπου λειτουργίας εάν
(i) M_{ik} = «ανακοίνωση» και M_{jl} = «μονόδρομη», ή
(ii) M_{ik} = «μονόδρομη» και M_{jl} = «ανακοίνωση», ή
(iii) M_{ik} = «απαίτηση-απάντηση» και M_{jl} = «αίτηση-απάντηση», ή
(iv) M_{ik} = «αίτηση-απάντηση» και M_{jl} = «απαίτηση-απάντηση».

Έστω τώρα ότι δύο Υπηρεσίες Διαδικτύου επικοινωνούν μέσω λειτουργιών που έχουν συνθεσιμότητα τρόπου λειτουργίας. Εφόσον αυτές οι Υπηρεσίες Διαδικτύου μπορεί να υποστηρίζουν διαφορετικά πρωτόκολλα σύνδεσης (SOAP, HTTP, ή MIME), είναι σημαντικό να διασφαλισθεί ότι «καταλαβαίνονται» μεταξύ τους στη μορφή μηνυμάτων και στο επίπεδο πρωτοκόλλου. Τουλάχιστον ένα από τα πρωτόκολλα που αναμένονται από τη μια Υπηρεσία Διαδικτύου πρέπει να υποστηρίζεταισε από την άλλη. Για παράδειγμα, θα ήταν δύσκολο για μια υπηρεσία που περιμένει να λάβει μηνύματα στο MIME πρωτόκολλο να αλληλεπιδράσει με μια άλλη υπηρεσία που διαμορφώνει τα μηνύματά της σε HTTP. Ο ακόλουθος κανόνας, που καλείται συνθεσιμότητα σύνδεσης, ελέγχει ότι οι Υπηρεσίες Διαδικτύου υποστηρίζουν τουλάχιστον ένα κοινό πρωτόκολλο σύνδεσης.

Ορισμός 8

Συνθεσιμότητα σύνδεσης. Δύο υπηρεσίες $WS_i = (D_i, O_i, B_i, P_i, C_i)$ και $WS_j = (D_j, O_j, B_j, P_j, C_j)$ έχουν συνθεσιμότητα σύνδεσης εάν $B_i \cap B_j \neq \emptyset$

5.2.2. Συνθεσιμότητα μηνυμάτων

Οι αλληλεπιδράσεις μεταξύ Υπηρεσιών Διαδικτύου περιλαμβάνουν την ανταλλαγή μηνυμάτων. Ένα μήνυμα αποτελείται από μια ή περισσότερες παραμέτρους, καθεμία από τις οποίες έχει διαφορετικό τύπο δεδομένων. Επομένως είναι σημαντικό να ελεγχθεί εάν οι τύποι δεδομένων των παραμέτρων που στέλνονται από την υπηρεσία είναι συμβατοί με τους τύπους δεδομένων των παραμέτρων που λαμβάνονται από την υπηρεσία συνεργάτη της. Θεωρούνται δύο πρωταρχικές μέθοδοι συμβατότητας τύπων δεδομένων: η άμεση και η έμμεση συμβατότητα. Δύο παράμετροι είναι άμεσα συμβατές εάν έχουν τον ίδιο τύπο δεδομένων. Μια παράμετρος p είναι έμμεσα συμβατή με μια p' εάν ο τύπος της p προκύπτει από τον τύπο της p'. Για παράδειγμα, μια παράμετρος τύπου θετικού ακεραίου είναι έμμεσα συμβατή με μια ακέραια παράμετρο. Σημειώνεται επίσης ότι αντίθετα με την άμεση συμβατότητα, ή έμμεση συμβατότητα είναι ασύμμετρη.

Η έννοια της συμβατότητας τύπων δεδομένων επεκτείνεται στα μηνύματα ως εξής: Ένα μήνυμα M έχει συμβατό τύπο δεδομένων, με ένα μήνυμα M' εάν κάθε παράμετρος του M είναι άμεσα ή έμμεσα συμβατή με μια παράμετρο του M'. Σημειώνεται ότι δεν είναι απαραίτητο να αντιστοιχίζονται όλες οι παράμετροι του M' στις παραμέτρους του M. Ο λόγος είναι ότι ένα μήνυμα εισόδου μιας λειτουργίας υπηρεσίας μπορεί να χρησιμοποιεί μόνο ένα υποσύνολο των παραμέτρων που στέλνονται μέσω ενός μηνύματος εξόδου μιας «διπλής» λειτουργίας. Για παράδειγμα, έστω ότι ο πάροχος της car broker δεν ενδιαφέρεται να γνωρίζει το χρώμα των αυτοκινήτων που διαφημίζονται ως ειδικές προσφορές. Σε αυτήν την περίπτωση, ορίζει το μήνυμα εισόδου της CB::receiveSpecialOffers σαν να αποτελείται από τις εξής παραμέτρους: "make", "model","year", "mileage", "price". Το μήνυμα εισόδου είναι συμβατό με το μήνυμα εξόδου της CD::specialOffers, παρόλο που το μήνυμα εξόδου περιέχει μια επιπλέον παράμετρο.

Έστω ότι η παράμετρος p είναι συμβατή (άμεσα ή έμμεσα) με μια παράμετρο p'. Για να αντιστοιχιστούν μεταξύ τους οι p και p', πρέπει επίσης να έχουν συμβατές σημασιολογίες. Για παράδειγμα, μια συνολική τιμή δε θα πρέπει να αντιστοιχιστεί με μια τιμή πριν τους φόρους. Ανάλογα, μια τιμή σε δολάρια δε θα πρέπει να αντιστοιχιστεί σε μια τιμή σε ευρώ. Τέλος συγκρίνονται οι μονάδες και οι επιχειρησιακοί ρόλοι των p και p'. Και οι δύο παράμετροι θα πρέπει να έχουν την ίδια μονάδα και τους ίδιους επιχειρησιακούς ρόλους. Ο κανόνας συμβατότητας μηνυμάτων συγκρίνει τα μηνύματα εισόδου και εξόδου για κάθε ζεύγος λειτουργιών. Η ιδέα είναι να ελεγχθεί ότι κάθε είσοδος μιας λειτουργίας έχει συμβατό τύπο δεδομένων με την έξοδο της άλλης λειτουργίας. Η μονάδα και ο επιχειρησιακός ρόλος εισόδου θα πρέπει να είναι ίδιοι με της εξόδου. Αυτό σημαίνει ότι οι παράμετροι κάθε μηνύματος εισόδου αντιστοιχούν σε όλες ή κάποιες παραμέτρους που περιέχονται στο μήνυμα εξόδου της άλλης λειτουργίας:

Ορισμός 9

Συνθεσιμότητα μηνυμάτων. Δύο λειτουργίες $op_{ik} = (D_{ik}, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}, Q_{ik})$ και $op_{jl}=(D_{jl}, M_{jl}, In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl})$ έχουν συνθεσιμότητα μηνυμάτων εάν:

1. $\forall\ p \in In_{ik}, \exists\ p' \in In_{jl}$ | p έχει συμβατό τύπο δεδομένων με την p', και $U(p) = U(p')$ και $R(p) = R(p')$.
2. $\forall\ p \in In_{jl}, \exists\ p' \in In_{ik}$ | p έχει συμβατό τύπο δεδομένων με την p', και $U(p) = U(p')$, και $R(p') = R(p')$.

5.2.3. Σημασιολογική συνθεσιμότητα λειτουργιών

Αυτός ο κανόνας βεβαιώνει ότι οι διασυνδεδεμένες λειτουργίες έχουν συμβατούς σκοπούς και κατηγορίες. Για παράδειγμα, η συνάρτηση της CB::sendMePriceQuote (δηλ, "request for quotation") είναι διαφορετική από τη συνάρτηση CD::testDrive (δηλ."reservation"). Σημασιολογικά θα ήταν λάθος να αντιστοιχιστούν αυτές οι λειτουργίες καθώς παρέχουν διαφορετικές επιχειρησιακές συναρτήσεις. Ανάλογα, η IN::applyForInsurance δεν είναι σημασιολογικά συμβατή με την CB::calculatePayment καθώς αυτές οι λειτουργίες έχουν διαφορετικά πεδία ενδιαφέροντος. Για να οριστεί η συμβατότητα μεταξύ κατηγοριών λειτουργιών, θεωρούνται οι δύο λειτουργίες opik = (Dik, M ik, In ik, Out ik, P ik, C ik, Qik) και opjl=(Djl, Mjl,Injl, Outjl, Pjl, Cjl, Qjl).

Η Cik είναι συμβατή με την Cjl εάν:

1. (C_{ik}.Domain = C_{jl}.Domain) ή (C_{ik}.Domain $\in$ C_{jl}.Synonyms) ή (C_{jl}.Domain $\in$ C_{ik}.Synonyms), ή (C_{ik}.Synonyms $\cap$ C_{jl}.Synonyms $\neq \emptyset$), και
2. C_{ik}.Specialization $\subseteq$ C_{jl}.Specialization

Η πρώτη συνθήκη επαληθεύει ότι οι περιοχές ενδιαφέροντος είναι όμοιες ή συνώνυμες. Η δεύτερη συνθήκη βεβαιώνει ότι η op_{jl} παρέχει όλα τα χαρακτηριστικά της κατηγορίας της op_{ik}. Η συμβατότητα μεταξύ σκοπών ορίζεται με τον ίδιο τρόπο όπως μεταξύ κατηγοριών. Με βάση την έννοια της συμβατότητας μεταξύ κατηγοριών και σκοπών ορίζεται και ο κανόνας σημασιολογικής συμβατότητας λειτουργιών. Η op_{ik} είναι σημασιολογικά συμβατή λειτουργία με την op_{jl} εάν ο σκοπός της opik είναι συμβατός με το σκοπό της op_{jl} και η κατηγορία της op_{ik} είναι συμβατή με την κατηγορία της op_{jl}:

Ορισμός 10

Σημασιολογική συνθεσιμότητα λειτουργιών. Η λειτουργία op_{ik} = (D_{ik}, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}, Q_{ik}) είναι σημασιολογικά συνθέσιμη με την op_{jl}=(D_{jl}, M_{jl},In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl}) εάν (i) η P_{ik} είναι συμβατή με την P_{jl} και (ii) η C_{ik} είναι συμβατή με τη C_{jl}.

5.2.4. Ποιοτική Συνθεσιμότητα

Οι κανόνες ποιοτικής συνθεσιμότητας ελέγχουν τις ποιοτικές ιδιότητες των λειτουργιών που αλληλεπιδρούν. Έστω μια λειτουργία op_{ik} που καλεί μια άλλη λειτουργία op_{jl}. Η συνθετιμότητα κόστους επιβεβαιώνει ότι το ποσό σε δολάρια που διατίθεται να πληρώσει η op_{ik} είναι τουλάχιστον ίσο με το ποσό που απαιτεί η op_{jl}. Η συνθεσιμότητα ασφάλειας εξασφαλίζει ότι εάν η op_{ik} χρησιμοποιεί μηχανισμούς ασφαλείας για να ανταλλάξει μηνύματα, τότε η op_{jl} τα χρησιμοποιεί επίσης. Η συνθεσιμότητα μυστικότητας συγκρίνει τα χαρακτηριστικά μυστικότητας της op_{ik} και της op_{jl}. Οι προτιμήσεις μυστικότητας της op_{ik} θα έπρεπε να εντάσσονται στα χαρακτηριστικά μυστικότητας που εκθέτει η op_{jl}. Εάν ο πάροχος της op_{ik} δεν θέλει να αποκαλύψει μια παράμετρο p (δηλαδή, p $\in$ $Quality_{ik}$.privacy), τότε η p θα πρέπει επίσης να ανήκει στο $Quality_{jl}$.privacy. Ο επόμενος ορισμός συνοψίζει τους κανόνες ποιοτικής συνθεσιμότητας:

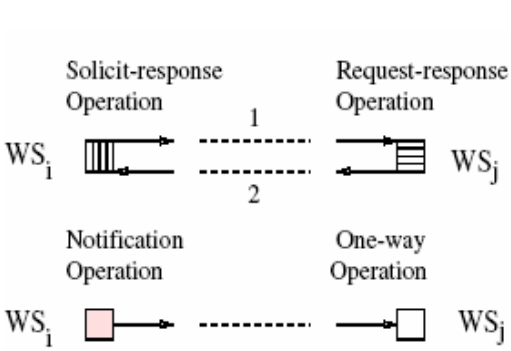
Ορισμός 11

Ποιοτική Συνθεσιμότητα. Η op_{ik} = (D_{ik}, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}, Q_{ik}) είναι ποιοτικά συνθέσιμη με την op_{jl}=(D_{jl}, M_{jl},In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl}) εάν:

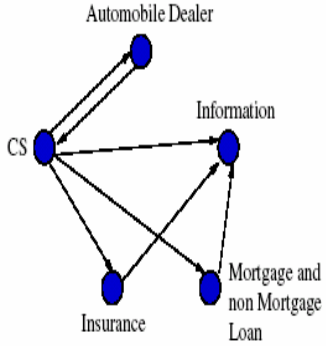
1. Q_{ik}.Fees $\geq$ Q_{jl}.Fees, και
2. (Q_{ik}.Security = true) $\Rightarrow$ (Q_{jl}.Security = true), και
3. Q_{ik}.Privacy $\subseteq$ Q_{jl}.Privacy.

5.2.5. Αρτιότητα σύνθεσης

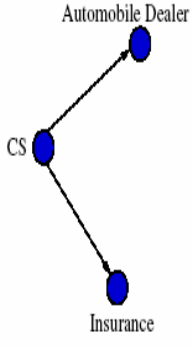
Μια άλλη πλευρά της σύνθεσης υπηρεσιών που πρέπει να ληφθεί υπόψη είναι εάν ο συνδυασμός ενός συνόλου υπηρεσιών με ένα συγκεκριμένο τρόπο παρέχει μια προστιθέμενη αξία. Για παράδειγμα ο συνδυασμός των υπηρεσιών airline και hotel θα παρείχε μια σύνθετη υπηρεσία travel preparation. Η ιδέα είναι ο ορισμός ενός κανόνα που καλείται αρτιότητα σύνθεσης, για να ελέγχει εάν οι σύνθετες υπηρεσίες είναι αξιόπιστες, ότι παρέχουν δηλαδή μια προστιθέμενη αξία. Εδώ εισάγεται η έννοια των προτύπων σύνθεσης (composition templates). Αυτοί οι γράφοι δημιουργούνται μέσω σχέσεων προτεραιότητας. Όπως φαίνεται στο Σχήμα 10α, η Υπηρεσία Διαδικτύου WSi προηγείται μιας άλλης υπηρεσίας WSj εάν η λειτουργία της WSi καλεί μια λειτουργία της WSj.Παρακάτω δίνεται ο ορισμός της σχέσης προτεραιότητας:



Σχήμα 9: Αρτιότητα Σύνθεσης



Σχήμα 9β



Σχήμα 9γ

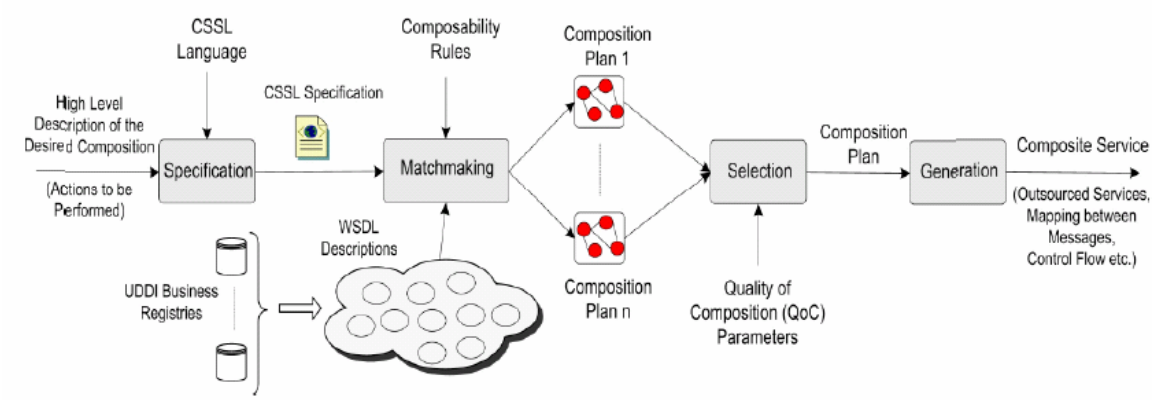
Ορισμός 12

Σχέση προτεραιότητας. Έστω $WS_i = (D_i, O_i, B_i, P_i i, C_i)$ και $WS_j = (D_j , O_j , B_j , P_j , C_j)$ δύο Υπηρεσίες Διαδικτύου. Η WS_i προηγείται *της* WS_j εάν $\exists\ op_{ik} \in Oi\ \exists\ op_{jl} \in Oj$ | (i) (M_{ik}= «ανακοινωση» και M_{jl}= «μονόδρομη»), ή (ii) (M_{ik}= «απαίτησης απάντησης» και M_{jl}= «αίτησης-απάντηση»).

Ένα πρότυπο σύνθεσης σχετίζεται με κάθε σύνθετη υπηρεσία και δίνει τη γενική δομή της. Έχει μοντελοποιηθεί από ένα κατευθυνόμενο γράφο (V, E) όπου το V είναι ένα σύνολο από ονόματα κατηγοριών υπηρεσίας και το E είναι ένα σύνολο ακμών. Μια ειδική κορυφή αντιστοιχεί στη σύνθετη υπηρεσία και έχει την ειδική τιμή "CS". Μια ακμή (vi, vj) ∈ E σημαίνει ότι μια υπηρεσία με όνομα κατηγορίας vi προηγείται μιας υπηρεσίας με όνομα κατηγορίας vj . Το σχήμα 9β δίνει το πρότυπο που αντιστοιχεί στη σύνθετη υπηρεσία car broker. Οι υπηρεσίες Lemon check, credit history, και driving history αναπαρίστανται από τον ίδιο κόμβο στο γράφο καθώς έχουν το ίδιο όνομα κατηγορίας (δηλ. "information").Τα πρότυπα σύνθεσης χρησιμοποιούνται για να συγκρίνουν τιμές που προστίθενται από διαφορετικές συνθέσεις. Για παράδειγμα, το πρότυπο του σχήματος 9γ είναι υπογράφος του προτύπου του σχήματος 9β. Αυτό σημαίνει ότι η δεύτερη σύνθετη υπηρεσία θα παρείχε ένα υποσύνολο των λειτουργιών που παρέχονται από την πρώτη.

Για να ελεγχθεί εάν μια σύνθεση υπηρεσιών είναι άρτια, ορίζεται η έννοια των αποθηκευμένων προτύπων. Τα αποθηκευμένα πρότυπα χωρίζονται σε δύο ομάδες. Η πρώτη ομάδα περιλαμβάνει πρότυπα που είναι προκαθορισμένα από ειδικούς των περιοχών ενδιαφέροντος. Για παράδειγμα, η βιομηχανία ταξιδίων θα συμφωνούσε ότι μια σύνθετη υπηρεσία travel preparation συνδυάζει airline, hotel, και car rental υπηρεσίες. Η δεύτερη ομάδα περιλαμβάνει πρότυπα που μαθαίνονται από το σύστημα. Κάθε φορά που ορίζεται μια σύνθετη υπηρεσία, το σύστημα αποθηκεύει το πρότυπο της στον κατάλογο. Για παράδειγμα , έστω ότι ένας συνθέτης ορίζει μια υπηρεσία της οποίας το πρότυπο απεικονίζεται στο σχήμα 9β. Εάν το πρότυπο δεν υπάρχει ήδη στον κατάλογο, το σύστημα θα αποθηκεύσει για μελλοντική χρήση.

Εφόσον όλα τα αποθηκευμένα πρότυπα παρέχουν έμφυτα προστιθέμενες αξίες, χρησιμοποιούνται για να ελέγξουν την αρτιότητα των σχεδίων σύνθεσης.



Σχήμα 10:Παρουσίαση της προτεινόμενης προσέγγισης για σύνθεση υπηρεσιών

Ορισμός 13

Αρτιότητα σύνθεσης. Μια σύνθεση υπηρεσιών είναι άρτια εάν το πρότυπό της είναι ένας υπογράφος ενός αποθηκευμένου προτύπου. Τα αποθηκευμένα πρότυπα χρησιμοποιούνται posteriori για να ελέγξουν την αρτιότητα των σύνθετων υπηρεσιών, όταν δηλαδή έχουν δημιουργηθεί. Σημειώνεται ότι οι κανόνες αρτιότητας σύνθεσης δεν χρησιμοποιούνται για να καθορίσουν εάν οι Υπηρεσίες Διαδικτύου είναι συνθέσιμες, αλλά για να καθορίσουν εάν η σύνθεση ενός συνόλου υπηρεσιών παρέχει μια προστιθέμενη αξία. Ακόμα και αν η σύνθεση δεν είναι άρτια, οι συνθέτες έχουν τη δυνατότητα να αποφασίσουν εάν θέλουν να θεωρήσουν μια τέτοια σύνθεση αποδεκτή.

5.3. Αυτόματη σύνθεση Υπηρεσιών Διαδικτύου

Με βάση το μοντέλο συνθεσιμότητας, προτείνεται μια προσέγγιση για την αυτόματη σύνθεση Υπηρεσιών Διαδικτύου. Η προσέγγιση αυτή αποτελείται από τέσσερις εννοιολογικά ξεχωριστές φάσεις: την *προδιαγραφή*, το *ταίριασμα*, την *επιλογή*, και την *παραγωγή*. Η φάση προδιαγραφής επιτρέπει υψηλού επιπέδου περιγραφές των επιθυμητών συνθέσεων χρησιμοποιώντας μια γλώσσα που καλείται CSSL (Composite Service Specification Language). Η φάση ταιριάσματος χρησιμοποιεί κανόνες συνθεσιμότητας για να παράγει σχέδια σύνθεσης που συμβαδίζουν με τις προδιαγραφές των συνθετών. Σχέδιο σύνθεσης είναι μια λίστα από component υπηρεσίες και αλληλεπιδράσεις μεταξύ τους για να δημιουργηθεί η σύνθετη υπηρεσία.

Ο αλγόριθμος ταιριάσματος χρησιμοποιεί ως είσοδο τις προδιαγραφές του συνθέτη και έναν κατάλογο προηγούμενων διεπαφών υπηρεσιών που περιγράφονται στο WSDL (επεκταμένη με σημασιολογικές δομές). Οι συνθέτες επιλέγουν ένα πλάνο που δημιουργείται (φάση επιλογής) με βάση τις παραμέτρους ποιότητας σύνθεσης. Χρησιμοποιώντας το επιλεγμένο σχέδιο, παράγεται αυτόματα μια λεπτομερής περιγραφή της σύνθετης υπηρεσίας (φάση παραγωγής). Αυτή η περιγραφή περιλαμβάνει τη λίστα από υπηρεσίες, τις αντιστοιχίες μεταξύ των σύνθετων και των oustourced λειτουργιών υπηρεσιών και μηνυμάτων, και τον έλεγχο ροής των outsourced λειτουργιών. Ο έλεγχος ροής αναφέρεται στην σειρά εκτέλεσης των λειτουργιών που outsourced από τη σύνθετη υπηρεσία. Η προδιαγραφή είναι η φάση που απαιτεί την παρεμβολή του συνθέτη για να περιγράψει την επιθυμητή σύνθεση. Η φάση ταιριάσματος παράγει αυτόματα σχέδια σύνθεσης με βάση τις προδιαγραφές του συνθέτη. Η φάση επιλογής χρησιμοποιεί τις παραμέτρους QoC για να επιλέξει το καλύτερο σχέδιο. Αυτές οι παράμετροι δίνονται από τους συνθέτες με βάση τα προφίλ τους που ορίζονται στη φάση προδιαγραφής. Η φάση δημιουργίας παρέχει αυτόματα μια περιγραφή της παραγόμενης σύνθετης υπηρεσίας σε μια γλώσσα «στόχο».

5.3.1. Φάση Προδιαγραφής

Για την προδιαγραφή των σύνθετων υπηρεσιών ορίζεται μια XML γλώσσα που καλείται CSSL (Composite ServiceSpecification Language),. Η CSSL είναι αρκετά εύκολη ώστε να επιτρέψει περιγραφές υψηλού επιπέδου σύνθετων υπηρεσιών. Οι συνθέτες πρέπει μόνο να έχουν μια γενική ιδέα σχετικά με την υπηρεσία που ενδιαφέρονται να παρέχουν. Δεν απαιτείται να γνωρίζουν τις τεχνικές λεπτομέρειες όπως τις περιγραφές των σύνθετων υπηρεσιών, τα χαρακτηριστικά τους, και πως ενώνονται μεταξύ τους. Υπάρχουν αρκετές διαφορές μεταξύ της CSSL και προηγούμενων γλωσσών σύνθεσης υπηρεσιών. Καταρχήν, η CSSL υιοθετεί ένα μοντέλο οντολογιών για να διευκολύνει τη σημασιολογία των Υπηρεσιών Διαδικτύου. Οι περισσότερες από τις υπάρχουσες γλώσσες δεν λαμβάνουν υπόψη τις σημασιολογικές δυνατότητες των Υπηρεσιών Διαδικτύου. Δεύτερον, η CSSL προδιαγραφή μιας σύνθετης υπηρεσίας δεν αναφέρεται σε καμία oustourced υπηρεσία. Αυτό είναι σε αντίθεση με άλλες γλώσσες όπου οι συνθέτες εισάγουν αναφορές σε σύνθετες υπηρεσίες στις προδιαγραφές σύνθετων υπηρεσιών τους. Τρίτον, οι CSSL προδιαγραφές χρησιμοποιούνται ως σημείο εισόδου για την (ημι-) αυτόματη παραγωγή σύνθετων υπηρεσιών. Τέλος, η CSSL ορίζει μια γλώσσα τύπου WSDL για σύνθετες υπηρεσίες. Επεκτείνει τη γλώσσα WSDL για να επιτρέψει: (1) την περιγραφή σημασιολογικών χαρακτηριστικών των Υπηρεσιών Διαδικτύου και (2) την προδιαγραφή του ελέγχου ροής μεταξύ λειτουργιών σύνθετων υπηρεσιών. Αυτό κάνει τον ορισμό σύνθετων υπηρεσιών τόσο απλό όσο και ο ορισμός απλών υπηρεσιών. Επιπλέον, επιτρέπει την υποστήριξη επαναληπτικής σύνθεσης υπηρεσιών. Οι σύνθετες υπηρεσίες μπορεί να θεωρηθούν ως WSDL υπηρεσίες και επομένως να χρησιμοποιηθούν ως components για νέες συνθέσεις.

Τα κύρια χαρακτηριστικά της CSSL παρουσιάζονται μέσο του παραδείγματος στον πίνακα 11. Για χάρη της σαφήνειας, παραλείπονται οι αναφορές σε XML namespaces. Το υψηλότερο στοιχείο της CSSL προδιαγραφής είναι το *service* που περιλαμβάνει το όνομα της σύνθετης υπηρεσίας. Οι ιδιότητες κατηγορίας υπηρεσίας ορίζονται από το στοιχείο *category*. Κάθε στοιχείο *operation* επιτρέπει την προδιαγραφή του ονόματος λειτουργίας, την περιγραφή, τον τρόπο λειτουργίας, την κατηγορία, και την ποιότητα. Επίσης περιέχει στοιχεία εισόδου και/ή εξόδου. Για παράδειγμα, η λειτουργία receiveSpecialOffers είναι μονόδρομη και παρέχει πληροφορίες τιμής και πώλησης για την βιομηχανία αυτοκινήτων. Το μήνυμα εισόδου της λειτουργίας, που ονομάζεται offer περιέχει πέντε παραμέτρους. Κάθε παράμετρος έχει ένα τύπο δεδομένων XML, μονάδα και ρόλο. Για τη διευκόλυνση των μηνυμάτων εισόδου και εξόδου, παρέχεται ένα σύνολο προκαθορισμένων μηνυμάτων που χρησιμοποιούνται από τους συνθέτες ως βάση για τον καθορισμό λειτουργιών σύνθετων υπηρεσιών. Οι συνθέτες μπορούν να ορίζουν νέα μηνύματα, να χρησιμοποιήσουν προκαθορισμένα μηνύματα, είτε να τροποποιήσουν προκαθορισμένα μηνύματα. Για παράδειγμα, μπορεί να αποφασίσουν να αφαιρέσουν το κομμάτι TOD (terms of delivery or transport) από ένα προκαθορισμένο μήνυμα “request for quotation” εάν δεν ενδιαφέρονται να παρέχουν τέτοιες πληροφορίες. Η CSSL ενεργοποιεί επίσης την προδιαγραφή του ελέγχου ροής των λειτουργιών σύνθετων υπηρεσιών. (flow ιδιότητα). Οι λειτουργίες μπορεί να εκτελεστούν ακολουθιακά είτε παράλληλα. Για παράδειγμα, η προδιαγραφή της υπηρεσίας car broker δείχνει ότι η υπηρεσία ελέγχει πρώτα την ιστορία πληρωμής των πελατών της (source attribute) πριν ζητήσει επιδότηση εκ μέρους τους (target attribute).Σημειώνεται επίσης ότι η CSSL επιτρέπει την προδιαγραφή των συνθηκών των ελέγχων ροής.

```
<service name="car broker"/>
 <category domain="brokerage">
 ".....
 <binding name="SOAP"/>
 <message name="offer">
 <parameter name="price" type="float" unit="US dollar" role="extendedPrice"/>
 <parameter name="make" type="string" ...../>
 <parameter name="model" type="string" ...../>
 <pararmeter name="year" type="gYear" ...../>
 <parameter name="mileage" type="integer" ...../>
 </message>
 ".....
 <operation name="receiveSpecialOffers" mode="one way"/>
 <input name="offer"/>
 <category domain="automobile dealer">
 <synonyms>
 <synonym value="car dealer"/>
 </synonyms>
 ".....
 <purpose function="price-sales catalogue"/>
 ".....
 <quality>
 <fees value=0/>
 ".....
 </operation>
 <flow source="getPayingHistory" target="applyForFinancing">
```

Πίνακας 11: Η CSSL προδιαγραφή για την σύνθετη υπηρεσία car broker

5.3.2. Φάση ταιριάσματος

Όταν είναι διαθέσιμες οι CSSL προδιαγραφές, το επόμενο βήμα είναι η παραγωγή των αντίστοιχων σχεδίων σύνθεσης χρησιμοποιώντας έναν αλγόριθμο ταιριάσματος. (Πίνακας 12). Καθώς ο αριθμός των παραγόμενων σχεδίων μπορεί να είναι μεγάλος, οι συνθέτες έχουν τη δυνατότητα να ελέγξουν τον αριθμό των παραγόμενων σχεδίων μέσο της εισόδου *nb requested plans*. Ο αλγόριθμος χρησιμοποιεί διεπαφές υπηρεσιών αντί για ολόκληρες περιγραφές σύνθετων υπηρεσιών για τον έλεγχο συμβατότητας. Αυτό έχει το σημαντικό πλεονέκτημα της μείωσης του αριθμού των υπηρεσιών που θα προσπελαστούν. Πράγματι, η ίδια διεπαφή μπορεί να χρησιμοποιείται σε αρκετές υπάρχουσες υπηρεσίες με διαφορετικές υλοποιήσεις. Για παράδειγμα, η βιομηχανία αυτοκινήτου μπορεί να ορίσει μια διεπαφή για την πώληση αυτοκινήτων. Οι υπηρεσίες Car dealer θα επαναχρησιμοποιούσαν τότε αυτή τη διεπαφή για να δημιουργήσουν τις δικές τους υπηρεσίες. Η γενική αρχή του αλγορίθμου ταιριάσματος είναι να αντιστοιχίσει κάθε λειτουργία op_{ik} = (D_{ik}, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}, Q_{ik}) της σύνθετης υπηρεσίας WSi σε μια ή περισσότερες λειτουργίες op_{jl} = (D_{jl}, M_{jl}, In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl}) της υπάρχουσας υπηρεσίας WSj. Ο αλγόριθμος ψάχνει για Υπηρεσίες Διαδικτύου WSj = (Dj, Oj, Bj, Pj, Cj) έτσι ώστε η P_{ik} και η C_{ik} να είναι συμβατές με ένα τουλάχιστον στοιχείο της Pj και της Cj , αντίστοιχα. Τότε ο αλγόριθμος επιβεβαιώνει ότι οι υπηρεσίες που αλληλεπιδρούν έχουν συμβατότητα σύνδεσης.

Οι Υπηρεσίες Διαδικτύου οργανώνονται σε κοινότητες οι οποίες παρέχουν τα μέσα για μια οντολογική οργάνωση του χώρου διαθέσιμων υπηρεσιών. Κάθε κοινότητα συσταδοποιεί Υπηρεσίες Διαδικτύου με βάση την κατηγορία τους. Όλες οι υπηρεσίες που έχουν παρόμοιες κατηγορίες ανήκουν στη ίδια κοινότητα κατηγορίας. Μια υπηρεσία μπορεί να ανήκει σε διαφορετικές κοινότητες. Η χρήση των κοινοτήτων υπηρεσιών επιταχύνει τη διαδικασία ανακάλυψης σχετικών σύνθετων υπηρεσιών. Έστω ότι η κατηγορία C_{ik} δεν είναι συμβατή με την κατηγορία της WSj. Σε αυτήν την περίπτωση η C_{ik} δεν είναι συμβατή με υπηρεσίες που ανήκουν

στην κοινότητα της WSj. Επομένως αυτές οι Υπηρεσίες Διαδικτύου θα περικοπούν από το χώρο υπηρεσίας. Για κάθε ζεύγος λειτουργιών (op_{ik}, op_{jl}), ελέγχει επίσης τη συμβατότητα τρόπου λειτουργίας και τη συμβατότητα μηνυμάτων. Κάθε επανάληψη της δήλωσης *while* παράγει ένα σχέδιο σύνθεσης. Το ταιριασμένο σύνολο περιέχει όλες τις component λειτουργίες που έχουν ήδη συνδεθεί σε μια λειτουργία σύνθετης υπηρεσίας. Η χρήση αυτού του συνόλου αποτρέπει τη δημιουργία σχεδίων σύνθεσης που μπορεί να αναφέρονται από προηγούμενα δημιουργημένα σχέδια. Για παράδειγμα, έστω ότι τα σχέδια plan1 = {(op_{i1},op_{j1}), (op_{i2},op_{j2})} και plan2 = {(op_{i1},op_{j3}),(op_{i2},op_{j4})}, έχουν ήδη παραχθεί. Εφόσον τα σχέδια plan3 = {(opi_1,op_{j1}), (op_{i2},op_{j4})} και plan4 = {(opi_1,opj_3), (op_{i2},op_{j2})} μπορούν να αναφερθούν από τα plan1 και plan2, δεν υπάρχει λόγος να παραχθούν ξανά.

Ο αλγόριθμος ταιριάσματος χρησιμοποιεί τις ακόλουθες συναρτήσεις για να ελέγξει την συνθεσιμότητα: purpose_compatible(), category_compatible(), quality_composable(), message_composable(), και sound(). Οι συναρτήσεις purpose_compatible() και category_compatible() επιστρέφουν σωστό ή λάθος ανάλογα με το εάν η λειτουργία σύνθετης υπηρεσίας έχει ένα συμβατό σκοπό ή κατηγορία με το σκοπό ή κατηγορία της λειτουργίας component υπηρεσίας. Η quality_composable() επιστρέφει σωστό εάν η λειτουργία σύνθετης υπηρεσίας είναι ποιοτικά συνθέσιμη με μια λειτουργία component υπηρεσίας. Οι δύο άλλες συναρτήσεις δίνονται στον πίνακα 13. Η συνάρτηση message_compatible() επιστρέφει σωστό ή λάθος ανάλογα με το εάν ένα μήνυμα Mi είναι συμβατό με το Mj. Για να επιτραπεί το ένα προς ένα ταίριασμα μεταξύ των παραμέτρων των Mi και Mj, χρησιμοποιείται το σύνολο matched. Αυτό το σύνολο περέχει τις παραμέτρους του Mj που έχουν ήδη αντιστοιχιστεί στις παραμέτρους του Mi. Η συνάρτηση sound() ελέγχει την αρτιότητα του δημιουργημένου σχεδίου. Μόλις υπολογιστεί ένα πρότυπο για το δημιουργημένο σχέδιο, συγκρίνεται με τα αποθηκευμένα πρότυπα.

```
(01) Input : WSi, repository, nb_requested_plans {
(02)  nb_generated_plans = 0
(03)  matched = ∅
(04)  while nb_generated_plans ≤ nb_requested_plans do
(05)   { plan = ∅
(06)    for each operation opik ∈ Oi do
(07)     { found = false
(08)      for each service WSj from repository | category_compatible(Cik,Cj)
(09)                                  and purpose_compatible(Pik,Pj) and (Bi ∩ Bj ≠ ∅) do
(10)      { for each operation opjl ∈ Oj | (modeik and modejl are dual) and (opjl ∉ matched)
(11)          if purpose_compatible(Pik,Pjl) and category_compatible(Cik,Cjl) and quality_composable(opik,opjl)
(12)              and message_composable(inik,outjl) and message_composable(injl,outik)
(13)           then { found = true
(14)                  plan = plan ∪ {(opik,opjl)}
(15)                  matched = matched ∪ {opjl}
(16)                  break }
(17)       if found then break
(18)      } /* for in line (08) */
(19)      if ¬found
(20)         then { output("no matchmaking for",opik)
(21)                break }
(22)     } /* for in line (06) */
(23)    if ¬found then break
(24)              else if sound(plan)
(25)                   then output(plan,ST) /* ST is a Stored Template */
(26)                   else if relevant(plan,τrelevance) and complete(plan,τcompletness)
(27)                      then output(plan,ST,τrelevance,τcompletness) /* Test for QoC parameters */
(28)                      else output(plan,"not sound",τrelevance,τcompletness)
(29)   nb_generated_plans = nb_generated_plans + 1
(30)  } /* while in line (04) */ }
```

Πίνακας 12: Ο αλγόριθμος ταιριάσματος

```
(01) function message_composable(Mi,Mj):boolean {
(02)   matched = ∅
(03)   for each param pik ∈ Pi do
(04)    { found = false
(05)     for each param pjl ∈ Pj | pjl ∉ matched do
(06)      if (T(pik) = T(pjl) or
(07)         T(pjl) is derived from T(pik)) and
(08)         (U(pik) = U(pjl)) and (R(pik) = R(pjl))
(09)         then { found = true
(10)                matched = matched ∪ {pjl}
(11)                break }
(12)     if ¬found then return false
(13)    } /* for in line (03) */
(14)  return true
(15) }
(16)
(17)
```

```
function sound(plan):boolean {
  for each element (opik,WSj,opjl) ∈ plan do
   if modeik ∈ {"notification","solicit-response"}
    then template = template ∪ ("CS",Cj)
    else template = template ∪ (Cj,"CS")
  for each stored template ST do
   for each pair (vp,vq) ∈ template do
    { found = false
      for each pair (vr,vs) ∈ ST do
       if (vp,vq) = (vr,vs)
          then { found = true
                 break }
     if ¬ found then break
    } /* for in line (07) */
  if found then return true
          else return false
}
```

Πίνακας 13: Συνθεσιμότητα μηνυμάτων και συναρτήσεις ελέγχου αρτιότητας

5.3.3. Φάση επιλογής

Στο τέλος της φάσης ταιριάσματος έχουν παραχθεί αρκετά σχέδια σύνθεσης. Για τη διευκόλυνση της επιλογής των σχετικών σχεδίων, ορίζονται τρεις παράμετροι ποιότητας σύνθεσης (quality of composition - QoC): **κατάταξη**, **σχετικότητα**, και **πληρότητα**. Μπορούν να οριστούν και άλλες QoC παράμετροι με βάση το κόστος. Παρακάτω παρουσιάζονται ορισμοί για την κατάταξη, τη σχετικότητα και την πληρότητα :

Κατάταξη Σύνθεσης:
Η κατάταξη μιας σύνθεσης δίνει μια προσέγγιση της σημασίας της. Για κάθε σχέδιο ορίζεται ένα πρότυπο σύνθεσης CT. Έστω ότι το CT είναι ένας υπογράφος του αποθηκευμένου προτύπου STi. Χρησιμοποιείται η συνάρτηση R(Reference) που ορίζεται σε ένα σύνολο από αποθηκευμένα πρότυπα. Το R(STi) δίνει τον αριθμό των φορών που έχουν δημιουργηθεί υπηρεσίες με πρότυπα που είναι υπογράφοι του STi. Η κατάταξη του CT όσον αφορά το STi είναι η αναλογία των αναφορών στο ST. Ορίζεται ως εξής (n είναι ο αριθμός των αποθηκευμένων προτύπων):

$$Ranking(CT, ST_i) = \frac{\mathcal{R}(ST_i)}{\sum_{k=1}^{n} \mathcal{R}(ST_k)}$$

***Σχετικότητα Σύνθεσης*:**
Αυτή η παράμετρος, που δηλώνεται ως CR, δίνει μια προσέγγιση της αρτιότητας σύνθεσης. Συγκρίνει τις ακμές ενός προτύπου σύνθεσης CT, με τις ακμές ενός αποθηκευμένου προτύπου STi. CR(CT,STi) είναι ο λόγος των ακμών του CT που υπάρχουν στο STi. Ορίζεται ως εξής (E και Ei είναι οι ακμές των CT και STi αντίστοιχα):

$$CR(CT, ST_i) = \frac{|\, E \cap E_i \,|}{|\, E \,|}$$

Πληρότητα Σύνθεσης: Αυτή η παράμετρος, που δηλώνεται ως CC, δίνει μια αναλογία των λειτουργιών σύνθετων υπηρεσιών που είναι συνθέσιμες με λειτουργίες component υπηρεσιών. Το CC επιτρέπει τη δημιουργία σχεδίων των οποίων οι σύνθετες υπηρεσίες μπορεί να μην είναι πλήρως συνθέσιμες με component υπηρεσίες. Η CC παίρνει τιμή από συνθέτες υπηρεσίες και εξαρτάται από το επίπεδο ειδίκευσής τους. Πράγματι, η τιμή της CC είναι σχετικά χαμηλή, καθώς ο αλγόριθμος μπορεί να επιστρέψει σχέδια στα οποία το 75% των λειτουργιών σύνθετων υπηρεσιών δεν είναι συνθέσιμες με λειτουργίες component υπηρεσιών. Σε αυτήν την περίπτωση, οι συνθέτες μπορεί να χρειαστεί να αλλάξουν τις προδιαγραφές τους έτσι ώστε η επιθυμητή

υπηρεσία να μπορεί να χειριστεί χαρακτηριστικά άλλων υπηρεσιών. Ο ακόλουθος τύπος ορίζει την παράμετρο CC για μια σύνθετη υπηρεσία WSi:

$$CC(WS_i) = \frac{|\, Composable(O_i) \,|}{|\, O_i \,|}$$, όπου

Composable $(O_{i)} = \{op_{ik} \in O_i \mid \exists\ WS_J\ \exists\ O_j$ έτσι ώστε η op_{ik} να είναι συντακτικά και σημασιολογικά συνθέσιμη με την $op_{jl}\}$.

Flow Model	Global Model
<serviceProvider name="myCarDealer"> <locator type="static" service="carDealer.com"> </serviceProvider> ..... <activity name="Activity_1"> <performedBy serviceProvider="myCarDealer"> <implement> <export> <target portType="Port_1" operation="receiveSpecialOffers"/> </export> </implement> </activity> <activity name="Activity_2"> ..... <target portType="Port_1" operation="askForPayingHistory"/> ..... <activity name="Activity_3"> ..... <target portType="Port_1" operation="applyForFinancing"/> ..... <controlLink source="Activity_2" target="Activity_3">	<plugLink> <source serviceProvider="carBroker" portType="Port_1" operation="receiveSpecialOffers"/> <target serviceProvider="carDealer" portType="portCarDealer" operation="specialOffers"> </plugLink> <plugLink> <source serviceProvider="carBroker" portType="Port_1" operation="insuranceQuote"> <target serviceProvider="insurance" portType="portInsurance" operation="applyForInsurance"> </plugLink> <plugLink> <source serviceProvider="carBroker" portType="Port_1" operation="applyForFinancing"> <target serviceProvider="financing" portType="portFinancing" operation="financingQuote"> </plugLink>

Πίνακας 14: WSFL περιγραφές που παρήχθησαν από την υπηρεσία car broker

Τα σχέδια σύνθεσης ταξινομούνται και επιστρέφονται με βάση την κατάταξή τους. Τα σχέδια με υψηλότερη κατάταξη επιστρέφονται πρώτα. Αυτό υποθέτει ότι διατηρείται ένα συντελεστής κατάταξης για κάθε αποθηκευμένο πρότυπο. Οι συνθέτες ορίζουν κατώφλια σχετικότητας και πληρότητας που αντιστοιχούν στις παραμέτρους σχετικότητας και πληρότητας. Τα σχέδια επιστρέφονται στους συνθέτες εάν έχουν σχετικότητα και πληρότητα μεγαλύτερη από τα αντίστοιχα κατώφλια. Οι παράμετροι QoC μπορούν να καθοριστούν μέσα στις CSSL προδιαγραφές έτσι ώστε τα καλύτερα σχέδια να επιλέγονται αυτόματα και να επιστρέφονται στους χρήστες.

5.3.4. Φάση Παραγωγής

Η τελευταία φάση στην παρούσα προσέγγιση στοχεύει στην παραγωγή λεπτομερούς περιγραφής μιας σύνθετης υπηρεσίας. Η περιγραφή περιλαμβάνει τη λίστα των outsourced υπηρεσιών, τις αντιστοιχίες μεταξύ λειτουργιών σύνθετων και component υπηρεσιών, τις αντιστοιχίες μεταξύ μηνυμάτων και παραμέτρων, και τον έλεγχο ροής και δεδομένων μεταξύ σύνθετων υπηρεσιών. Δύο σημαντικά χαρακτηριστικά αυτής της φάσης είναι η εξατομίκευση (customization) και η επεκτασιμότητα (extensibility). Η εξατομίκευση αναφέρεται στη δυνατότητα δημιουργίας περιγραφών σύνθετων υπηρεσιών σε διαφορετικές γλώσσες όπως η WSFL (Web Services Flow Language), η XLANG, και η BPEL4WS (Business Process Execution Language for Web Services). Οι συνθέτες ορίζουν τη γλώσσα στόχο στην CSSL

προδιαγραφή τους. Η επεκτασιμότητα αναφέρεται στην προοπτική να συμπεριληφθούν επιπλέον γλώσσες σύνθεσης. Πράγματι, η δομή των σχεδίων σύνθεσης είναι αρκετά αφηρημένη ώστε να χρησιμοποιείται σε ένα ενδιάμεσο επίπεδο μεταξύ των CSSL προδιαγραφών και των περισσότερων υπαρχουσών γλωσσών σύνθεσης Υπηρεσιών Διαδικτύου.

6. Ο σχεδιασμός αλγορίθμων μεσολαβητών Ποιότητας Υπηρεσίας για Υπηρεσίες Διαδικτύου με QoS ιδιότητες

Οι περισσότερες υλοποιήσεις Υπηρεσιών Διαδικτύου δεν εγγυώνται τα επίπεδα ποιότητας υπηρεσίας που παρέχονται στους χρήστες τους. Προς το παρόν, το UDDI είναι απλά ένας κατάλογος που επιτρέπει στους πελάτες να αναζητούν Υπηρεσίες Διαδικτύου με βάση τη λειτουργικότητα τους, αλλά χωρίς πληροφορίες Ποιότητας Υπηρεσίας (QoS). Στην συνέχεια καθορίζεται μια γενική αρχιτεκτονική Υπηρεσιών Διαδικτύου με ιδιότητες ποιότητας υπηρεσίας, η QCWS, ορίζοντας ένα υποσύστημα διαπραγμάτευσης Ποιότητας Υπηρεσίας μεταξύ πελατών (clients) και παροχέων (providers) Υπηρεσιών Διαδικτύου [15]. Στο QCWS, ο μεσολαβητής (broker) Ποιότητας Υπηρεσίας συλλέγει τις πληροφορίες QoS σχετικά με τους παρόχους υπηρεσιών, λαμβάνει αποφάσεις επιλογής υπηρεσιών για τους πελάτες και έπειτα διαπραγματεύεται με κάποιους από τους παρόχους ώστε να ικανοποιήσει τις απαιτήσεις Ποιότητας Υπηρεσίας. Από πλευράς υλοποίησης, ένας μεσολαβητής QoS μπορεί να είναι μέρος του πελάτη, μέρος του εξυπηρετητή, είτε ανεξάρτητη Υπηρεσία Διαδικτύου. Οι λειτουργίες ενός μεσολαβητή μπορεί να διαφέρουν ανάλογα με τις ρυθμίσεις του.

Η μελέτη που παρουσιάζεται στη συνέχεια εστιάζει σε συστήματα όπου ο μεσολαβητής QoS λειτουργεί ως το front-end ενός εξυπηρετητή (server). Η βασική λειτουργία είναι να βοηθάει τους εξυπηρετητές να αποφασίσουν πόσους πόρους θα πρέπει να αναθέσουν σε κάθε πελάτη για να ικανοποιήσει τις απαιτήσεις QoS τους, και αφού τους αναθέσει ελαχιστοποιεί την διακύμανση της QoS για κάθε πελάτη. Μελετούνται δυο αλγορίθμοι κατανομής πόρων που χρησιμοποιούνται από τον μεσολαβητή. Ο πρώτος αλγόριθμος, ο HQ, χρησιμοποιείται από εξυπηρετητές που δε παρέχουν διαφορετικά επίπεδα ποιότητας υπηρεσίας στους πελάτες (Οι περισσότεροι από τους σημερινούς εξυπηρετητές είναι αυτής της κατηγορίας.). Κάθε πελάτης λαμβάνει το ίδιο ποσό πόρων του συστήματος. Ο αλγόριθμος προσαρμόζει τους πόρους που παρέχονται ανάλογα με τον αριθμό των ενεργών πελατών στο σύστημα. Ο δεύτερος αλγόριθμος, ο RQ, έχει σχεδιαστεί για QoS-capable εξυπηρετητές που μπορούν να παρέχουν διαφορετικά επίπεδα ποιότητας σε διαφορετικούς πελάτες. Ο αλγόριθμος κρατάει ένα συγκεκριμένο ποσό πόρων για έναν εικονικό πελάτη έτσι ώστε μελλοντικοί πελάτες να λάβουν ένα ικανοποιητικό επίπεδο υπηρεσίας χρησιμοποιώντας τους δεσμευμένους πόρους. Ο σκοπός αυτής της έρευνας είναι να παρέχει υποστήριξη QoS σε Υπηρεσίες Διαδικτύου έτσι ώστε ένας πελάτης να μπορεί να λάβει συνεχή επίπεδα υπηρεσίας ανεξάρτητα από άλλες αιτήσεις στον ίδιο εξυπηρετητή. Οι αλγόριθμοι των μεσολαβητών που παρουσιάζονται εδώ μπορούν να χρησιμοποιηθούν τόσο για παραδοσιακούς (legacy) όσο και για εξυπηρετητές με ιδιότητες ποιότητας υπηρεσίας. Χωρίς τέτοιους μηχανισμούς κατανομής πόρων και ελέγχου αποδοχής, οι χρήστες μπορεί να αντιμετωπίσουν αστάθεια της ποιότητα υπηρεσίας ανάλογα με το δυναμικό φόρτο του εξυπηρετητή.

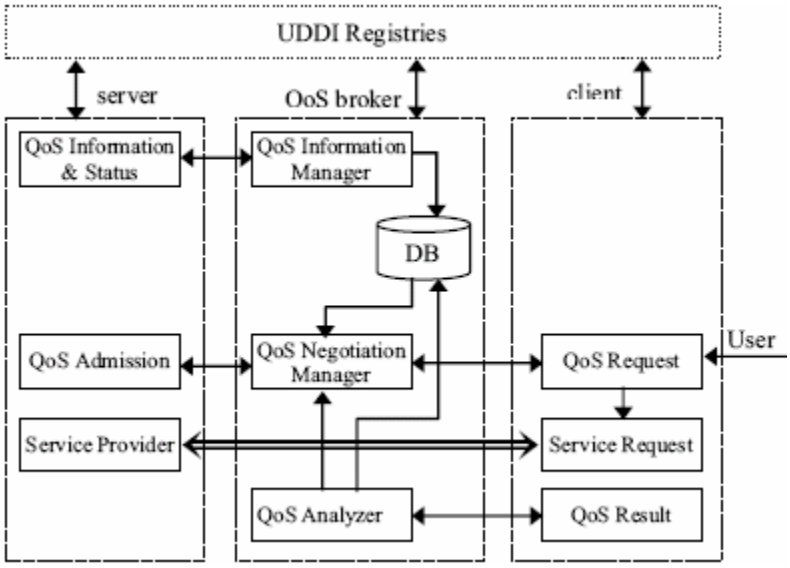
6.1. Η αρχιτεκτονική QCWS

Στη συνέχεια παρουσιάζεται μια αρχιτεκτονική Υπηρεσιών Διαδικτύουμε ιδιότητες ποιότητας υπηρεσίας, η QCWS. Έχει τρεις οντότητες: τον εξυπηρετητή, τον μεσολαβητή QoS και τον πελάτη. Οι οντότητες αυτές συνεργάζονται για να παρέχουν την επιθυμητή ποιότητα υπηρεσίας που ζητάει ο χρήστης.

6.1.1. Εξυπηρετητής

Σήμερα υπάρχουν δυο είδη εξυπηρετητών που παρέχουν Υπηρεσίες Διαδικτύου. Οι υπηρεσίες από το πρώτο είδος δεν παρέχουν QoS υποστήριξη και αναφέρονται εδώ ως legacy εξυπηρετητές. Η έννοια των επιπέδων ποιότητας υπηρεσιών δεν υπάρχει στους legacy εξυπηρετητές καθώς όλοι οι πελάτες μεταχειρίζονται όμοια και δρομολογούνται χρησιμοποιώντας δρομολογητές του λειτουργικού συστήματος. Οι περισσότεροι σημερινοί πάροχοι Υπηρεσιών Διαδικτύου ανήκουν σ’ αυτή την κλάση. Γι’ αυτούς τους legacy

εξυπηρετητές, η μόνη πληροφορία QoS που μπορεί να πάρει ο μεσολαβητής είναι ο φόρτος του εξυπηρετητή. Το άλλο είδος εξυπηρετητών παρέχει QoS υποστήριξη και καλούνται QoS εξυπηρετητές. Έχουν την ιδιότητα να αναθέτουν διαφορετικά ποσά πόρων του συστήματος σύμφωνα με τις QoS απαιτήσεις τους. Οι QoS πληροφορίες που παρέχονται από τους QoS εξυπηρετητές περιλαμβάνουν τα επίπεδα υπηρεσίας με τα αντίστοιχα κόστη, τη μέγιστη χωρητικότητα υπηρεσίας και την τρέχουσα διαθέσιμη χωρητικότητα σε κάθε επίπεδο υπηρεσίας. Ένα άλλο σημαντικό είδος συναρτήσεων του εξυπηρετητή κάνει έλεγχο αποδοχής. Αφού ένας μεσολαβητής επιλέξει μια υπηρεσία (και επίπεδο υπηρεσίας αν είναι διαθέσιμο) για ένα πελάτη, στέλνει μια αίτηση για επιβεβαίωση στο εξυπηρετητή (admission). Ο εξυπηρετητής δέχεται ένα αίτημα όταν μπορεί να κρατήσει ένα ικανό ποσό πόρων για να πετύχει την επιθυμητή ποιότητα.



Σχήμα 11: Αρχιτεκτονική του συστήματος

6.1.2. Ο μεσολαβητής QoS

Ένας μεσολαβητής QoS λαμβάνει τα αιτήματα QoS υπηρεσίας από τους πελάτες και αναγνωρίζει κατάλληλες υπηρεσίες για αυτούς. Τα βασικά υποσυστήματα του περιλαμβάνουν έναν διαχειριστή πληροφοριών QoS, έναν διαχειριστή διαπραγμάτευσης QoS και έναν αναλυτή QoS.

6.1.3. Ο διαχειριστής πληροφοριών QoS

Ο διαχειριστής πληροφοριών QoS συλλέγει πληροφορίες QoS του εξυπηρετητή που απαιτούνται για την διαπραγμάτευση και ανάλυση της QoS. Ελέγχει τους UDDI κατάλόγους για να πάρει πληροφορίες για τους εξυπηρετητές και επικοινωνεί με τους εξυπηρετητές για QoS πληροφορίες. Αυτό γίνεται περιοδικά ώστε να διατηρεί τις πληροφορίες του μεσολαβητή συνεπείς και ενημερωμένες με τους UDDI καταλόγους. Οι πληροφορίες που συλλέγονται αποθηκεύονται στη βάση δεδομένων του μεσολαβητή. Περιέχει επίσης QoS στατιστικές πληροφορίες που δημιουργούνται από τον αναλυτή QoS.

6.1.4. Διαχειριστή Διαπραγμάτευσης QoS

Ο διαχειριστής διαπραγμάτευσης QoS είναι ο πυρήνας του μεσολαβητή QoS. Εκτελεί επιλογές υπηρεσιών και αποκατάσταση υπηρεσιών για τους πελάτες. Αφού λάβει την αίτηση υπηρεσίας του πελάτη και τις απαιτήσεις QoS, ένας διαχειριστής διαπραγμάτευσης QoS ψάχνει τη βάση δεδομένων του μεσολαβητή για κατάλληλες υπηρεσίες. Εάν βρεθούν περισσότερες από μια υποψήφιες υπηρεσίες, χρησιμοποιείται ένας αλγόριθμος λήψης αποφάσεων για να επιλέξει

την πιο κατάλληλη. Οι QoS πληροφορίες τόσο από τον εξυπηρετητή όσο και από τον αναλυτή QoS θα χρησιμοποιηθούν για να ληφθεί απόφαση. Μόλις επιλεγεί ο υποψήφιος, ο διαχειριστής διαπραγμάτευσης QoS διαπραγματεύεται με τον εξυπηρετητή για να ζητήσει μια δέσμευση QoS. Αυτό γίνεται με τη συνεργασία του ελεγκτή αποδοχής QoS στην πλευρά του εξυπηρετητή. Εάν η διαπραγμάτευση δεν είναι επιτυχημένη, ο μεσολαβητής πρέπει να αναγνωρίσει έναν άλλο υποψήφιο εξυπηρετητή και να επαναλάβει όλη τη διαδικασία διαπραγμάτευσης.

6.1.5. Αναλυτής QoS

Αφού τελειώσει ένα έργο Υπηρεσίας Διαδικτύου, ένας πελάτης μπορεί να αθροίσει την εμπειρία QoS και να στείλει τις πληροφορίες στον αναλυτή QoS του μεσολαβητή QoS. Ο αναλυτής QoS ελέγχει τα ανεπεξέργαστα δεδομένα για να παράγει στατιστικές πληροφορίες σχετικά με μια υπηρεσία και τα βάζει στη βάση δεδομένων του μεσολαβητή ως ιστορικές πληροφορίες QoS. Ο διαχειριστή διαπραγμάτευσης QoS χρησιμοποιεί αυτές τις πληροφορίες κατά την επιλογή υπηρεσίας και τη φάση λήψης απόφασης.

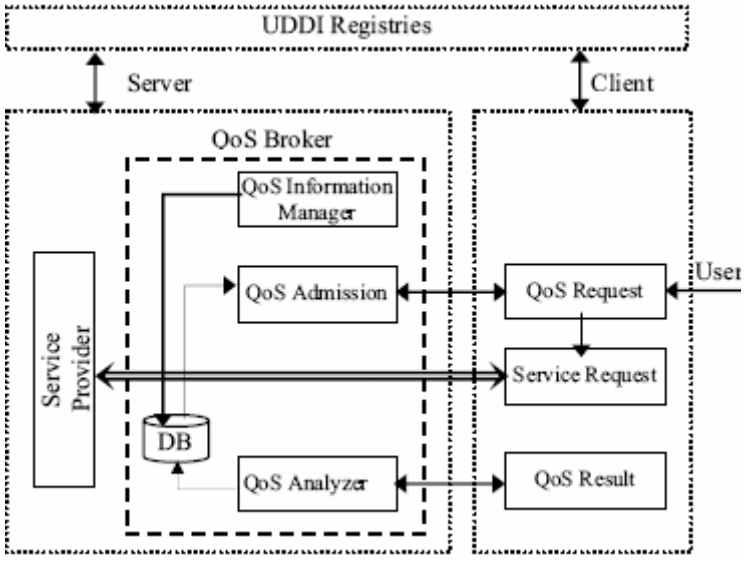
6.1.6. Πελάτης

Ως τελικοί χρήστες των Υπηρεσιών Διαδικτύου, οι πελάτες στέλνουν τις αιτήσεις υπηρεσιών και τις απαιτήσεις QoS σε ένα μεσολαβητή και τον αφήνουν να επιλέξει την πιο κατάλληλη υπηρεσία γ'αυτούς. Οι πελάτες επίσης συλλέγουν τις πληροφορίες QoS μετά από κάθε κλήση υπηρεσίας και τα στέλνουν στον μεσολαβητή.

6.2. Αλγόριθμοι Allocation Πόρων Μεσολαβητή QoS

6.2.1. Η QCWS αρχιτεκτονική Εξυπηρετητή-Μεσολαβητή

Στη συνέχεια μελετούνται συστήματα όπου ο μεσολαβητής QoS λειτουργεί ως front-end ενός εξυπηρετητή. Σε αυτή την εξειδικευμένη αρχιτεκτονική εξυπηρετητή-μεσολαβητή (σχήμα 13) συνδυάζονται οι πληροφορίες QoS και το υποσύστημα κατάστασης (status module) ενός υποσυστήματος διαχείρισης πληροφοριών ποιότητας υπηρεσίας σε ένα μεσολαβητή. Επιπλέον, το υποσύστημα αποδοχής QoS συνδυάζεται με τον διαχειριστή διαπραγμάτευσης QoS. Αυτά τα δυο νέα υποσυστήματα (Διαχειριστής Πληροφορίας QoS και Αποδοχή QoS) ανήκουν όλα στον μεσολαβητή, δηλαδη ο μεσολαβητής ελέγχει την αποδοχή για έναν εξυπηρετητή και καθορίζει πόσους πόρους θα πρέπει να διαθέσει σε κάθε πελάτη.



Σχήμα 12: Αρχιτεκτονική Server-Broker QCWS

Οι εξυπηρετητές για Υπηρεσίες Διαδικτύου συνήθως έχουν πολύ δυναμικό φόρτο εργασίας καθως οι πελάτες έρχονται και φεύγουν. Το πρότυπο άφιξης των πελατών, οι ανάγκες πόρων τους και οι τρόποι χρήσης των πόρων είναι απρόβλεπτοι. Εάν ένας εξυπηρετητής αναθέτει υπερβολικούς πόρους του συστήματος σε πελάτες, νέες αιτήσεις μπορεί να μην γίνουν αποδεκτές. Μια άλλη πιθανότητα είναι ο εξυπηρετητής να μειώσει την ποιότητα υπηρεσίας ενός ή περισσότερων πελατών για να δεχτεί νέες αιτήσεις. Η μείωση ποιότητας διαταράσει τη σταθερότητα των ενεργειών του πελάτη και ονομάζεται αστάθεια QoS. Το περιβάλλον του συστήματος μοντελοποιήθηκε ως ένα σύνολο από πελάτες που ζητούν υπηρεσίες τυχαία, μπαίνουν στο σύστημα για να χρησιμοποιήσουν πόρους και έπειτα βγαίνουν από το σύστημα ελευθερώνοντας τους πόρους που τους είχαν διατεθεί. Το μοντέλο του συστήματος μπορεί να χαρακτηριστεί από τις εξής παραμέτρους:

- Όλοι οι πελάτες έχουν ίδια συνάρτηση χρησιμότητας (utility) U(r), που είναι αυξανόμενη και κοίλη
- Ο συνολικός αριθμός πελατών που μπορούν να ζητήσουν μια υπηρεσία είναι N (σταθερός)
- Ο συνολικός αριθμός πόρων είναι R
- Οι πελάτες είναι ανεξάρτητοι μεταξύ τους

Οι όροι που χρησιμοποιούνται στη μελέτη είναι:

- **Συνολική Χρησιμότητα Συστήματος (Ug(t))**: Άθροισμα της χρησιμότητας όλων των πελατών σε χρόνο t
- **Μέση Χρησιμότητα Συστήματος (Uavg):** Το διάστημα της στιγμιαίας συνολικής χρησιμότητας του συστήματος αναφορικά με το χρόνο ως προς τη χρονική περίοδο στην οποία μετράται
- **Επαναρύθμιση:** Επαναρύθμιση σημαίνει ότι οι πόροι ξαναμοιράζονται στους υπάρχοντες πελάτες και τον εισερχόμενο πελάτη. Συμβαίνει κάθε φορά που υπάρχει αστάθεια QoS.
- **Ρυθμός επαναρύθμισης:** Ο συνολικός αριθμός από επαναρυθμίσεις διαιρεμένος με το χρόνο στον οποίο μετράται.

6.2.2. Αλγόριθμοι QCWS Resource Allocation

6.2.2.1. HQ: Αλγόριθμος Ομογενούς Resource Allocation

Πολλοί πάροχοι Υπηρεσιών Διαδικτύου παρέχουν legacy εξυπηρετητές που δεν έχουν κανένα έλεγχο στην ποιότητα υπηρεσίας. Σε αυτή την περίπτωση, οι πελάτες πρέπει να βασίζονται στον μεσολαβητή για να βεβαιώσουν το επιθυμητό επίπεδο QoS. Ο μεσολαβητής χρησιμοποιεί ομοιογενή κατανομή των πόρων για QoS αποδοχή και ανάθεση. Η ιδέα του αλγορίθμου ομογενούς κατανομής πόρων είναι να κατανέμει ισότημα τους διαθέσιμους πόρους μεταξύ όλων των πελατών που χρησιμοποιούν την υπηρεσία. Μια βασική υπόθεση για την υιοθέτηση του αλγορίθμου είναι ότι όλοι οι πελάτες πρέπει να έχουν τις ίδιες ελάχιστες απαιτήσεις QoS.

Η υπόθεση αυτή δεν είναι όσο περοριστική δείχνει και είναι αρκετά συνηθισμένη. Για παράδειγμα, στην παρακολούθηση βίντεο μέσω Διαδικτύου, η αποδεκτή ποιότητα εικόνας είναι σχεδόν ίδια για κάθε χρήστη. Αν θεωρηθεί ότι το ποσό των πόρων που χρησιμοποιούνται για να παράγουν αυτό το κατώφλι ποιότητας είναι r μονάδες, και υπάρχει ένα σύνολο από R μονάδες πόρων στο σύστημα, τότε ο μέγιστος αριθμός πελατών που μπορεί να δεχτεί ο εξυπηρετητής είναι R/r έτσι ώστε όλοι να λάβουν ένα αποδεκτό αποτέλεσμα εξόδου. Όταν υπάρχουν λιγότεροι από R/r πελάτες στο σύστημα (π.χ. a, όπου a<R/r), ο μεσολαβητής μπορεί να δώσει σε κάθε πελάτη περισσότερους πόρους (R/a>r) και αυτοί θα λάβουν καλύτερη ποιότητα από την

αναμενόμενη. Όσο περισσότεροι πελάτες μπαίνουν στο σύστημα, μειώνεται το μερίδιο πόρων που τους αντιστοιχεί και η ποιότητα που λαμβάνουν μειώνεται. Μόλις ο αριθμός των πελατών φτάσει το R/r, ο μεσολαβητής δε μπορεί να δεχτεί νέες αιτήσεις γιατί θα προκαλέσει την πτώση της ποιότητας όλων των πελατών κάτω από το επιθυμητό όριο.

Ένας αλγόρθμος ομογενούς ανάθεσης πόρων PB(A1, A2) παρουσιάζεται στο [16]. Ο αλγόριθμος ορίζει δυο κατώφλια A1 και A2 (A1 ≤ R ≤ A2). Ανάλογα με τον αριθμό ενεργών έργων K στο σύστημα, το σύστημα διαθέτει διαφορετικά ποσά πόρων σε κάθε έργο (R/A1, R/A2 ή R/K). Στο QCWS, επεκτείνεται ο παραπάνω αλγόριθμς ομοιογενούς ανάθεσης πόρων σε περισσότερα στάδια, (A1, A2, … , Am) με A1 ≤ A2 ≤ … ≤ Am. Τα κατώφλια επιλέγονται από στατιστικά δεδομένα από προηγούμενες εκτελέσεις. Ο αλγόριθμος HQ λειτουργεί ως εξής:

1. Από ιστορικές στατιστικές πληροφορίες στον αριθμό των πελατών που ζήτησαν την υπηρεσία, παράγεται η συνάρτηση συγκεντρωτικής πιθανοτικής κατανομής (cdf), που ονομάζεται F( ), του αριθμού των πελατών στο σύστημα

2. Χρησιμοποιείται η F και ένα προκαθορισμένο μέγεθος βήματος (STEP, 0<STEP<0.5) για να αναγνωρισθούν τα σημεία κατωφλίων A1 .. Am:

```
k=0; Threshold = STEP;

for client_number =1 to N
   if F(client_number) ≥ Threshold
     Ak = client_number; k=k+1;
     Threshold = Threshold + STEP;
   Endif
endfor
```

3. Ορίζεται η πολιτική κατανομής: Εάν ο τρέχον αριθμός πελατών είναι ≤ A1, οι πόροι που δίνονται σε κάθε πελάτη είναι R/ A1

Εάν Ai ≤ τρέχον αριθμός πελατών ≤ Ai+1(1 ≤ i < m), οι πόροι που δίνονται σε κάθε πελάτη είναι R/Ai+1. Εάν ο τρέχον αριθμός πελατών είναι ≥ Am, οι πόροι που δεσμεύονται είναι R/(αριθμός πελατών).

Επαναρυθμίσεις	**Αριθμός πελατών**
1. Ο αριθμός των πελατών αλλάζει από A_i σε A_i+1 ή από A_{i+1} σε A_i ($1 \le i \le m$)	$\mathbf{A_i}$
2. Ο αριθμός των πελατών είναι k ≥ A_m κάθε φορά που ένας πελάτης αναχωρεί από το σύστημα. (Ο αριθμός των πελατών αλλάζει από k σε k+1 ή από k+1 σε k).	**k**

Πίνακας 15: Οι επανακατανομές που συμβαίνουν και οι πελάτες που επαναδιαμορφώνονται.

Στον HQ, όσο μικρότερο είναι το μέγεθος του βήματος, τόσο περισσότερα κατώφλια και στάδια θα υπάρχουν. Παρόλο που μπορεί να πετύχει μεγαλύτερη μέση χρησιμότητα συστήματος, θα προκαλέσει επίσης έναν υψηλότερο ρυθμό επαναρυθμίσεων.

6.2.2.2. RQ: Μη ομογενής Resource Allocation

Εάν ένας μεσολαβητής λειτουργεί ως front-end ενός QoS-capable εξυπηρετήτη, ένας αλγόριθμος μη ομοιογενούς κατανομής μπορεί να χρησιμοποιηθεί για κατανομή των πόρων. Εφόσον διαφορετικοί πελάτες μπορεί να έχουν διαφορετικές QoS απαιτήσεις, ο μεσολαβητής

QoS θα έπρεπε να αποφασίσει ποιο επίπεδο υπηρεσίας επιλέγεται για κάθε πελάτη. Η βασική ιδέα του αλγορίθμου μη ομοιογενούς κατανομής RQ είναι να κατανέμει διαφορετικά ποσά πόρων για διαφορετικούς πελάτες ανάλογα με τις απαιτήσεις τους. Για να μειώσει την αστάθεια, ο μεσολαβητής δημιουργεί έναν εικονικό πελάτη για να δεσμεύσει κάποιους αχρησιμοποίητους πόρους. Για κάθε εισερχόμενη αίτηση υπηρεσίας, εάν οι δεσμευμένοι πόροι είναι αρκετοί (πάνω από το καθορισμένο κατώφλι), ο μεσολαβητής δεσμεύει άμεσα το απαιτούμενο ποσό πόρων για το νέο πελάτη. Εάν οι δεσμευμένοι πόροι δεν είναι αρκετοί, ο μεσολαβητής θα πρέπει να επαναρυθμίσει την κατανομή πόρων μεταξύ των υπάρχοντων πελατών ώστε να επιτρέψει στον εισερχόμενο πελάτη να λάβει ένα ικανοποιητικό επίπεδο ποιότητας. Στον RQ, κάθε πελάτης έχει μια συνάρτηση χρησιμότητας. Ο εικονικός πελάτης έχει επίσης μια συνάρτηση χρησιμότητας. Η συνολική χρησιμότητα του συστήματος καθορίζεται από το άθροισμα όλων των χρησιμοτήτων των πελατών και την χρησιμότητα που επιτυγχάνεται από τον εικονικό πελάτη. Ο αλγόριθμος RQ περιγράφεται παρακάτω.
Σημειώσεις:

- W: το βάρος του εικονικού πελάτη
- Nr: ο αριθμός των πελατών που θα επαναρυθμιστούν σε μια επαναρύθμιση
- Tl: low-water επίπεδο, δείχνει εάν υπάρχουν δεσμευμένοι πόροι στον εικονικό πελάτη
- Uc(r): η συνάρτηση χρησιμότητας του πελάτη
- Uvc(r): η συνάρτηση χρησιμότητας του εικονικού πελάτη

Αλγόριθμος RQ:
1. Χρησιμοποιώντας τις ιστορικές στατιστικές πληροφορίες, ο μεσολαβητής αρχικά παράγει το μέσο αριθμό πελατων (avgc) και τη συνάρτηση F( ) των αριθμών cdf των πελατών
2. Σύμφωνα με τις ελάχιστες απαιτήσεις του πελάτη, ο μεσολαβητής αποφασίζει το κατώτερο επίπεδο υπηρεσίας για τον πελάτη
3. Καθορίζει τον τύπο υπολογισμού του low-water επιπέδου T_l. Στον QCWS, ορίζεται το Tl ως μια συνάρτηση του τρέχοντα αριθμού πελατών (curc)στο σύστημα:
$$T_l = (1 - F(cuc)) * avgc/N \quad (1)$$
4. Αρχικά οι δεσμευμένοι πόροι είναι v =R
5.Όριζεται το βάρος του εικονικού πελάτη (W) και ο αριθμός των πελατών Nr που πρέπει να επαναρυθμιστούν όταν οι δεσμευμένοι πόροι είναι κάτω από T_l *R
6. Όταν φτάσει ένας πελάτης, η πολιτική κατανομής εκτελείται ως εξής:

Υπολόγισε το T_l
If (v $\geq$ T_l *R)
Διαθέσε τους πόρους r στους εισερχόμενους πελάτες για να βελτιστοποιήσεις την joint utility U:
$U = U_c(r) + W * U_{vc} (v - r);$
Else
{
Αποφάσισε ποιοί Nr πελάτες θα ξαναρυθμιστούν
Επαναρύθμισε Nr πελάτες για να βελτιστοποιήσεις την joint utility U:
$U = (Nr + 1) * Uc(r) + W * Uvc (v - (Nr + 1) * r);$
}
Ξαναυπολόγισε τους δεσμευμένους πόρους

Οι κανόνες για την επιλογή των Nr πελατών που θα επαναρυθμιστούν είναι:

- Επέλεξε τους πελάτες που έχουν μεγαλύτερες διαφορές μεταξύ των ελάχιστων απαιτήσεών τους και τους πόρους που τους έχουν ήδη ανατεθεί.

- Εάν το αποτέλεσμα της νέας ρύθμισης προκαλέσει σε ένα πελάτη να λαμβάνει ποιότητα υπηρεσίας κάτω από το ελάχιστο όριο το οποίο θα έπρεπε, αφαίρεσε αυτόν τον πελάτη από τη λίστα Nr και επέλεξε έναν άλλο υποψήφιο σύμφωνα με τον κανόνα b. Επανέλαβε αυτό το βήμα μέχρι να βρεθεί διαθέσιμη κατανομή.

6.3. Μελέτη Απόδοσης

Στα πλαίσια της μελέτης πραγματοποιήθηκε μια προσομοίωση για την απόδοσης των αλγορίθμων HQ και RQ. Στη συνέχεια παρουσιάζονται τα αποτελέσματα της προσομοίωσης.

6.3.1. Υποθέσεις

Οι ακόλουθες υποθέσεις χρησιμοποιούνται στην προσομοίωση.

- Υπάρχουν συνολικά N πελάτες
- Οι πελάτες είναι ανεξάρτητοι μεταξύ τους
- Το συνολικό ποσό πόρων είναι R, R<N
- Το ποσό των πόρων που έχει ανατεθεί σε κάθε πελάτη είναι μεταξύ 0 και1
- Όλοι οι πελάτες έχουν την ίδια συνάρτηση χρησιμότητας

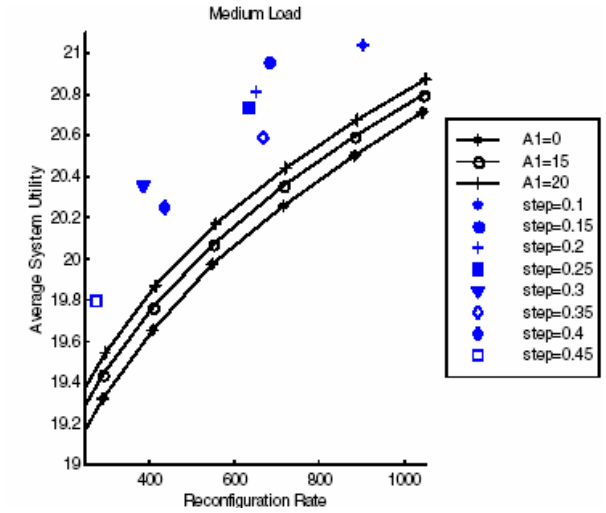
$$U_e(r) = \frac{1}{1-e^{-1}}(1-e^{-r}) \quad 0 \le r \le 1$$

- Η άφιξη και η αναχώρηση των πελατών κατανέμονται εκθετικά με λ=μ=1
- Η συνάρτηση χρησιμότητας για τον εικονικό πελάτη στο RQ είναι:

$$U_{res}(r) = \frac{1}{1-e^{-\alpha}}(1-e^{-\alpha r/R}) \quad 0 \le r \le R$$

Στην προσομοίωσή αυτή, οι συνολικοί πόροι είναι R=20. Καθορίζεται χαμηλό φορτίο συστήματος με N=40, ένα μέσο φορτίο με N=60 και ένα υψηλό φορτίο με N=80. Υπάρχει επίσης η υπόθεση ότι η ελάχιστη απαίτηση του πελάτη μπορεί πάντα να ικανοποιηθεί. Η απόδοση του συστήματος μετριέται από το tradeoff από τη μέση χρησιμότητα συστημάτων εναντία στο ποσοστό επαναρύθμισης.

6.3.2. Απόδοση του HQ

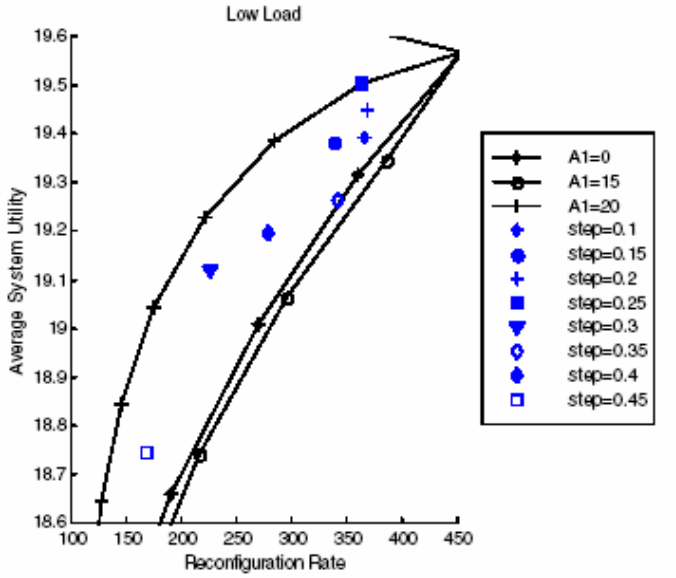


Σχήμα 13: Ρυθμός επαναρυθμίσεων και μέση χρησιμότητα χρησιμοποιώντας τον HQ (N=60)

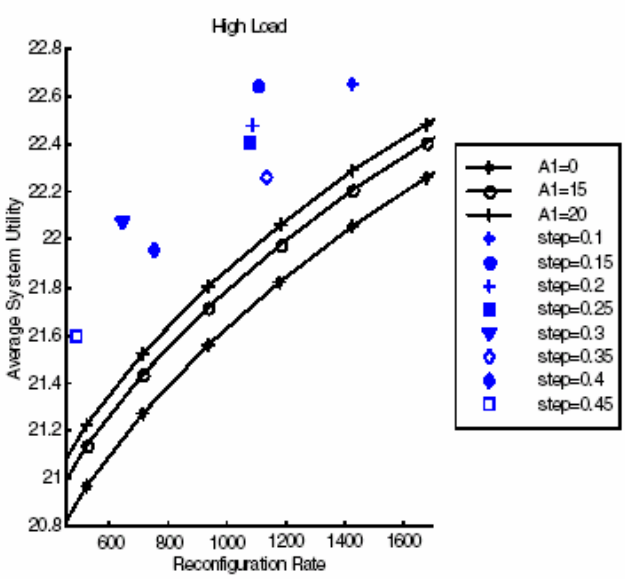
Το σχήμα 13 παρουσιάζει τη μέση χρησιμότητα του συστήματος ως πρός τον ρυθμό επαναρυθμίσεων για τον HQ αλγόριθμο σε ένα μεσαίου φόρτου σύστημα. Μελετούνται δύο πολιτικές:

- Πολιτική 1: Η βασική πολιτική PB(A1, A2) με τρια επίπεδα πελάτη, A1=0,15,20 και A2=R ως N. Οι τρείς χαμηλότερες γραμμές στο σχήμα 14 δείχνουν τα αποτελέσματα. Κάθε γραμμή είναι μια συγκεκριμένη τιμή της A1, και τα σημεία σε μια καμπύλη αντισοιχούν σε πολλές διαφορετικές A2 τιμές.
- Πολιτική 2: Ο multi-segment αλγόριθμος ομοιογενούς κατανομής HQ (A1, A2,…, Am). Τα σημεία κατωφλίων καθορίζονται από την συγκεντρωτική πιθανοτική κατανομή του αριθμού των ενεργών πελατών και το μέγεθος του βήματος. Επιλέγεται το μέγεθος βήματος από 0.1 ως 0.45, με αυξήσεις 0.05, και παράγονται τα σημεία κατωφλίων για κάθε μέγεθος βήματος. Τα αποτελέσματα για διαφορετικά μεγέθη βήματος φαίνονται στο σχήμα 15 Εφόσον χρησιμοποιείται το μέγεθος βήματος για να αποφασιστεί το A1, A2, … , Am, κάθε μέγεθος βήματος αντιστοιχεί σε ένα σημείο στο γράφημα.

Στο σχήμα 13 φαίνεται ότι για μεσαίου φόρτου συστήματα η πολιτική 2 έχει πάντα καλύτερα αποτελέσματα από την πολιτική 1. Για κάποιο ρυθμό επαναρύθμισης, η πολιτική 2 μπορεί να πετύχει υψηλότερη μέση χρησιμότητα συστήματος. Ανάμεσα σε όλα τα μεγέθη βήματος που χρησιμοποιούνται στην πολιτική 2, τα μικρότερα βήματα συνήθως μπορούν να πετύχουν υψηλότερη χρησιμότητα κάνοντας περισσότερες επαναρυθμίσεις. Η υψηλότερη χρησιμότητα επιτυγχάνεται σε βάρος του υψηλού ρυθμού επαναρύθμισης. Παρόλα αυτά, παρατηρείται ότι το βήμα μεγέθους 0.15 έχει το καλύερο tradeoff. Τα σχήματα 14 και 15 δείχνουν την μέση χρησιμότητα του συστήματος ως προς τον ρυθμό επαναρύθμισης σε συστήματα χαμηλού και υψηλού φόρτου. Το σχήμα 14 δείχνει για χαμηλού φόρτου συστήματα , όπου A1=20 στην πολιτική 1 μπορεί να πετύχει την καλύτερη απόδοση του συστήματος, αλλά η πολιτική 2 είναι ακόμα καλύτερη για A1=0 και 15 της πολιτικής 1. Για υψηλού φόρτου συστήματα, το σχήμα δείχνει ότι η πολιτική 2 είναι συνεχώς καλύτερη από την 1. Ανάλογα φαίνεται ότι το βήμα 0.15 μπορεί να πετύχει το καλύτερο tradeoff. Επιπλέον, όσο πιο μεγάλος ο φόρτος του συστήματος τόσο καλύτερη είναι η πολιτική 1 σε σύγκριση με την πολιτική 2. Αυτό σημαίνει ότι ο αλγόριθμος HQ χρησιμοποιείται καλύτερα σε υψηλού και μεγάλου φόρτου συστήματα.



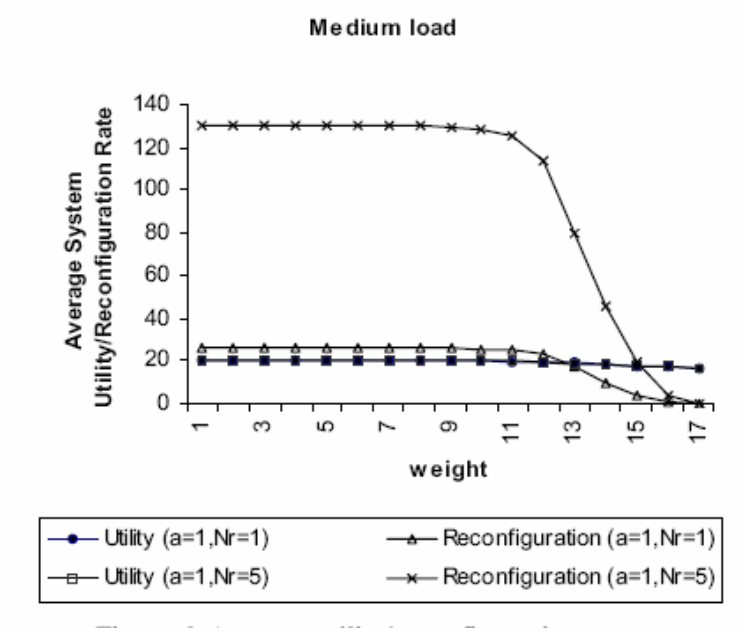
Σχήμα 14: Ρυθμός επαναδιαμόρφωση και μέση utility χρησιμοποιώντας τον HQ (N=40)



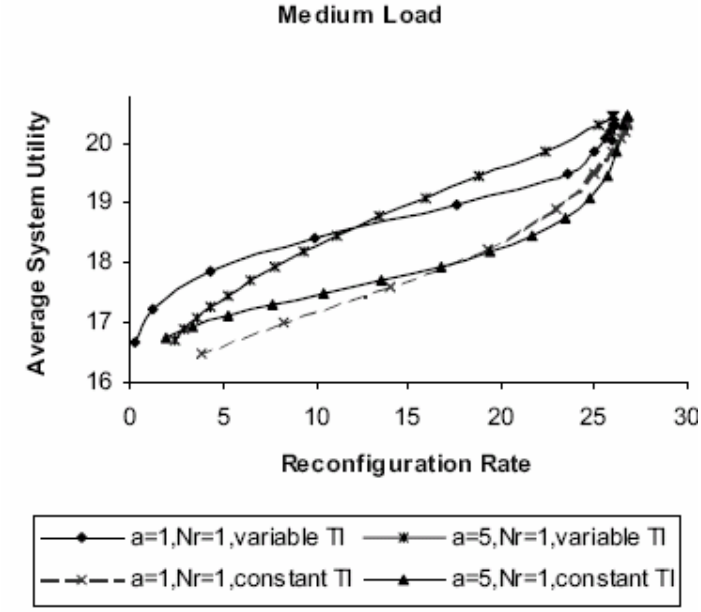
Σχήμα 15: Ρυθμός επαναδιαμόρφωση και μέση utility χρησιμοποιώντας τον HQ(N=80)

6.3.3. Απόδοση του RQ

Για τον αλγόριθμο μη ομοιογενούς κατανομής πόρων RQ, έχει φτιαχτεί μια προσομοίωση σε συστήματα μεσαίου φόρτου για [W= 1...17 βήμα 1, α = 1,5; Nr = 1,5]. Το σχήμα 16 δείχνει τη σύγκριση για μεσαία χρησιμότητα συστήματος και την επαναρύθμιση συστήματος μεταξύ N_r =1 και 5 όταν N_r= 1. Αυτό δείχνει ότι παρόλο που οι χρησιμότητες συστήματος είναι σχεδόν ίδιες, ο ρυθμός επαναρύθμισης είναι πολύ υψηλότερος για μεγαλύτερα N_r (N_r =5). Η μελέτη του N_r =1 παίζει ρόλο στην επιλογή των W, α και T_l. Για ρυθμίσεις low-water επιπέδου, συγκρίνεται η απόδοση που επιτυγχάνεται χρησιμοποιώντας Tl σύμφωνα με την Eq. (1) και την απόδοση χρησιμοποιώντας τη σταθερά που ορίζεται από το (1- F(avgc))*avgc/N. Το σχήμα 17 δείχνει το αποτέλεσμα. Προκύπτει ότι η χρήση της T_l ως συνάρτηση του αριθμού πελατών είναι καλύτερη από τη χρήση της T_l ως σταθεράς. Αυτό συμβαίνει επειδή όταν υπάρχουν λιγότεροι πελάτες στο σύστημα, η πιθανότητα να φτάσουν νέοι πελάτες είναι μεγαλύτερη και περισσότεροι πόροι θα πρέπει να δεσμευθούν γι'αυτούς, ενώ όταν υπάρχουν περισσότεροι πελάτες στο σύστημα, η πιθανότητα για νέες αφίξεις πελατών θα μειωθεί και λιγότεροι πόροι πρέπει να δεσμευθούν γι' αυτούς. Το σχήμα 17 δείχνει επίσης την περίπτωση όπου με μεταβλητή T_l, όταν ο ρυθμός επαναρύθμισης είναι χαμηλός (<13), η απόδοση για [α =1, N_r =1] είναι ελαφρώς καλύτερη από τις άλλες. Όταν ο ρυθμός επαναρύθμισης είναι υψηλός, [α =5, N_r =1] μπορεί να πετύχει τη βέλτιστη απόδοση. Η διαφορά μεταξύ τους είναι πολύ μικρή. Έτσι, και οι δύο τιμές είναι αποδεκτές



Σχήμα 16: Μέση utility/ επαναδιαμόρφωση vs βάρος



Σχήμα 17: Απόδοση με χρήση μεταβλητής και σταθεράς T_1

Αφού έχει αποφασιστεί το a, το N_r και το T_l, πρέπει να αποφασιστεί το W, το βάρος του εικονικού πελάτη στον μεσολαβητή. Χρησιμοποιείται η ακόλουθη συνάρτηση χρησιμότητας μεσολαβητή b(W) για να μετρηθεί η συνολική αξία κάτω από διαφορετικούς ρυθμούς επαναρύθμισης και μεσαία χρησιμότητα συστήματος σε διαφορετικές W τιμές.

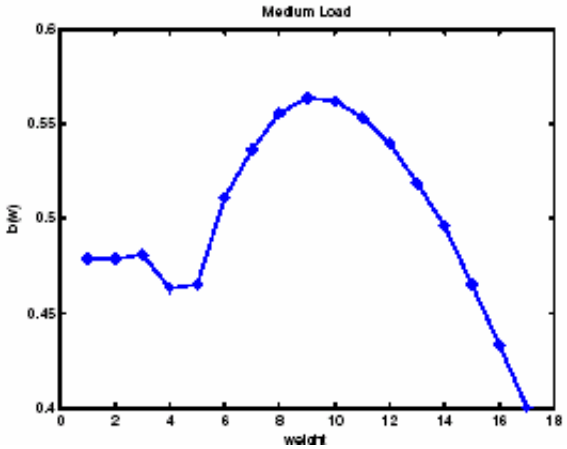
$$b(w) = w_r * (1 - \frac{r - avgr}{stdr}) + w_u * (\frac{u - avgu}{stdu})$$

Όπου:

- w_r : το βάρος του ρυθμού επαναρύθμισης
- w_u : το βάρος για τη μέση utility συστήματος, $w_r + w_u = 1$

- r,u: ρυθμός επαναρυθμίσεων και μέση χρησιμότητα συστήματος σε βάρος weight=w
- avgr,avgu: μέσος ρυθμός επαναρύθμισης και μέση χρησιμότητα συστήματος για όλα τα βάρη που μετρήθηκαν
- stdr, stdu: τυπική απόκλιση του ρυθμού επαναρύθμισης και μέσης χρησιμότητα του συστήματος για όλα τα βάρη που μετρήθηκαν

Το σχήμα 18 δείχνει τη χρησιμότητα του μεσολαβητή ως προς το βάρος [α=5, Nr =1] and w_r = w_u =0.5.Δείχνει ότι με w =9, ο μεσολαβητής μοπρεί να πετύχει τη βέλτιστη απόδοση.



Σχήμα 18: Απόδιση του μεσολαβητή ως προς το βάρος

6.4. Συμπεράσματα για τον QCWS

Σε αυτή την εργασία, μελετήθηκε μια αρχιτεκτονική Υπηρεσιών Διαδικτύου με βάση χρήση μεσολαβητή μεταξύ εξυπηρετητών και πελατών. Μελετήθηκαν οι δύο αλγόριθμοι κατανομής πόρων, ο HQ και ο RQ, που χρησιμοποιούνται από τον μεσολαβητή QoS στην περίπτωση που ο μεσολαβητής λειτουργεί ως το front-end ενός εξυπηρετητή. Ο σκοπός των αλγορίθμων είναι να πετύχει υψηλή μέση χρησιμότητα συστήματος και να αποφύγει τις συχνές επαναρυθμίσεις των πόρων. Χρησιμοποιώντας έναν αριθμό πελατών μέσα σε ένα εξυπηρετητή, ο εξυπηρετητής προσπαθεί να διατηρήσει ένα σταθερό ποσό πόρων για κάθε πελάτη μέχρι ο αριθμός τους να αυξηθεί ή να μειωθεί σημαντικά. Τότε, θα γίνει κάποια ρύθμιση των πόρων για να επιτευχθεί υψηλή χρησιμοποιήση του συστήματος. Με τους αλγόριθμους που προτείνονται, η κατανομή πόρων σε ένα εξυπηρετητή μπορεί να προσαρμοστεί αποτελεσματικά έτσι ώστε οι πελάτες να μη αντιμετωπίζουν πολύ ασταθείς αποδόσεις

.

7. Χειρισμός Εξαιρέσεων σε Συνεργατικές Διαδικασίες με βάση Υπηρεσίες Διαδικτύου

Οι Υπηρεσίες Διαδικτύου χρησιμοποιούνται όλο και περισσότερο στην ενοποίηση ετερογενών και αυτόματων εφαρμογών σε δια-επιχειρησιακές συνεργασίες. Ένα βασικό πρόβλημα που αντιμετωπίζεται είναι η υποστήριξη υψηλής ποιότητας εκτέλεσης των σύνθετων συνεργατικών διαδικασιών. Μια σημαντική πτυχή που δεν έχει αντιμετωπιστεί ικανοποιητικά μέχρι στιγμής είναι ο δυναμικός χειρισμός εξαιρέσεων κατά την εκτέλεση της διαδικασίας. Για την αντιμετώπιση αυτού του προβλήματος προτείνεται μια προσέγγιση με κανόνες (rules based) για τον αυτόματο έλεγχο των περιορισμών για συνεργατικές διαδικασίες με βάση Υπηρεσίες Διαδικτύου [19].

Η συνεργασία εταιριών και οργανισμών οδηγεί στην αύξηση των συνεργατικών επιχειρησιακών διαδικασιών που ενοποιούν αυτόνομες και ετερογενείς διαδικασίες. Οι Υπηρεσίες Διαδικτύου χρησιμοποιούνται με αυξανόμενο ρυθμό στη διευκόλυνση τέτοιων ενοποιήσεων εφαρμογών καθώς περικλείουν εφαρμογές και παρέχουν αναγνώσιμες από μηχανές, βασισμένες σε XML διεπαφές για κλήσεις υπηρεσιών, επιτρέποντας έτσι στους παρόχους υπηρεσιών να διατηρούν την αυτονομία τους. Οι Υπηρεσίες Διαδικτύου μπορούν να χρησιμοποιηθούν για να περικλύσουν σύνθετες legacy εφαρμογές είτε ολόκληρα workflows. Η υλοποίηση πραγματικών συνεργατικών διαδικασιών απαιτεί την ενοποίηση Υπηρεσιών Διαδικτύου διαφορετικής πολυπλοκότητας και διαφορετικών παρόχων. Για παράδειγμα, σε μια συνεργατική διαδικασία διαφορετικοί συνεργάτες συνεργάζονται για να ικανοποιήσουν τα αιτήματα του πελάτη, σύμφωνα με καθορισμένο χρόνο παράδοσης και άλλους περιορισμούς ποιότητας.

Η διαβεβαίωση ότι τέτοιες διαδικασίες εξυπηρετούν αξιόπιστα το σκοπό τους αποτελεί πρόκληση λόγο του υψηλού βαθμού αυτονομίας και ανομοιογένειας των συνεργατών. Η επίτευξη υψηλής ποιότητας εκτέλεσης Υπηρεσιών Διαδικτύου επηρεάζεται από διάφορα χαρακτηριστικά των υπηρεσιών, όπως ο χρόνος απόκρισης, το κόστος, είτε περιορισμοί των παραμέτρων εισόδου και εξόδου. Η υποστήριξη της ποιότητας υπηρεσίας Υπηρεσιών Διαδικτύου είναι ένα θέμα που απασχολεί τις πρόσφατες μελέτες, πολλές από τις οποίες εστιάζουν στη δυναμική επιλογή του καλύτερου παρόχου για μια συγκεκριμένη Υπηρεσία Διαδικτύου, ενώ άλλες στη δυναμική διαχείριση πόρων για την υποστήριξη ικανοποιητικά γρήγορης εκτέλεσης Υπηρεσιών Διαδικτύου.

Παρόλα αυτά, οι περισσότερες προσεγγίσεις δεν έχουν ευέλικτους μηχανισμούς χειρισμού εξαιρέσεων για την υποστήριξη της ποιότητας εκτέλεσης συνεργατικών διαδικασιών. Σε προηγούμενες έρευνες έχουν προταθεί πρότυπα τα οποία υποστηρίζουν ένα βασικό χειρισμό εξαιρέσεων ελέγχοντας εάν προκύπτει ένα προκαθορισμένο μήνυμα λάθους σε συγκεκριμένα βήματα της διαδικασίας. Τέτοιες εξαιρέσεις αντιμετωπίζονται καλώντας εναλλακτικές υπηρεσίες. Παρόλα αυτά, οι εξαιρέσεις που δεν καλύπτονται από τα μηνύματα λάθους, όπως παραβιάσεις των ποιοτικών περιορισμών, ή που δεν προκύπτουν σε προκαθορισμένα βήματα της διαδικασίας, δε μπορούν να ανιχνευθούν αποτελεσματικά. Επομένως απαιτείται μια πιο ευέλικτη προσέγγιση χειρισμού λαθών ώστε να υποστηρίξει κατάλληλα την ποιότητα των συνεργατικών υπηρεσιών.

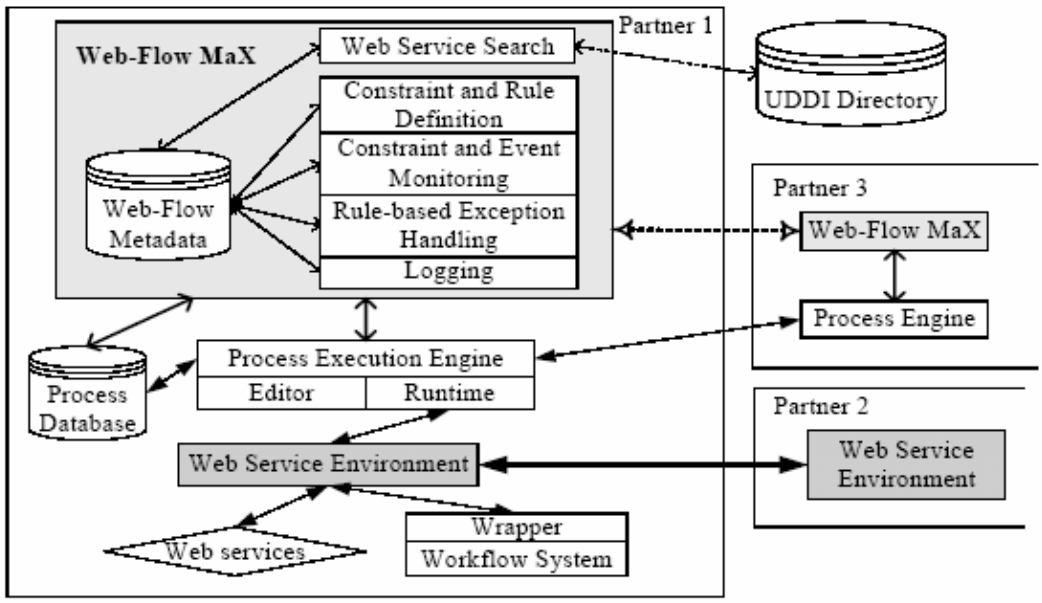
Στην συνέχεια παρουσιάζεται μια νέα προσέγγιση δυναμικού χειρισμού εξαιρέσεων σε διαδικασίες με βάση Υπηρεσίες Διαδικτύου που υποστηρίζει τον καθορισμό περιορισμών ποιότητας υπηρεσιών, επιπλέον των συνθηκών που μπορεί να προσφέρει μια υπηρεσία. Χρησιμοποιείται μια προσέγγιση με κανόνες για χειρισμό εξαιρέσεων όπως η παραβίαση περιορισμών ή άλλων γεγονότων που συμβαίνουν κατά την εκτέλεση της διαδικασίας.

7.1. Αρχιτεκτονική του Web-Flow

Το σύστημα Web-Flow που παρουσιάζεται εδώ στοχεύει στη παροχή ποιότητας για συνεταιριστικές επιχειρησιακές διαδικασίες ενοποιώντας τις Υπηρεσίες Διαδικτύου που είναι διαθέσιμες τοπικά ή παρέχονται από εξωτερικούς συνεργάτες. Διαχωρίζει τις λειτουργικότητες

παρακολούθησης και χειρισμού εξαιρέσεων μέσα σε ένα ειδικευμένο υποσύστημα λογισμικού, το MaX (Monitoring and eXception handling). Αυτό επιτρέπει μια γενική λύση που μπορεί να χρησιμοποιηθεί σε συνδυασμό με διαφορετικές μηχανές εκτέλεσης διαδικασιών για καθορισμό και εκτέλεση των συνεργατικών διαδικασιών (Σχήμα 19). Η μηχανή εκτέλεσης διαδικασιών χρησιμοποιεί ένα περιβάλλον Υπηρεσιών Διαδικτύου για να καλέσει τοπικές και εξωτερικές Υπηρεσίες Διαδικτύου περιλαμβάνοντας απλές εφαρμογές είτε ολόκληρα workflows. Εναλλακτικά, εξωτερικές υπηρεσίες μπορεί να κληθούν από τη μηχανή επεξεργασίας του συνεργάτη. Οι υπηρεσίες υποτίθεται ότι παρέχουν μια WSDL διεπαφή, αν απαιτείται μέσο ενός wrapper.

Οι συνεργατικές (cooperative) διαδικασίες μπορεί να καθοριστούν σύμφωνα με δύο βασικά μοντέλα συνεργασίας. Σε ένα απλό μοντέλο συνεργασίας (cooperation model), μια συνεργατική διαδικασία αντιστοιχεί σε ένα workflow του οποίου οι δραστηριότητες μπορεί να αναφέρονται σε εξωτερικές Υπηρεσίες Διαδικτύου. Επομένως, οι πάροχοι των ενοποιημένων υπηρεσιών δε γνωρίζουν τη συνολική διαδικασία. Σε ένα σύνθετο μοντέλο συνεργασίας ,οι εμπλεκόμενοι συνεργάτες συμφωνούν σε μια συνεργατική διαδικασία (συμπεριλαμβανομένων των υπηρεσιών που χρησιμοποιούνται και της σειράς με την οποία καλούνται). Οι συνεργάτες έχουν ισοδύναμη λειτουργικότητα και αλληλεπιδρούν με peer-to-peer τρόπο. Αυτό σημαίνει μειωμένη αυτονομία καθώς οι αλλαγές στις συνεργατικές διαδικασίες πρέπει να είναι συντονισμένες μεταξύ των συνεργατών. Το Web-Flow υποστηρίζει και τα δύο μοντέλα: το πρώτο μοντέλο μπορεί να υποστηριχθεί χρησιμοποιώντας την λειτουργικότητα της Υπηρεσίας Διαδικτύου, το δεύτερο μπορεί να υποστηριχθεί με την εκμετάλλευση της άμεσης επικοινωνίας μεταξύ των μηχανών επεξεργασίας.



Σχήμα 19: Η Web Flow αρχιτεκτονική

Αρχικά, το μοντέλο απλής συνεργασίας πρέπει να ελέγχει τη συνεργατική διαδικασία. Αυτή η προσέγγιση υποστηρίζει μέγιστη αυτονομία στους παρόχους υπηρεσιών αλλά υπονοεί ότι η παρακολούθηση ποιότητας και ο χειρισμός εξαιρέσεων πρέπει να εκτελεστούν κατά μεγάλο μέρος από τον κόμβο που ελέγχει τη συνεργατική διαδικασία.Το υποσύστημα λογισμικού Web-Flow MaX αποτελείται από τέσσερα βασικά μέρη:

- Το τμήμα **περιορισμών και καθορισμών κανόνων** (constraint and rule definition) χρησιμοποιείται για να καθορίσει τους περιορισμούς ποιότητας για κλήσεις

υπηρεσιών και τους κανόνες χειρισμού εξαιρέσεων για γεγονότα που συμβαίνουν κατά τη διάρκεια της εκτέλεσης διαδικασίας.

- Το τμήμα **περιορισμών και παρακολούθησης γεγονότων** (constraint and event monitoring) ελέγχει εάν οι καλούμενες υπηρεσίες παραβιάζουν κάποιο περιορισμό ή εάν συμβαίνει κάποιο άλλο γεγονός.
- Το τμήμα **χειρισμού εξαιρέσεων** (exception handling) χρησιμοποιεί τους καθορισμένους κανόνες για να καθορίσει εάν ένα γεγονός αποτελεί εξαίρεση και πως πρέπει να αντιμετωπιστεί. Ο κύριος στόχος της διαχείρισης εξαιρέσεων είναι να συνεχίσει επιτυχώς την συνεργατική διαδικασία, έτσι ώστε οι περιορισμοί ποιότητας να ικανοποιούνται στο μέγιστο δυνατό βαθμό. Αυτό απαιτεί ενέργειες όπως την εκτέλεση επιπλέον υπηρεσιών ή την προσαρμογή της υπηρεσίας ώστε να αντισταθμίζει τα αποτελέσματα της εξαίρεσης.
- Το τμήμα **καταγραφής** (logging) καταγράφει όλα τα γεγονότα και τις εξαιρέσεις μαζί με το χειρισμό που εκτελείται. Ο στόχος είναι η χρήση αυτών των δεδομένων για βελτιστοποίηση της διαδικασίας, και συγκεκριμένα την παροχή συστάσεων για χειρισμό εξαιρέσεων ώστε τελικά να παρέχει έξοδο που να καθορίζει νέους κανόνες χειρισμού εξαιρέσεων.

Επιπλέον κομμάτια της αρχιτεκτονικής είναι η αναζήτηση Υπηρεσιών Διαδικτύου για την ανακάλυψη των κατάλληλων παρόχων για ένα υποέργο που θα εκτελεστεί με τη χρήση καταλόγων υπηρεσιών, και το Web-Flow metadata repository που διατηρεί μετα-δεδομένα όπως περιορισμούς και κανόνες για παρακολούθηση ποιότητας και χειρισμό εξαιρέσεων. Πληροφορίες σχετικά με συνεργατικές διαδικασίες και τις χρησιμοποιούμενες διαδικασίες μπορεί να προκύψουν από τη βάση δεδομένων των διαδικασιών.

Το υποσύστημα λογισμικού Web-Flow MaX χρησιμοποιεί μια πολυεπίπεδη προσέγγιση για χειρισμό εξαιρέσεων. Αρχικά ο χειρισμός εξαιρέσεων μπορεί να συμβεί στον ιστοχώρο όπου εκτελείται μια Υπηρεσία Διαδικτύου, είτε από την ίδια την Υπηρεσία Διαδικτύου είτε από το Web-Flow MaX υποσύστημα του συνεργάτη. Για παράδειγμα, μια Υπηρεσία Διαδικτύου που ψάχνει για προσφορές σε ένα συγκεκριμένο επίπεδο τιμών μπορεί να ελέγξει τις προσφορές πριν τις επιστρέψει και να εκτελέσει μια νέα αναζήτηση εάν δεν βρεθούν κατάλληλα αποτελέσματα. Τέτοιοι μηχανισμοί είναι τυπικά αόρατοι για το χρήστη της υπηρεσίας αλλά απαιτούν μια περίπλοκη Υπηρεσία Διαδικτύου η ύπαρξη της οποίας δεν μπορεί πάντα να υποτεθεί. Επομένως, τα επιπλέον επίπεδα χειρισμού εξαιρέσεων συμβαίνουν στη συνεργατική διαδικασία και από το Web-Flow MaX υποσύστημα του ιστοχώρου που καλεί μια απομακρυσμένη Υπηρεσία Διαδικτύου. Ο χειρισμός εξαιρέσεων του Web-Flow MaX είναι αρκετά ανεξάρτητος από τις αντίστοιχες υλοποιήσεις Υπηρεσιών Διαδικτύου και ενεργοποιεί επομένως την επιβολή γενικών πολιτικών χειρισμού εξαιρέσεων. Μια προσέγγιση με κανόνες έχει το πλεονέκτημα ότι ο βασικός ορισμός διαδικασιών είναι καθαρά ανεξάρτητος από τους κανόνες εξαιρέσεων, διευκολύνοντας επομένως την αναγνωσιμότητα και τη συντήρηση και των δύο. Επιπλέον, επιθυμητές ενέργειες δεν προκύπτουν εως ότου συμβεί μια εξαίρεση, και έτσι αλλαγές σε επιχειρησιακές πολιτικές μπορούν να ληφθούν υπόψη χωρίς αλλαγές στον ορισμό των διαδικασιών.

7.2. Περιορισμοί Ποιότητας

Με βάση την ανάλυση των διαφορετικών σεναρίων συνεργατικών επιχειρησιακών διαδικασιών και προηγούμενες ταξινομήσεις [20], ορίζονται στη συνέχεια διάφοροι τύποι περιορισμών ποιότητας των που υποστηρίζονται στο Web-Flow:

1. Οι **περιορισμοί Μετα-δεδομένων** αναφέρονται στις συνθήκες της περιγραφής των Υπηρεσιών Διαδικτύου, όπως στο UDDI των υπηρεσιών ή τα WSDL μετα-δεδομένα. Τέτοιοι περιορισμοί μπορούν να περιορίσουν πολλές διαφορετικές πτυχές, π.χ. ο πάροχος μιας υπηρεσίας (συγκεκριμένες επιχειρήσεις, συγκεκριμένες γεωγραφικές θέσεις...) ή ένα

κόστος που μπορεί να πρέπει να πληρωθεί για τη χρήση μιας υπηρεσίας (περιορισμός κόστους).

2. Οι **περιορισμοί Εκτέλεσης** αναφέρονται στις συνθήκες της φυσικής εκτέλεσης των Υπηρεσιών Διαδικτύου, σε συγκεκριμένα όρια χρόνου απόκρισης ή στο μέγιστο αριθμό επαναπροσπαθειών για αποτυχημένες εκτελέσεις. Οι περιορισμοί χρόνου απόκρισης καθορίζουν το μέγιστο χρόνο αναμονής για μια απάντηση από μια υπηρεσία ως σταθερό σημείο στο χρόνο (ημερομηνία) ή ως χρονικό διάστημα.
3. Οι **περιορισμοί Εισόδου** θέτουν τους όρους των παραμέτρων εισόδου μιας κλήσης Υπηρεσίας Διαδικτύου.
4. Οι **περιορισμοί Αποτελεσμάτων** καθορίζουν τις συνθήκες των αποτελεσμάτων μιας εκτέλεσης Υπηρεσίας Διαδικτύου.

Αυτές οι συνθήκες τυπικά αναφέρονται στα XML έγγραφα που επιστρέφονται από μια υπηρεσία. Μπορούν να χρησιμοποιηθούν για να ελέγξουν εάν ο χρόνος παράδοσης ή η τιμή ενός προϊόντος κυμαίνεται σε αποδεκτά όρια ή εάν οι ρυθμίσεις ένα προϊόντος που επιστρέφεται ταιριάζουν με τις απαιτήσεις του χρήστη.

Οι περιορισμοί Μετα-δεδομένων αναφέρονται σε στατικές πληροφορίες, οι οποίες καθορίζονται όταν καταχωρείται μια υπηρεσία. Αυτοί οι περιορισμοί είναι κυρίως χρήσιμοι για την επιλογή υπηρεσίας, για την ανακάλυψη δηλαδή μιας κατάλληλης υπηρεσίας κατά την εκτέλεση της διαδικασίας ή αφού προκύψει μια εξαίρεση. Τα άλλα είδη περιορισμών είναι πιο δυναμικά από την άποψη ότι αναφέρονται σε πληροφορίες που σχετίζονται με την πραγματική εκτέλεση των Υπηρεσιών Διαδικτύου (όχι μόνο τις προδιαγραφές). Σχετίζονται κυρίως με το δυναμικό χειρισμό εξαιρέσεων. Υπάρχουν αρκετές επιπλέον ιδιότητες που χαρακτηρίζουν τους περιορισμούς ποιότητας και τη χρήση τους:

- **Παρακολούθηση***:* Δυναμικοί ποιοτικοί περιορισμοί (χρόνος απόκρισης, περιορισμοί εισόδου, αποτελεσμάτων) μπορεί είτε να παρακολουθούνται από την πλευρά του πελάτη είτε από την πλευρά του παρόχου μιας υπηρεσίας. Το πρώτο εναλλακτικό είναι τυπικό ενός απλού μοντέλου συνεργασίας που διατηρεί υψηλή αυτονομία στους παρόχους υπηρεσιών. Ένα σύνθετο μοντέλο συνεργασίας μπορεί να υποστηρίζει και τις δύο προσεγγίσεις.
- ***Αυστηρότητα****:* Οι περιορισμοί μπορεί να είναι υποχρεωτικοί είτε απλά επιθυμητοί (υψηλή ή χαμηλή αυστηρότητα). Μια υπηρεσία που παραβιάζει κάποιον περιορισμό του δεύτερου τύπου μπορεί να είναι παρόλα αυτά χρήσιμη.
- ***Εμβέλεια****:* Οι ποιοτικοί περιορισμοί αναφέρονται τυπικά σε μια συγκεκριμένη υπηρεσία ή λειτουργία υπηρεσίας, αλλά μπορεί και να σχετίζονται με συγκεκριμένη διαδικασία ή χρήστη. Επιπλέον, μπορεί να υπάρχουν global περιορισμοί που να εφαρμόζονται σε αρκετές ή όλες τις υπηρεσίες, δηλαδή να επιβάλουν επιχειρησιακούς κανονισμούς σε επιλέξιμους παρόχους υπηρεσιών/συνεργάτες ή να ορίζουν μια προκαθορισμένη τιμή για μέγιστους χρόνους απόκρισης. Επιπλέον, ένας περιορισμός μπορεί να εξαρτάται από το περιεχόμενο.

Στο Web-Flow, έχει οριστεί μια δηλωτική προδιαγραφή με βάση XML για κάθε είδος ποιοτικών περιορισμών. Οι περιορισμοί που καθορίζονται ως λογικά κατηγορήματα σύγκρισης μπορούν να συνδυαστούν με σύνθετες συνθήκες χρησιμοποιώντας τους λογικούς τελεστές AND, OR και NOT. Ο XML κώδικας του πίνακα 16 δείχνει ένα operation-specific περιορισμό αποτελεσμάτων που καθορίζει ότι η τιμή για μια κράτηση δωματίου ξενοδοχείου πρέπει να είναι μικρότερη ή ίση από τη μέγιστη τιμή που καθορίζεται από το χρήστη.

```
<service name="HotelSearchService">
        <operation operationName="hotelReservation">
        <resultConstraint name="RC1"
        scope="HotelSearchService:hotelReservation"
        strictness="high">
        <parameterCondition>
        <lessEqual>
        <leftOperand>
        <xPathQuery>/hotelDetails/maximumPrice</xPathQuery>
        <parameter name="hotelReservationRequest" type="Input"/>
        </leftOperand>
        <rightOperand>
        <xPathQuery> /reservation/reservationDetails/price</xPathQuery>
        <parameter name="hotelReservationResponse" type="Output"/>
        </rightOperand>
        </lessEqual>
        </parameterCondition>
        </resultConstraint>
        </operation>
...
</service>
```

Πίνακας 16: Παράδειγμα περιορισμού αποτελέσματος

Για να ελεγχθεί ο περιορισμός πρέπει το έγγραφο εξόδου της λειτουργίας να συγκριθεί με το έγγραφο εισόδου που χρησιμοποιήθηκε πριν για την κλήση της Υπηρεσίας Διαδικτύου. Επομένως τα δεδομένα του εγγράφου εισόδου πρέπει να αποθηκευτούν μέχρι η υπηρεσία να επιστρέψει μια απάντηση και να αξιολογηθεί ο περιορισμός. Η ετικέτα *xPathQuery* καθορίζει ποιο κομμάτι του εγγράφου που ορίζεται από την ετικέτα *parameter* πρέπει να χρησιμοποιηθεί για την αξιολόγηση. Όλοι οι ποιοτικοί περιορισμοί καθορίζονται στο υποσύστημα ορισμού των περιορισμών και κανόνων του Web-Flow MaX και διατηρούνται στο Web-Flow repository μετα-δεδομένων. Οι ποιοτικοί περιορισμοί υπηρεσιών και λειτουργιών αποθηκεύονται στο Web-Flow extended service description (*.wesd). Οι περιορισμοί διαδικασιών είναι τμήμα της Web-Flow εκτεταμένης περιγραφής διαδικασιών (*.wepd) μιας συνεργατικής διαδικασίας. Οι global περιορισμοί διατηρούνται ξεχωριστά.

7.3. Δυναμικός Χειρισμός Εξαιρέσεων

Ο δυναμικός χειρισμός εξαιρέσεων στο Web-Flow βασίζεται σε μια προσέγγιση με κανόνες χρησιμοποιώντας Event-Condition-Action (ECA) κανόνες. Γεγονότα που μπορεί να καταλήξουν σε εξαιρέσεις περιλαμβάνουν τις κλήσεις Υπηρεσιών Διαδικτύου, την υποδοχή αποτελεσμάτων Υπηρεσιών Διαδικτύου ή μηνυμάτων λάθους, ή r ειδοποιήσεις μέσω ηλεκτρονικού ταχυδρομείου ή τηλεφωνήματος. Επιπλέον γεγονότα μπορεί να είναι και οι ενημερώσεις βάσεων δεδομένων. Τα γεγονότα ανιχνεύονται από τον περιορισμό και την παρακολούθηση γεγονότων του Web-Flow MaX υποσυστήματος το οποίο παρατηρεί όλα τα μηνύματα που στέλνονται και λαμβάνονται από τη μηχανή επεξεργασίας κατά την εκτέλεση της επεξεργασίας. Το μέρος προαιρετικής συνθήκης ενός κανόνα χρησιμοποιείται για να ελέγξει τους διάφορους ποιοτικούς περιορισμούς και να καθορίσει επιπλέον συνθήκες π.χ. στο Web-Flow οι πληροφορίες καταγραφής ή τα μετα-δεδομένα, που πρέπει να διατηρούνται έτσι ώστε ένα γεγονός να καταλήγει σε εξαίρεση. Αυτό μπορεί να είναι χρήσιμο για να εκφράσει περιορισμούς εξαρτώμενους από το περιεχόμενο. Χρησιμοποιείται ένας επεξεργαστής ερωτημάτων μιας

μηχανής βάσης δεδομένων για να ελέγξει τα μετα-δεδομένα, τους περιορισμούς εισόδου και εξόδου. Η παραβίαση του περιορισμού του χρόνου απόκρισης μπορεί να ανιχνευθεί από το υποσύστημα παρακολούθησης εάν δεν ληφθεί απάντηση για μια σύγχρονη κλήση υπηρεσίας μέχρι να λήξει η προθεσμία. Ένας περιορισμός εκτέλεσης που προσδιορίζει ένα μέγιστο αριθμό επαναπροσπαθειών ελέγχεται χρησιμοποιώντας τις πληροφορίες καταγραφής του Web-Flow. Αυτό καθορίζει το χειρισμό μιας εξαίρεσης. Οι πιθανότητες που υπάρχουν περιλαμβάνουν την εξουσιοδότηση του χειρισμού εξαιρέσεων στην αντίστοιχη διαδικασία, την χειροκίνητη αντίδραση, την αναζήτηση για μια εναλλακτική Υπηρεσία Διαδικτύου, την κλήση μιας συγκεκριμένης Υπηρεσίας Διαδικτύου, την ακύρωση της διαδικασίας, και τη δυναμική προσαρμογή της καλούσας διαδικασίας. Στο Web-Flow, ένας προκαθορισμένος κανόνας δημιουργείται για κάθε δυναμικό ποιοτικό περιορισμό καθορίζοντας ότι ο χρήστης πρέπει να ειδοποιηθεί εάν παραβιαστεί ο περιορισμός. Αυτοί οι κανόνες μπορεί να επεξεργαστούν χειροκίνητα για να καθορίσουν εναλλακτικές αυτόματες αντιδράσεις.

EVENT	λήψη μηνύματος εξόδου ο
CONDITION	παραβίαση του περιορισμού RC1 και #επαναλήψεων>2
ACTION	callService(service2.B)
PRIORITY	3

Πίνακας 17: Παράδειγμα κανόνα χειρισμού εξαιρέσεων

Στον πίνακα 17 φαίνεται ένα παράδειγμα κανόνα που καθορίζει ότι μια εναλλακτική λειτουργία υπηρεσίας (η λειτουργία Β της υπηρεσίας 2) θα πρέπει να καλείται εάν ο περιορισμός αποτελέσματος που ονομάζεται RC1 παραβιάζεται και η υπηρεσία έχει ήδη κληθεί δύο φορές. Οι κανόνες προτεραιότητας χρησιμοποιούνται για να καθορίσουν τη σειρά της εκτέλεσης κανόνων ανά γεγονός.(1 = υψηλότερη προτεραιότητα).

Η επεξεργασία γεγονότων περιλαμβάνει την καταγραφή του γεγονότος και τον έλεγχο των σχετικών κανόνων. Αυτό εκτελείται αμέσως μόλις ανιχνευθεί ένα γεγονός από το υποσύστημα περιορισμών και παρακολούθησης γεγονότων του Web-Flow MaX. Εάν ένας κανόνας βρεθεί για ένα γεγονός καθορίζοντας μια αυτόματα εκτελέσιμη πράξη, το Web-Flow θα την εκτελέσει. Εάν η εκτέλεση αποτύχει απαιτείται μια χειροκίνητη αντίδραση. Για να μειωθεί η εξάρτηση στον χειροκίνητο χειρισμό εξαιρέσεων θα πρέπει να αξιολογηθούν όλα τα καταγεγραμμένα βήματα χειρισμού εξαιρέσεων. Όταν συμβεί μια εξαίρεση ζητώντας χειροκίνητη αντίδραση, οι καταγεγραμμένες πληροφορίες θα πρέπει να αναζητηθούν για να παρουσιάσουν τις ενέργειες που λήφθηκαν σε παρόμοιες περιπτώσεις εξαιρέσεων στο παρελθόν για το χρήστη. Μια δυσκολία με αυτό το βήμα είναι να βρεθούν κατάλληλες μετρικές για την ομοιότητα των εξαιρέσεων.

7.4. Συμπεράσματα για το δυναμικό χειρισμό εξαιρέσεων

Η υποστήριξη της υψηλής ποιότητας εκτέλεσης συνεργατικών διαδικασιών με βάση Υπηρεσίες Διαδικτύου απαιτεί το δυναμικό χειρισμό εξαιρέσεων. Η προσέγγιση του Web-Flow που παρουσιάσθηκε σε αυτό το κεφάλαιο βασίζεται στον καθορισμό μιας ποικιλίας από ποιοτικούς περιορισμούς για ετερογενείς και αυτόνομες Υπηρεσίες Διαδικτύου. Ένα υποσύστημα με κανόνες για παρακολούθηση και χειρισμό εξαιρέσεων αντιμετωπίζει αυτόματα πολλές εξαιρέσεις όπως η παραβίαση ποιοτικών περιορισμών ή λάθη υπηρεσιών.

8. Αποτελεσματική Πρόσβαση σε Υπηρεσίες Διαδικτύου

Οι Υπηρεσίες Διαδικτύου μπορεί να είναι συνδεδεμένες με συγκεκριμένες πηγές δεδομένων, ή μπορεί να είναι αρκετά γενικές (generic) ώστε να λειτουργούν με ένα ευρύ σύνολο από πηγές. Η χρήση των Υπηρεσιών Διαδικτύου αποτελείται γενικά από την κλήση λειτουργιών μέσω της αποστολής και λήψης μηνυμάτων. Παρόλα αυτά, μια σύνθετη εφαρμογή, όπως ένα ταξιδιωτικό πακέτο που προσπελαύνει διαφορετικές Υπηρεσίες Διαδικτύου, χρειάζεται έναν ολοκληρωμένο και αποτελεσματικό τρόπο να διαχειριστεί και να παραδώσει τις λειτουργίες των Υπηρεσιών Διαδικτύου. Μια περιεκτική middleware πλατφόρμα θα απαιτούσε τεχνικές όπως η περιγραφή υπηρεσιών, η ανακάλυψη, η σύνθεση, η παρακολούθηση, η ασφάλεια, και η μυστικότητα, ώστε να βοηθήσει στη διαχείριση αυτόνομων και ετερογενών Υπηρεσιών Διαδικτύου.

Στη συνέχεια παρουσιάζεται μια νέα υποδομή που προσφέρει σύνθετες και βελτιστοποιημένες δυνατότητες ερωτημάτων για Υπηρεσίες Διαδικτύου. Εν συντομία, οι χρήστες υποβάλλουν δηλωτικά ερωτήματα που επιλύει η υποδομή μέσω συνδυασμένων κλήσεων διαφορετικών λειτουργιών Υπηρεσιών Διαδικτύου. Τα ερωτήματα (Queries) στοχεύουν σε Υπηρεσίες Διαδικτύου και στη ροή πληροφοριών που ανταλλάσουν οι υπηρεσίες κατά τη διάρκεια της κλήσης των λειτουργιών τους. Το προτεινόμενο πρότυπο ερωτημάτων διευκολύνει τη διατύπωση και την υποβολή ερωτημάτων και τον μετασχηματισμό τους σε πραγματικές κλήσεις λειτουργιών Υπηρεσιών Διαδικτύου. Το πρότυπο βελτιστοποίησης χρησιμοποιεί παραμέτρους ποιότητας Υπηρεσίας Διαδικτύου (QoWS) για να καλύψει τις απαιτήσεις των χρηστών [21].

8.1. Σενάριο: Προγραμματισμός ενός ταξιδιού

Ένα συχνό παράδειγμα χρήσης Υπηρεσιών Διαδικτύου είναι αυτό ενός τουρίστα ο οποίος ψάχνει ένα καλό ταξιδιωτικό πακέτο. Στη συνηθισμένη περίπτωση, ο τουρίστας θα έψαχνε τις ιστοσελίδες αερογραμμών χρησιμοποιώντας μια μηχανή αναζήτησης, η οποία θα επέστρεφε πολυάριθμα αποτελέσματα. Αφού θα επέλεγε κάποιες ιστοσελίδες και εκτελώντας ερωτήματα σε αυτές, θα επέλεγε κάποια αεροπορική εταιρία που θα φαινόταν να παρέχει την πιο συμφέρουσα λύση. Στη συνέχεια, θα χρησιμοποιούσε ένα κατάλογο Διαδικτύου με πληροφορίες για διαμονή. Έπειτα από μερικά ερωτήματα, θα αποφάσιζε σε ποια ξενοδοχεία να μείνει κατά τη διάρκεια του ταξιδιού του. Στο τρίτο βήμα, θα αναζητούσε κάποιους ιστοτόπους ενοικίασης αυτοκινήτων μέχρι να επιλέξει και πάλι το συμφερότερο. Τέλος, θα χρησιμοποιούσε τις σημειώσεις του ώστε να υπολογίσει τα συνολικά έξοδα του ταξιδιού και να αποφασίσει τις αγορές του. Επιστρέφοντας στους ιστοτόπους, θα έκανε τις κρατήσεις για τα αεροπορικά εισιτήρια, την ενοικίαση αυτοκινήτου και το ξενοδοχείο.

Αυτό το σενάριο υπογραμμίζει τις δυσκολίες που συναντούν οι χρήστες για να διεκπεραιώσουν πολύπλοκα ερωτήματα στο Διαδίκτυο. Επιπλέον του ότι αντιμετωπίζει μια χρονοβόρα και δύσκολη διαδικασία, ο χρήστης πιθανότατα θα χάσει τις καλύτερες προσφορές. Στην ουσία, ο χρήστης χρειάζεται ένα σύστημα που να του επιτρέπει να υποβάλλει αποτελεσματικά ερωτήματα σε Υπηρεσίες Διαδικτύου. Ο ταξιδιώτης σε αυτή την περίπτωση θα χρειαζόταν να εκφράσει απλά τις ανάγκες του μέσω δηλωτικών ερωτημάτων πάνω από καλά ορισμένες διεπαφές. Στη συνέχεια αναλύεται μια γενική προσέγγιση για τη βελτιστοποιημένη αναζήτηση σε Υπηρεσίες Διαδικτύου.

8.2. Ερωτήματα σε Υπηρεσίες Διαδικτύου

Η πρωταρχική χρήση Υπηρεσιών Διαδικτύου ήταν έως τώρα η κλήση διαδικασιών με την αποστολή και λήψη μηνυμάτων. Αν και αυτή η χρήση καλύπτει τις απαιτήσεις των απλών εφαρμογών, οι σύνθετες εφαρμογές που προσπελαύνουν διαφορετικές Υπηρεσίες Διαδικτύου χρειάζονται έναν ενοποιημένο τρόπο για τη διαχείριση και την παράδοση λειτουργιών τους. Επιπλέον, καθώς όλο και περισσότερες επιχειρήσεις αρχίζουν να αναπτύσσουν Υπηρεσίες

Διαδικτύου, πολλές από αυτές θα ανταγωνίζονται προσφέροντας παρόμοιες υπηρεσίες (π.χ. διαφορετικές αερογραμμές που προσφέρουν την ίδια σύνδεση μεταξύ δύο πόλεων).

Εντούτοις, οι Υπηρεσίες Διαδικτύου θα διαφέρουν στον τρόπο που προσφέρουν αυτές τις λειτουργίες (απαιτώντας για παράδειγμα διαφορετικές παραμέτρους εισόδου) και στις συνθήκες χρήσης τους (διαφορετικά επίπεδα QoWS). Επιπλέον, η ικανοποίηση των αιτημάτων των χρηστών μπορεί να μην απαιτεί την επιστροφή ακριβούς απάντησης, καθώς πολλοί χρήστες μπορεί να ικανοποιηθούν από εναλλακτικές ή μερικές απαντήσεις. Σαν σημαντικό βήμα στην εξέταση αυτών των προκλήσεων, προτείνεται μια νέα προσέγγιση στην υποβολή ερωτημάτων σε Υπηρεσίες Διαδικτύου και τη ροή πληροφορίας κατά την κλήση των λειτουργιών τους. Ανάλογα με το ερώτημα, η απάντηση του ίσως οδηγεί στην κλήση πολλών λειτουργιών Υπηρεσιών Διαδικτύου και τον συνδυασμό των εξόδων τους. Η προτεινόμενη μέθοδος δίνει έμφαση στη βελτιστοποίηση της query-resolution διαδικασίας, επιτρέποντας μερικές (partial) απαντήσεις.

8.2.1. Ένα μοντέλο ερωτημάτων για Υπηρεσίες Διαδικτύου

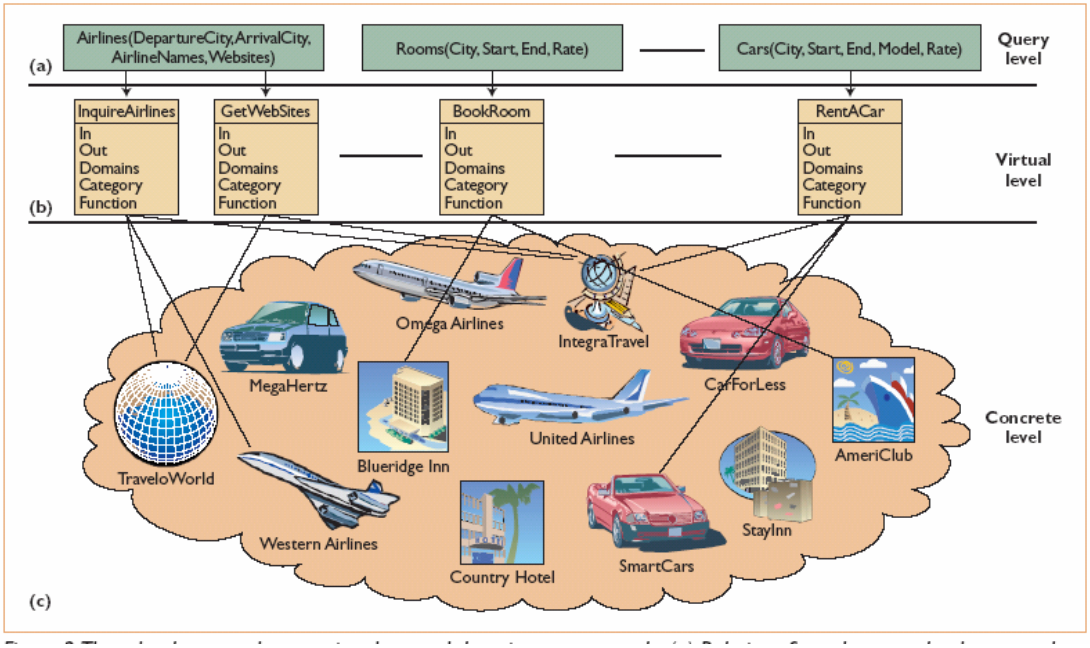
Για τη διευκόλυνση της αλληλεπίδρασης μεταξύ χρηστών και Υπηρεσιών Διαδικτύου, πρέπει οι Υπηρεσίες Διαδικτύου σε μια δεδομένη εφαρμογή να αναπαρασταθούν με γενικό τρόπο. Ορίζεται ένα σύνολο από domain-specific λειτουργίες που καλούνται εικονικές λειτουργίες (ονομάζονται έτσι επειδή δεν ανήκουν σε κάποια πραγματική Υπηρεσία Διαδικτύου) που θα χρησιμοποιηθεί σε ένα μοντέλο ερωτημάτων τριών επιπέδων:

• Στην κορυφή, το **επίπεδο ερωτημάτων** αποτελεί ένα σύνολο από σχέσεις που δίνουν στους χρήστες μια διεπαφή για τη διατύπωση και την υποβολή δηλωτικών ερωτημάτων που απευθύνονται στις Υπηρεσίες Διαδικτύου. Διαφορετικοί κανόνες αντιστοίχησης ορίζουν διαφορετικά σύνολα από σχέσεις πάνω στις εικονικές λειτουργίες.

• Το **εικονικό επίπεδο** αποτελείται από λειτουργίες τύπου Υπηρεσιών Διαδικτύου που προσφέρει συνήθως μια συγκεκριμένη περιοχή εφαρμογών (π.χ. εύρεση τιμών δωματίων ξενοδοχείων και έλεγχος διαθεσιμότητας πτήσης).

• Το **concrete επίπεδο** αναπαριστά το χώρο των Υπηρεσιών Διαδικτύου που προσφέρεται στο Διαδίκτυο, δηλαδή τους πιθανούς υποψήφιους για την απάντηση των ερωτημάτων. Επειδή οι Υπηρεσίες Διαδικτύου είναι a priori άγνωστες, το μοντέλο ερωτημάτων πρέπει να τις ανακαλύψει και να τις αντιστοιχήσει με τις εικονικές λειτουργίες που εμφανίζονται στο ερώτημα.

Στο παράδειγμα του τουρίστα, θα μπορούσε να βρει διαθέσιμα δωμάτια και ενοικίαση αυτοκινήτου σε μια δεδομένη πόλη και σε μια δεδομένη χρονική περίοδο απλά διατυπώνοντας ένα ερώτημα που χρησιμοποιεί τις σχέσεις Rooms(City,Start, End, Rate) και Cars(City, Start,End, Model, Rate). Το μοντέλο ερωτημάτων τότε θα ταίριαζε αυτές τις δύο σχέσεις στις αντίστοιχες εικονικές λειτουργίες BookRoom και RentACar. Η διαδικασία ταιριάσματος στη συνέχεια θα αντιστοίχιζε αυτές τις εικονικές λειτουργίες σε διάφορες διαδικασίες από το concrete επίπεδο.



Σχήμα 20: Σχήμα ερωτημάτων τριών επιπέδων.

8.2.1.1. Από τις Σχέσεις στις εικονικές διαδικασίες

Οι σχέσεις στο επίπεδο ερωτημάτων καθορίζουν μια συγκεκριμένη άποψη της περιοχής εφαρμογών. Παρουσιάζονται ως συνδετικά ερωτήματα πάνω σε εικονικές λειτουργίες, αναπαριστώντας επομένως τις εικονικές λειτουργίες σαν να ήταν σχέσεις. Ακριβέστερα, έστω R το σύνολο των σχέσεων που ορίζει το επίπεδο ερωτημάτων και VOP το σύνολο των εικονικών λειτουργιών. Για μια σχέση Ri ε R,

$$R_i(x_1, x_2, \ldots, x_n) :- \bigwedge Vop_j(y_{j_1}, y_{j_2}, \ldots, y_{j_m})$$

όπου x_i είναι οι ιδιότητες του R_i, το Vop_j ε VOP, και y_{ji} είναι οι αντίστοιχες μεταβλητές εισόδου και εξόδου της διαδικασίας. Αυτός ο ορισμός σημαίνει ότι για να πάρουμε τις πλειάδες R_i , πρέπει να καλέσουμε τις διαφορετικές λειτουργίες Vop_j. Δεν επιβάλει καμία σειρά ή όριο concurrency ανάμεσα σε αυτές τις λειτουργίες. Ένα παράδειγμα του κανόνα αντιστοίχησης είναι:

Airlines (DepartureCity, ArrivalCity,AirlineNames, WebSites) **:-**
InquireAirlines (DepartureCity,ArrivalCity,AirlineNames),
GetWebSites (AirlineNames, WebSites)

Ο παραπάνω κανόνας αντιστοίχησης ορίζει τη σχέση Airlines μέσω δύο εικονικών λειτουργιών: η InquireAirlines επιστρέφει τον κατάλογο αερογραμμών που λειτουργούν μεταξύ δύο πόλεων, και η GetWebSites επιστρέφει τους ιστοτόπους για έναν κατάλογο αερογραμμών. Η υποδομή ερωτημάτων μπορεί να ανακτήσει τις πλειάδες της Airlines καλώντας τη λειτουργία InquireAirlines και έπειτα την GetWebSites. Οι χρήστες μπορούν να χρησιμοποιήσουν απευθείας τις εικονικές λειτουργίες για να προσπελάσουν τις Υπηρεσίες Διαδικτύου, αλλά η χρήση των σχέσεων έχει δύο πλεονεκτήματα: Καταρχήν, επιτρέπει στους χρήστες να διατυπώσουν και να υποβάλλουν ερωτήματα τύπου βάσεων δεδομένων με φυσικό τρόπο. Επιπλέον, παρέχει μια άποψη προσανατολισμένη σε ένα συγκεκριμένο σύνολο χρηστών που ενδιαφέρονται για ένα συγκεκριμένο κομμάτι της υπηρεσίας.

8.2.1.2. Αναπαράσταση Εικονικών Λειτουργιών

Οι εικονικές λειτουργίες αναπαριστούν τις λειτουργίες που παρέχει ένα συγκεκριμένο πεδίο εφαρμογών. Για οποιαδήποτε εικονική λειτουργία εμφανίζεται σε ένα ερώτημα, πρέπει να εντοπιστούν οι σχετικές λειτουργίες Υπηρεσιών Διαδικτύου. Αυτό απαιτεί την σημασιολογική περιγραφή όπως και τις συντακτικές ιδιότητες. Έστω ότι οι επιχειρησιακοί συνεργάτες θα συμφωνούσαν σε μια κοινή οντολογία για την περιγραφή concrete Υπηρεσιών Διαδικτύου και εικονικών λειτουργιών.Κάθε λειτουργία, εικονική είτε concrete, περιγράφεται σημασιολογικά μέσω της λειτουργίας (function) και της κατηγορίας (category).

Η λειτουργία περιλαμβάνει τις εξής ιδιότητες:

•**Λειτουργικότητα** (Functionality): αναπαριστά την επιχειρησιακή λειτουργικότητα που παρέχεται από την λειτουργία (booking and listing).

•**Συνώνυμα** (Synonyms): περιέχει έναν κατάλογο εναλλακτικών ονομάτων για τη λειτουργία (π.χ. reservation είναι ένα συνώνυμο του booking).

Η κατηγορία περιλαμβάνει τις εξής ιδιότητες:

•**Περιοχή** (Domain): δίνει τον τομέα ενδιαφέροντος της λειτουργίας (π.χ flight, accommodation, and tour).

•**Συνώνυμα** (Synonyms): μοιάζουν με αυτά που ορίζονται από το σκοπό της λειτουργίας.

Για να κληθεί μια υπηρεσία, πρέπει να δοθούν οι τιμές για τις μεταβλητές εισόδου της. Στη διατύπωση των ερωτημάτων, οι χρήστες είναι ελεύθεροι να ορίσουν οποιοδήποτε τύπο συνθηκών στις μεταβλητές. Αυτό μπορεί να οδηγήσει σε ένα σενάριο στο οποίο το σύστημα δεν μπορεί καλέσει λειτουργίες επειδή μερικές μεταβλητές εισόδου έχουν ελλιπείς τιμές. Εάν όλες οι περιγραφές των εικονικών λειτουργιών διευκρίνιζαν όλες τις πιθανές τιμές των μεταβλητών εισόδου, αυτό θα έλυνε το πρόβλημα (αν και αυτό δεν είναι πάντα δυνατό για όλες τις μεταβλητές εισόδου). Το σύστημα θα μπορούσε τότε να επεκτείνει μια μεταβλητή εισόδου σε όλες τις πιθανές τιμές που ικανοποιούν οποιαδήποτε συνθήκες στο ερώτημα. Η ακόλουθη πεντάδα αναπαριστά τυπικά κάθε εικονική λειτουργία:

Vop = (In, Out, Domains, Category, Function) όπου In είναι το σύνολο από μεταβλητές εισόδου, Out είναι το σύνολο από μεταβλητές εξόδου, Domains είναι το σύνολο του ζεύγους (x, range), όπου x είναι μια αριθμήσημη μεταβλητή που εμφανίζεται στο In, και range είναι το σύνολο όλων των πιθανών τιμών του x, η Category περιγράφει το πεδίο ενδιαφέροντος και η Function περιγράφει την επιχειρησιακή λειτουργία. Το ακόλουθο είναι ένα παράδειγμα μιας εικονικής λειτουργίας με όλες τις ιδιότητες:

Fares = (In, Out, Domains, Category, Function) όπου:

- In = (DepartureCity, ArrivalCity, DepartureDate),
- Out = (DepartureDate, ArrivalDate, ArrivalTime, Price),
- Domains= {(DepartureDate, [all dates between today's date and November 30, 2004])},
- Category = {Flight,Trip}, και
- Function ={Listing, Fares}.

8.2.1.3. Πολυεπίπεδη Αντιστοίχηση των Εικονικών Λειτουργιών

Δεδομένου ότι οι πάροχοι αρχίζουν να ανταγωνίζονται στις παροχές των Υπηρεσιών Διαδικτύου, θα εμφανιστούν διαφορές στην απαιτούμενη είσοδο, την έξοδο που επιστρέφεται, την QoWS, κλπ. Παραδείγματος χάριν, οι λεπτομέρειες μιας υπηρεσίας πρόβλεψης καιρού, και το κόστος για την πρόσβαση στην υπηρεσία μπορεί να ποικίλουν ανάλογα με τον πάροοχο. Αυτό σημαίνει ότι δε θα είναι πάντα δυνατό να βρεθεί μια ακριβής αντιστοιχία για μια δεδομένη εικονική λειτουργία. Αντί να βρίσκονται πάντα concrete λειτουργίες που ταιριάζουν απόλυτα με τις εικονικές λειτουργίες που εμφανίζονται σε ένα ερώτημα, ίσως οι ανάγκες των χρηστών να

ικανοποιούνταν από ένα πιο ευέλικτο σχήμα ταιριάσματος, που θα επέτρεπε στις ιδιότητες των εικονικών και concrete λειτουργιών να αντιστοιχίζονται.

Ορίζεται μια συνάρτηση similar για να ελέγξει εάν δυο ιδιότητες που εμφανίζονται σε δυο λειτουργίες είναι ίδιες: similar(x, y) είναι αληθές εάν τα x και y αντιστοιχούν στις ίδιες έννοιες όσον αφορά την κοινή οντολογία που καθορίζεται στην περιοχή εφαρμογών. Για οποιεσδήποτε δυο λειτουργίες op1 και op2,

- In($op1$) = In($op2$) εάν In($op1$) και In($op2$) έχουν τον ίδιο αριθμό μεταβλητών και
- $\forall x \in \text{In}(op_1)$ (resp. $\text{In}(op_2)$), $\exists y \in \text{In}(op_2)$ (resp. $\text{In}(op_1)$) | $\text{similar}(x, y)$ είναι αληθές

Ορίζεται Out($op1$) = Out($op2$) ανάλογα. Επίσης ορίζεται In($op1$) C In($op2$) και Out($op1$) C Out($op2$) με ανάλογο τρόπο, όπου το πρώτο σύνολο είναι υποσύνολο του δεύτερου.

Έστω ότι οι *Vop* και *Concop* είναι εικονικές και concrete λειτουργίες αντίστοιχα. Καθορίζονται τέσσερα διαφορετικά επίπεδα ταιριάσματος με βάσει τον τρόπο που συγκρίνονται οι ιδιότητες των εικονικών και των concrete λειτουργιών:

- *ακριβής αντιστοιχία,*
- *επικαλυπτόμενη αντιστοιχία,*
- *μερική αντιστοιχία,* και
- *μερική και επικαλυπτόμενη αντιστοιχία..*

Στην ακριβής αντιστοιχία, η concrete λειτουργία ταιριάζει με την εικονική λειτουργία σε όλες τις ιδιότητες. Δύο διαδικασίες ταιριάζουν ακριβώς εάν έχουν τις ίδιες μεταβλητές εισόδου και εξόδου, όπως επίσης και την ίδια κατηγορία και λειτουργία .

Μια επικαλυπτόμενη αντιστοιχία σχετίζεται με μια λειτουργία που παρέχει παραπλήσιες λειτουργίες με την εικονική λειτουργία. Δύο λειτουργίες επικαλύπτονται εάν έχουν τις ίδιες μεταβλητές εισόδου και εξόδου, και οι κατηγορία και λειτουργία τους επικαλύπτονται. Έστω μια εικονική λειτουργία

Fares = (In, Out, Domains, Category, Function) όπου:

- In = (DepartureCity, ArrivalCity, DepartureDate),
- Out = (DepartureDate, ArrivalDate, ArrivalTime, Price),
- Domains= {(DepartureDate, [all dates between today's date and November 30, 2004])},
- Category = {Flight, Trip}, και
- Function ={Listing, Fares}.

Επίσης μια concrete λειτουργία AirfareTickets = (In, Out, Category,Function) όπου:

- In = (DepartureCity, ArrivalCity,DepartureDate),
- Out = (DepartureDate,
- ArrivalDate, ArrivalTime, Price),
- Category= {Flight, Trip, Tour, Tourism}, και
- Function = {Listing, Fares, Quotes}, που παρέχει μια επικαλυπτόμενη αντιστοιχία.
-

Μια μερική αντιστοιχία προκύπτει στην περίπτωση όπου οι ιδιότητες εισόδου και εξόδου των δύο λειτουργιών δε συμπίπτουν. Δύο διαδικασίες η Vop και η Concop ταιριάζουν μερικώς εάν έχουν την ίδια Category και Function, και Out(Concop) $\subseteq$ Out (Vop) είτε In (Concop) $\subseteq$ In(Vop). Ένα παράδειγμα της πρώτης σχέσης υποσυνόλου είναι μια λειτουργία που δεν επιστρέφει όλα τα χαρακτηριστικά εξόδου που περιμένει η εικονική λειτουργία. Ένα παράδειγμα της δεύτερης σχέσης υποσυνόλου είναι δύο διαδικασίες fare1(<Travel Information>, IsStudent, Cost) και fare2(<Travel Information>, Cost).

Και οι δύο διαδικασίες επιστρέφουν τιμές για ένα ταξίδι που διευκρινίζεται από το <Travel Information>, αλλά η δεύτερη δε λαμβάνει υπόψη εάν ο αιτών είναι σπουδαστής, γεγονός που μπορεί να χαμηλώσει την τιμή.

Μια μερική και επικαλυπτόμενη αντιστοιχία. συνδυάζει το μερικό και επικαλυπτόμενο ταίριασμα. Δύο διαδικασίες Vop και Concop ταιράζουν μερικώς και επικαλυπτόμενα εάν Out(Concop)$\subseteq$ Out (Vop) ή In (Concop) $\subseteq$In(Vop) και οι ιδιότητες τους Category και Function επικαλύπτονται.

Ορίζεται ένας βαθμός ταιριάσματος (matching degree) σε κάθε επίπεδο που ποσοτικοποιεί την ακρίβεια του ταιριάσματος. Ο βαθμός ταιριάσματος επηρεάζει άμεσα την ποιότητα των αποτελεσμάτων του ερωτήματος.

8.2.1.4. Βελτιστοποίηση Ερωτημάτων σε Υπηρεσίες Διαδικτύου

Αρκετά σχέδια εκτέλεσης υπηρεσιών που χρησιμοποιούν διαφορετικές Υπηρεσίες Διαδικτύου μπορεί ενδεχομένως να επιλύουν την ίδια ερώτηση. Κατά συνέπεια πρέπει να τεθούν τα κατάλληλα κριτήρια για την επιλογή των καλύτερων σχεδίων εκτέλεσης. Ένα κύριο χαρακτηριστικό στη διάκριση μεταξύ ανταγωνιστικών Υπηρεσιών Διαδικτύου είναι η QoWS, η οποία περιλαμβάνει διάφορες ποσοτικές και ποιοτικές παραμέτρους που μετρούν πόσο καλά παρέχει η Υπηρεσία Διαδικτύου τις λειτουργίες της. Ο στόχος της διαδικασίας βελτιστοποίησης είναι η μεγιστοποίηση (ή ελαχιστοποίηση) της τιμής καθεμίας από τις εξής QoWS παραμέτρους:

•Το **Latency** αναπαριστά το μέσο χρόνο στον οποίο μια διαδικασία επιστρέφει αποτελέσματα από τη στιγμή που κλήθηκε.

•Το **Κόστος** είναι μονάδα του κόστους που ένας καταναλωτής μιας Υπηρεσίας Διαδικτύου πρέπει να πληρώσει για να καλέσει τις διαδικασίες της.

•Η **Διαθεσιμότητα** αναπαριστά την πιθανότητα μια Υπηρεσία Διαδικτύου είναι διαθέσιμη. Οι μεγάλες τιμές σημαίνουν υψηλή διαθεσιμότητα, και οι μικρές τιμές δείχνουν χαμηλή διαθεσιμότητα.

Το επιθυμητό είναι να ελαχιστοποιηθούν οι αρνητικές παράμετροι (latency και κόστος) και να μεγιστοποιηθούν οι θετικές παράμετροι (διαθεσιμότητα). Για την κανονικοποίηση των QoWS παραμέτρων, οι θετικές παράμετροι κυμαίνονται μεταξύ 0 και 1, και οι αρνητικές παράμετροι είναι μεγαλύτερες ίσες με 1.

8.2.1.5. Βαθμολογήσεις Υπηρεσιών Διαδικτύου

Στη διάρκεια ζωής μιας Υπηρεσίας Διαδικτύου μπορεί να προκύψουν διάφορες διακυμάνσεις με αποτέλεσμα η υπηρεσία να μην ικανοποιεί πάντα τις διαφημιζόμενες QoWS παραμέτρους της. Γενικά, οι περισσότεροι χρήστες μπορούν να δεχτούν μικρές διαφορές μεταξύ των διαφημισμένων και παρεχόμενων τιμών. Εντούτοις, οι μεγάλες διαφορές δείχνουν ότι η Υπηρεσίας Διαδικτύου υφίσταται μια υποβάθμιση απόδοσης στην παράδοση των λειτουργιών της. Για αυτό, το προτεινόμενο σύστημα ελέγχει τις QoWS παραμέτρους για τις κληθείσες Υπηρεσίες Διαδικτύου (μετρώντας τις διακυμάνσεις των QoWS παραμέτρων και παρέχοντας μια αξιολόγηση ή εκτίμηση για την υπηρεσία). Τέτοιες αξιολογήσεις διαδραματίζουν σημαντικό ρόλο στη διαδικασία βελτιστοποίησης. Κάθε φορά που ένας χρήστης επιλέγει και καλεί μια Υπηρεσία Διαδικτύου, μετράται η διαφορά μεταξύ της υποσχόμενης και παρεχόμενης QoWS. Αυτή η διαφορά που καλείται QoWS απόσταση είναι το άθροισμα των διαφορών για όλες τις QoWS παραμέτρους. Ακριβέστερα, αν pQi (αντίστοιχα dQi) ($1 \leq i \leq n$) είναι η τιμή της υποσχόμενης QoWS (αντίστοιχα. παραδοθείσας QoWS) για την QoWSi. Pos και neg είναι το σύνολο των QoWS παραμέτρων για μεγιστοποίηση και ελαχιστοποίηση αντίστοιχα. QoWSdist είναι η QoWS απόσταση που υπολογίζεται ως εξής:

$$QoWSdist = \sum_{QoWS_i \in pos} (dQ_i - pQ_i) + \sum_{QoWS_i \in neg} \left(\frac{1}{dQ_i} - \frac{1}{pQ_i} \right)$$

Οι Υπηρεσίες Διαδικτύου αρχικά λαμβάνουν την υψηλότερη βαθμολογία. Εάν μια Υπηρεσία Διαδικτύου έχει μια QoWS απόσταση κάτω από ένα ορισμένο κατώτατο αρνητικό όριο, τότε η βαθμολογία της μειώνεται. Αντίθετα, εάν η QoWS απόσταση έχει τιμή πάνω από ένα ορισμένο θετικό κατώτατο όριο, η βαθμολογία της αυξάνεται. Μια απλή κατάταξη των Υπηρεσιών Διαδικτύου θα χρησιμοποιούσε όλες τις QoWS διακυμάνσεις σε έναν ενιαίο τύπο. Εναλλακτικά, θα μπορούσε να χρησιμοποιηθεί η QoWS κάθε υπηρεσίας ξεχωριστά αναθέτοντας σε κάθε παράμετρο το δικό της rating. Θα μπορούσε τότε να χρησιμοποιηθεί το rating κάθε παραμέτρου για να ζυγίσει την αντίστοιχη QoWS στην συνάρτηση ή να προστεθούν όλα τα ratings και να τα χρησιμοποιηθούν για να ζυγίσουν όλο τον τύπο.

Αρκετά rating schemes υπάρχουν ήδη στο Διαδίκτυο, όπως τα συστήματα που χρησιμοποιούν το Epinions.com, το Bizrate.com, και το eBay. Αυτά στηρίζονται κυρίως στην ανατροφοδότηση των χρηστών, όπου οι καταναλωτές αξιολογούν προϊόντα, υπηρεσίες, και προμηθευτές με βάση την προσωπική τους εμπειρία. Αυτά τα συστήματα αξιολόγησης δεν πραγματοποιούν κανένα έλεγχο σε Υπηρεσίες Διαδικτύου και κλήσεις διαδικασιών, όπως θα έκανε η προτεινόμενη υποδομή. Παρόλα αυτά, θα μπορούσαν να χρησιμοποιηθούν αυτά τα συστήματα για την τελειοποίηση των ratings του προτεινόμενου συστήματος. Για παράδειγμα, ένας. Πράκτορας παρακολούθησης θα μπορούσε να επικοινωνεί μαζί τους περιοδικά για να ενημερώνει τα δικά του ratings.

8.2.1.6. Objective Συνάρτηση

Το εάν οι Υπηρεσίες Διαδικτύου επιτυγχάνουν να παραδώσουν τις λειτουργίες τους εξαρτάται κυρίως από την QoWS τους. Κατά συνέπεια, ο στόχος της διαδικασίας βελτιστοποίησης είναι να προσδιορίσει και να εφαρμόσει Υπηρεσίες Διαδικτύου με τις καλύτερες τιμές για αυτές τις QoWS παραμέτρους. Δύο άλλες παράμετροι, rating (rating) και βαθμός ταιριάσματος (md), βελτιώνουν τον ορισμό της επιτυχίας. Οποιαδήποτε λειτουργία Υπηρεσίας Διαδικτύου που καλείται να απαντήσει σε ένα ερώτημα μπορεί να χαρακτηριστεί με βάση τρία κριτήρια: τις υποσχόμενες QoWS παραμέτρους, το **rating**, και το **md**.

Υπάρχουν διάφορες προσεγγίσεις για τον υπολογισμό της γενικής ποιότητας μιας λειτουργίας βάσει των QoWS παραμέτρων. Η μέθοδος Simple Additive Weighting, που χρησιμοποιείται ευρέος στη λήψη αποφάσεων περιλαμβάνει τρία βασικά βήματα:

1. Κλιμάκωση των διαφορετικών QoWS παραμέτρων ώστε να είναι συγκρίσιμες
2. Εφαρμογή παρεχόμενων από χρήστη βαρών για κάθε παράμετρο, εάν καθορίζεται ως τμήμα της ερώτησης.
3. Πρόσθεση των ζυγισμένων και κλιμακωμένων QoWS παραμέτρων για κάθε λειτουργία.

Η ποιότητα λειτουργίας είναι το άθροισμα που αποκτάται στο τελευταίο βήμα. Ορίζεται μια *objective συνάρτηση F* που πρέπει να μεγιστοποιηθεί για κάθε concrete λειτουργία *op* που λαμβάνει μέρος στην επίλυση του ερωτήματος:

$$F(op) = rating(op) * md(op) * (P_{neg} + P_{pos})$$

όπου :

$$P_{neg} = \sum_{neg} \frac{pQ_i^{\max} - pQ_i}{pQ_i^{\max} - pQ_i^{\min}}$$

$$P_{pos} = \sum_{pos} \frac{pQ_i - pQ_i^{\min}}{pQ_i^{\max} - pQ_i^{\min}}$$

όπου *pQi* max είναι η μέγιστη τιμή για την ι-οστή QoWS παράμετρο για όλες τις concrete λειτουργίες που ταιριάζουν στην ίδια εικονική λειτουργία και *pQi* min είναι η ελάχιστη.

8.2.1.7. Βελτιστοποίηση και Επεξεργασία Ερωτημάτων

Οι χρήστες υποβάλλουν συνδυαστικά ερωτήματα (από σχέσεις και συνθήκες) πάνω σε σχέσεις στο επίπεδο ερωτημάτων. Τα ερωτήματα έχουν την ακόλουθη γενική μορφή: $Q(X):-\underset{i}{\wedge} R_i(X_{i_1}),\underset{k}{\wedge} C_k$

όπου Ri είναι σχέσεις στο επίπεδο ερωτημάτων. Τα X and Xi είναι πλειάδες μεταβλητών ενώ το C_k αναπαριστά συνθήκες σε μεταβλητές που εμφανίζονται στο ερώτημα. Η μορφή αυτών των συνθηκών είναι C_k = x op c, όπου x είναι μια μεταβλητή εισόδου ή εξόδου που εμφανίζεται σε οποιοδήποτε X_i, ceίναι μια σταθερά, και $op \in \{=, \neq, <, >, \geq, \leq,\}$. Πολλαπλές εμφανίσεις μιας μεταβλητής εκφράζουν ισότητα.

Το ερώτημα υποβάλλεται σε διάφορες μετατροπές που καταλήγουν σε ένα σχέδιο εκτέλεσης υπηρεσίας, που καθορίζει μια ακολουθία από σύνολα λειτουργιών. Η υποδομή ερωτημάτων μπορεί ταυτόχρονα να καλέσει διαδικασίες μέσα από το ίδιο σύνολο. Η βελτιστοποίηση εμφανίζεται στην Υπηρεσία Διαδικτύου και στα επίπεδα δεδομένων. Το πρώτο αφορά την επιλογή Υπηρεσίας Διαδικτύου και διαδικασιών και το δεύτερο αφορά την σειρά των κλήσεων διαδικασιών, τη δρομολόγηση δεδομένων, και τη διαχείριση λειτουργιών δεδομένων για τη συλλογή αποτελεσμάτων. Η βελτιστοποίηση του επιπέδου δεδομένων πρέπει να εξασφαλίσει ότι η ανακτώμενη σειρά είναι εφικτή: όποτε μια λειτουργία εμφανίζεται στην ακολουθία (πρέπει δηλαδή να κληθεί), όλες οι μεταβλητές εισόδου της πρέπει να έχουν οριστεί.

Έχει επινοηθεί ένας αλγόριθμος που φτιάχνει αποτελεσματικά σχέδια εκτέλεσης υπηρεσίας χρησιμοποιώντας μια προσέγγιση τοπικής επιλογής για βελτιστοποίηση. Ο αλγόριθμος επιλέγει την καλύτερη concrete λειτουργία για κάθε μια εικονική λειτουργία που εμφανίζεται στο ερώτημα (αφού εφαρμόσει τους κανόνες ταιριάσματος). Για να εξασφαλισθεί η δυνατότητα πραγματοποίησης του σχεδίου που προκύπτει ο αλγόριθμος χτίζει επαναληπτικές ακολουθίες από κλήσεις εικονικών λειτουργιών βάση της διαθεσιμότητας των μεταβλητών εισόδου για κάθε λειτουργία. Υποτίθεται ότι οι προμηθευτές περιγράφουν τις Υπηρεσίες Διαδικτύου τους χρησιμοποιώντας WSDL επαυξημένη με σημασιολογικές ιδιότητες. Επίσης τις δημοσιεύουν σε UDDI καταλόγους. Οι πάροχοι μπορούν να διαφημίσουν τις QoWS παραμέτρους χρησιμοποιώντας τα UDDI tModels. Η κύρια ιδέα είναι ότι ο αλγόριθμος χτίζει ακολουθίες από κλήσεις εικονικών υπηρεσιών βάσει της διαθεσιμότητας των μεταβλητών εισόδου. Ο αλγόριθμος έπειτα ανακαλύπτει concrete λειτουργίες, τις ταιριάζει με εικονικές λειτουργίες, και τις προσπελαύνει χρησιμοποιώντας την objective συνάρτηση εξασφαλίζοντας ότι η ακολουθία που προκύπτει είναι ακόμα εφικτή. Ο αλγόριθμος έχει τρεις φάσεις:

1. αρχικοποίηση και ανάπτυξη ερωτημάτων
2. ταξινόμηση εικονικών λειτουργιών, και
3. ανακάλυψη υπηρεσίας και λειτουργίες ταιριάσματος

Ο κύριος στόχος της φάσης αρχικοποίησης και ανάπτυξης ερωτημάτων είναι να συλλέξει όλες τις μεταβλητές που αποκτά ο αλγόριθμος από το ερώτημα είτε απευθείας (συνθήκες ισότητας) είτε χρησιμοποιώντας ranges (συνθήκες ανισότητας) που ορίζονται σε εικονικές λειτουργίες. Ο αλγόριθμος επίσης αρχικοποιεί το σχέδιο εκτέλεσης της υπηρεσίας σε ένα άδειο σύνολο. Στη συνέχεια αναπτύσσει το ερώτημα για να φτάσει σε εικονικές λειτουργίες χρησιμοποιώντας τους διαφορετικούς κανόνες αντιστοίχησης.

Στη φάση ταξινόμησης εικονικών λειτουργιών, ο αλγόριθμος επιλέγει επαναληπτικά τις εικονικές λειτουργίες που μπορεί να καλέσει στην ακολουθία που αναπαριστά το σχέδιο εκτέλεσης. Ο αλγόριθμος επιλέγει καθορίζοντας σε κάθε επανάληψη όλες τις εικονικές

λειτουργίες που έχουν τα χαρακτηριστικά εισόδου τους bound (δηλαδή των οποίων τις μεταβλητές ο αλγόριθμος μπορεί να αντικαταστήσει με διαθέσιμες σταθερές τιμές). Αυτές οι επιλεγμένες εικονικές λειτουργίες θα παρέχουν τελικά νέες bound μεταβλητές μέσω των χαρακτηριστικών εξόδου τους. Αυτές θα επέτρεπαν στον αλγόριθμο να επιλέξει άλλες λειτουργίες στις επόμενες επαναλήψεις. Εάν βρει μια λειτουργία που δεν έχει bound μεταβλητές, τότε το ερώτημα δεν είναι επιλύσιμο. Στο τέλος της φάσης αυτής, το σχέδιο εκτέλεσης υπηρεσίας αφορά τις εικονικές λειτουργίες.

Για παράδειγμα, έστω ένα ερώτημα Q που περιέχει τη σχέση Airlines που ορίστηκε προηγούμενα.

Q(AirlineNames, WebSites) :-Airlines(DepartureCity, ArrivalCity, AirlineNames, WebSites), DepartureCity = “Miami”, ArrivalCity = “Baltimore”

Ο αλγόριθμος αντικαθιστά τη σχέση Airlines με τις εικονικές λειτουργίες InquireAirlines και GetWebSites. Με βάση τις bound μεταβλητές DepartureCity και ArrivalCity, ο αλγόριθμος πρέπει να επιλέξει πρώτα την InquireAirlines. Η διαθεσιμότητα της ιδιότητας εισόδου AirlineNames θα επέτρεπε τότε στον αλγόριθμο να επιλέξει την GetWebSites στη συνέχεια της ακολουθίας.

Ο σκοπός της φάσης ανακάλυψης υπηρεσίας και λειτουργιών ταιριάσματος είναι να αντικαταστήσει τις εικονικές λειτουργίες με concrete λειτουργίες, βεβαιώνοντας ότι η ακολουθία που ανακτάται είναι ακόμα εφικτά βασισμένη σε διαθέσιμες συνδέσεις σε κάθε θέση της ακολουθίας. Ο αλγόριθμος διαπερνά την ακολουθία και αρχίζει μια αναζήτηση σε κάθε μια εικονική λειτουργία. Η αναζήτηση θα πρέπει να επιστρέφει μια concrete λειτουργία με την υψηλότερη τιμή για την objective function, όπως καθορίστηκε προηγουμένως. Αρχίζει ψάχνοντας σχετικές Υπηρεσίες Διαδικτύου μέσω των UDDI καταλόγων, χρησιμοποιώντας τις ιδιότητες Category και Function της εικονικής λειτουργίας για να φτιάξει ένα UDDI inquiry.

Η αναζήτηση χάχνει την περιγραφή κάθε επιστρεφόμενης Υπηρεσίας Διαδικτύου για διαδικασίες που ταιριάζουν με την εικονική λειτουργία χρησιμοποιώντας ένα από τα τέσσερα επίπεδα αντιστοιχίας. Για το προηγούμενο ερώτημα Q, διάφοροι διαφορετικοί πάροχοι θα μπορούσαν να προσφέρουν και τις δύο εικονικές λειτουργίες InquireAirlines και GetWebSites με διαφορετική QoWS. Ο αλγόριθμος επιλέγει την concrete λειτουργία με την υψηλότερη τιμή για την objective συνάρτηση. Επειδή μπορεί να εμφανιστεί μια μερική αντιστοιχία, ο αλγόριθμος πρέπει να ελέγξει ότι δεν υπάρχει εικονική λειτουργία στην ακολουθία στην οποία οι είσοδοι εξαρτώνται από ελλιπείς εξόδους (πράγμα που επιτρέπει μια μερική αντιστοιχία). Εάν απαιτούνται οι έξοδοι που λείπουν, ο αλγόριθμος εκτελεί την αναζήτηση ξανά μέχρι να βρει μια κατάλληλη concrete λειτουργία. Ο αλγόριθμος τερματίζει όταν έχει αντικαταστήσει όλες τις εικονικές λειτουργίες στην ακολουθία με concrete λειτουργίες, ή όταν καθορίσει ότι δεν μπορεί να ταιριάξει μια εικονική λειτουργία σε καμία υπάρχουσα concrete λειτουργία.

8.2.2. Υλοποίηση

Μια υλοποίηση μηχανής ερωτημάτων υπηρεσιών (service query engine) που εφαρμόζει την προσέγγιση ερωτημάτων που παρουσιάστηκε θα περιλάμβανε τα ακόλουθα συστατικά:

• Ο **service locator** που ανακαλύπτει τις WSDL περιγραφές προσπελαύνοντας τον UDDI κατάλογο. Μόλις ο service locator ανακαλύψει μια υπηρεσία, η μηχανή ερωτημάτων καλεί της λειτουργίες μέσω του SOAP binding stub.

• Ο **operation matchmaker** αλληλεπιδρά με τον service locator για να ανακτήσει τις περιγραφές των υπηρεσιών. Καθορίζει τις concrete λειτουργίες που θα χρησιμοποιηθούν στο σχέδιο εκτέλεσης υπηρεσίας.

• Ο **monitoring agent** παρακολουθεί τις κλήσεις των Υπηρεσιών Διαδικτύου. Ο στόχος του είναι να προσπελάσει την συμπεριφορά τους από την άποψη της παραδοθείσας QoWS.

• Ο **query optimizer**, το βασικό κομμάτι της μηχανής ερωτημάτων, καθορίζει το καλύτερο σχέδιο εκτέλεσης υπηρεσίας βάση της QoWS, των ratings υπηρεσιών, και των βαθμών ταιριάσματος.

• Η **execution engine** αναλαμβάνει αφού ο βελτιστοποιητής παράγει ένα αποτελεσματικό σχέδιο εκτέλεσης υπηρεσίας. Η μηχανή εκτέλεσης θεσπίζει το σχέδιο εκτέλεσης υπηρεσίας καλώντας πραγματικά Υπηρεσίες Διαδικτύου χρησιμοποιώντας SOAP. Η μηχανή ερωτημάτων υπηρεσιών λαμβάνει ερωτήματα Υπηρεσιών Διαδικτύου ως συνδυαστικά ερωτήματα πάνω σε σχέσεις. Αναλαμβάνει τη σωστή και βέλτιστη εκτέλεση των Υπηρεσιών Διαδικτύου.

8.3. Συμπεράσματα για το μοντέλο ερωτημάτων

Το προτεινόμενο πρότυπο επιλέγει και συνδυάζει τις κατάλληλες διαδικασίες βασισμένο στη σχετικότητα, την QoWS, τους βαθμούς ταιριάσματος, τα ratings, και την εφικτότητα. Το πρότυπο αυτό μπορεί να επεκταθεί με διάφορους τρόπους. Τεχνικές βελτιστοποίησης βάσει σημασιολογίας για Υπηρεσίες Διαδικτύου θα χρησιμοποιούσαν έξυπνους πράκτορες για να εκμεταλλευτούν το τρέχον πλαίσιο (τη σημασιολογία της εφαρμογής) προκειμένου να ενισχυθεί η βελτιστοποίηση. Θα μπορούσαν επίσης να σχεδιαστούν προσαρμοστικές τεχνικές που να αντισταθμίζουν τα αποτελέσματα του σχεδίου εκτέλεσης υπηρεσίας όταν συμβούν απρόβλεπτα γεγονότα κατά την εκτέλεση. Τέτοιες προσαρμοστικές τεχνικές θα μπορούσαν, παραδείγματος χάριν, να αντικαταστήσουν μια Υπηρεσία Διαδικτύου που δεν είναι διαθέσιμη με μια άλλη που παρέχει παρόμοιες λειτουργίες. Η στρατηγική αντικατάστασης δεν πρέπει να μειώσει τη γενική ποιότητα του σχεδίου εκτέλεσης υπηρεσίας

9. Το πλαίσιο WS-Resource

Οι διεπαφές των Υπηρεσιών Διαδικτύου παρέχουν συχνά στο χρήστη τη δυνατότητα να προσπελαύνει και να διαχειρίζεται την κατάσταση (state), δηλαδή τιμές δεδομένων που διατηρούνται κατά τη διάρκεια και καταλήγουν ως αποτέλεσμα των αλληλεπιδράσεων των Υπηρεσιών Διαδικτύου. Οι ανταλλαγές μηνυμάτων που εφαρμόζουν οι Υπηρεσίες Διαδικτύου συχνά αποσκοπούν στο να επιτρέψουν την πρόσβαση σε stateful (με καταστάσεις) πόρους. Παρόλα αυτά, η έννοια των stateful πόρων που εφαρμόζεται από την υλοποίηση των Υπηρεσιών Διαδικτύου δεν είναι ρητή στον καθορισμό των διεπαφών. Τα μηνύματα που στέλνουν και λαμβάνουν οι υπηρεσίες υπονοούν την ύπαρξη ενός σχετικού stateful είδους πόρου. Είναι επιθυμητό λοιπόν να οριστούν συμβάσεις Υπηρεσιών Διαδικτύου που να επιτρέπουν την ανακάλυψη και αλληλεπίδραση με stateful πόρους μεσω τυποποιημένων και διαλειτουργικών μεθόδων.

Η προσέγγιση WS-Resource μοντελοποιεί τις καταστάσεις σε ένα πλαίσιο Υπηρεσιών Ιστού. Μια WS-Resource ορίζεται ως η σύνθεση μιας Υπηρεσίας Διαδικτύου και ενός stateful πόρου που 1) εκφράζεται ως μια συσχέτιση ενός XML εγγράφου ορισμένου τύπου με ένα portType Υπηρεσιών Διαδικτύου, και 2) απευθύνεται και προσπελαύνεται σύμφωνα με ένα υπονοούμενο πρότυπο πόρων, μια συμβατική χρήση των WS-Addressing endpoint αναφορών. Στο υπονοούμενο πρότυπο πόρων, ένα αναγωριστικό (identifier) stateful πόρου συμπεριλαμβάνεται σε μια endpoint αναφορά και χρησιμοποιείται για να αναγνωρίσει τον stateful πόρο που θα χρησιμοποιηθεί στην εκτέλεση μιας ανταλλαγής μηνυμάτων Υπηρεσιών Διαδικτύου. Το πλαίσιο WS-Resource επιτρέπει στα WS-Resources να ορίζονται, να δημιουργούνται, να προσπελαύνονται, να παρακολουθούνται για αλλαγές και να καταστρέφονται μέσω μηχανισμών Υπηρεσιών Διαδικτύου, αλλά δεν απαιτεί το τμήμα Υπηρεσιών Διαδικτύου του WS-Resource που παρέχει πρόσβαση στους σχετικούς stateful πόρους να υλοποιηθεί ως επεξεργαστής stateful μηνυμάτων.

Στη συνέχεια παρουσιάζεται το πλαίσιο WS-Resource [23], ένα σύνολο από πέντε τεχνικές προδιαγραφές που ορίζουν την κανονιστική περιγραφή της προσέγγισης WS-Resource με όρους ανταλλαγής μηνυμάτων Υπηρεσιών Διαδικτύου και σχετικών XML ορισμών.

Αυτές οι τεχνικές προδιαγραφές ορίζουν τα μέσα με τα οποία :

• ένα WS-Resource μπορεί να καταστραφεί, είτε συγχρόνως με την αίτηση καταστροφής είτε μέσω μηχανισμών που παρέχουν χρονοπρογραμματισμένη καταστροφή. Συγκεκριμένες ιδιότητες πόρων [WS-ResourceProperties] μπορούν να χρησιμοποιηθούν για να παρακολουθήσουν και να ελέγξουν τη διάρκεια ζωής ενός WS-Resource (WSResourceLifetime)

• ο καθορισμός τύπου ενός WS-Resource μπορεί να συντεθεί από την περιγραφή διεπαφών μιας Υπηρεσίας Διαδικτύου και ένα XML έγγραφο ιδιοτήτων των πόρων. Η κατάσταση ενός WS-Resource μπορεί να ερωτηθεί και να τροποποιηθεί μέσω ανταλλαγών μηνυμάτων Υπηρεσιών Διαδικτύου (WS-ResourceProperties)

• μια endpoint αναφορά Υπηρεσιών Διαδικτύου (WS-Addressing) μπορεί να ανανεωθεί στην περίπτωση που οι πληροφορίες που περιλαμβάνονται μέσα σε αυτήν γίνουν άκυρες (WS-RenewableReferences)

•ετερογενείς συλλογές by-reference των Υπηρεσιών Διαδικτύου μπορούν να οριστούν, είτε οι υπηρεσίες είναι WS-Resources (WS-ServiceGroups) είτε όχι, και

• η αναφορά λαθών μπορεί να γίνει περισσότερο τυποποιημένη μέσω ενός XML σχήματος για λάθη και κανόνες του πώς αυτός ο τύπος λάθους χρησιμοποιείται και επεκτείνεται από τις Υπηρεσίες Διαδικτύου (WS-BaseFaults).

Στη συνέχεια παρουσιάζεται το πλαίσιο WS-Resource και οι πέντε τεχνικές προδιαγραφές του. Επίσης παρουσιάζεται ο τρόπος που ένας μηχανισμός ειδοποίησης δημοσίευσης /συνδρομής (WS-Notification) μπορεί να βασιστεί στο πλαίσιο WSResource, δημιουργώντας συνδρομές στις

αλλαγές καταστάσεων του WSResource για την παρακολούθηση και την αναφορά των αλλαγών της ιδιότητας πόρου.

9.1. WS-Resource

Ο ορισμός του WS-Resource κωδικοποιεί τη σχέση μεταξύ Υπηρεσιών Διαδικτύου και των stateful πόρων από την άποψη του υπονοούμενου προτύπου πόρων, ένα σύνολο συμβάσεων σχετικά με τις τεχνολογίες Υπηρεσιών Διαδικτύου, και ιδιαίτερα την XML, την WSDL, και την WS-Addressing [WSAddressing]. Αυτές οι συμβάσεις επιτρέπουν στην κατάσταση ενός πόρου που συμμετέχει στο υπονοούμενο σχέδιο πόρων να οριστεί και να συνδεθεί με την περιγραφή μιας διεπαφής Υπηρεσιών Διαδικτύου. Η κατάσταση ενός πόρου καθορίζεται βάση ενός εγγράφου ιδιοτήτων των πόρων.

Το πρότυπο πόρων ορίζει μια συμβατική χρήση του WS-Addressing στο οποίο ένας stateful πόρος αντιμετωπίζεται ως είσοδος για την επεξεργασία των ανταλλαγών μηνυμάτων που υλοποιούνται από μια Υπηρεσία Διαδικτύου. Μια endpoint αναφορά μπορεί να περιλαμβάνει ένα στοιχείο παιδί ReferenceProperties που αναγνωρίζει τον stateful πόρο που θα χρησιμοποιηθεί στην εκτέλεση της ανταλλαγής μηνυμάτων χρησιμοποιώντας αυτήν την EndpointReference. Αυτός ο τύπος endpoint αναφοράς αναφέρεται ως μια WS-Resourcequalified endpoint reference. Ένα μήνυμα αιτήματος που απευθύνεται σε μια Υπηρεσία Διαδικτύου που σχεδιάζεται από μια WS-Resource-qualified endpoint αναφορά πρέπει να περιλαμβάνει πληροφορίες ReferenceProperties από την endpoint αναφορά, όπως ορίζεται από την προδιαγραφή WSAddressing.

Επομένως, το WS-Resource πλαίσιο χρησιμοποιεί μια WS-Resource-qualified endpoint αναφορά για να αναπαραστήσει έναν "network-wide pointer" σε έναν WS-Resource. Μια WS-Resource-qualified endpoint αναφορά μπορεί να επιστραφεί ως αποτέλεσμα ενός μηνύματος αιτήματος σε ένα εργοστάσιο για τη δημιουργία ενός WS-Resource ή εναλλακτικά από την αξιολόγηση του ερωτήματος αναζήτησης στον κατάλογο υπηρεσιών, είτε ως αποτέλεσμα κάποιου application-specific αιτήματος Υπηρεσίας Διαδικτύου.

9.2. Κύκλος ζωής του WS-Resource

Η προδιαγραφή WS-ResourceLifetime χειρίζεται τρεις σημαντικές πλευρές του κύκλου ζωής ενός WSResource, δηλαδή τη δημιουργία, την αναγνώριση, και την καταστροφή.

9.2.1. Το πρότυπο WS-Resource Factory

Το πλαίσιο WS-Resource δεν επιχειρεί να ορίσει τις ανταλλαγές μηνυμάτων που χρησιμοποιούνται για τη δημιουργία νέων WS-Resources. Αντίθετα, απλά δηλώνει τα νέα WS-Resources μπορούν να δημιουργηθούν από τη χρήση ενός πρότυπου χρήσης για Υπηρεσίες Διαδικτύου που η προδιαγραφή WS-ResourceLifetime ονομάζει WS-Resource factory. Ένα WS-Resource factory είναι οποιαδήποτε Υπηρεσία Διαδικτύου είναι ικανή να δημιουργήσει ένα ή περισσότερα WS-Resources. Το μήνυμα απάντησης μιας λειτουργίας WS-Resource factory τυπικά περιέχει τουλάχιστον μια endpoint αναφορά που αναφέρεται στον νέο WS-Resource, παρόλο που ένα εργοστάσιο μπορεί να μετατρέψει την αναφορά στο νέο WSResource μέσω άλλων τρόπων, όπως τοποθετώντας τις WS-Resource-qualified endpoint αναφορές σε έναν κατάλογο για μεταγενέστερη ανάκτηση.

9.2.2. WS-Resource Ταυτότητα

Στη συνέχεια περιγράφεται και αντιπαραβάλλεται ο ρόλος και η χρήση της ταυτότητας των WS-Resources από την άποψη 1) της υλοποίησης των WS-Resources, και 2) ενός πελάτη υπηρεσιών στον οποίο παρέχεται μια endpoint αναφορά σε ένα WS-Resource. Η μορφή και το περιεχόμενο του stateful προσδιοριστικού των πόρων που μεταφέρεται στις ιδιότητες αναφοράς περιλαμβάνεται πλήρως μέσα στην υλοποίηση WS- Resource.

Το τμήμα Υπηρεσιών Διαδικτύου ενός WS-Resource μπορεί να κατασκευάσει μια διεύθυνση για το WSResource συμπεριλαμβάνοντας ένα προσδιοριστικό του stateful πόρου στο τμήμα αναφοράς ιδιοτήτων μιας endpoint αναφοράς WS- Addressing. Η endpoint αναφορά λέγεται τότε ότι είναι WS-Resource-qualified. Η WS-Resource-qualified endpoint αναφορά μπορεί έπειτα να τεθεί στην διάθεση άλλων οντοτήτων στο κατανεμημένο σύστημα, το οποίο μπορεί στη συνέχεια να χρησιμοποιήσει αυτήν την αναφορά σε άμεσα αιτήματα στον WS- Resource. Λογικά, αυτά τα αιτήματα "διατρέχουν" το τμήμα Υπηρεσιών Διαδικτύου του WS-Resource, το όποιο καταλαβαίνει το περιεχόμενο του εξαρτώμενου από την υλοποίηση stateful προσδιοριστικό πόρων που περιέχεται στην endpoint αναφορά WS-Address.

Μέρος της WS-Address endpoint αναφοράς προσδιορίζει την υπηρεσία, η οποία χρησιμοποιεί στη συνέχεια τις ιδιότητες αναφοράς για να προσδιορίσει τον stateful πόρο που χρησιμοποιείται στην εκτέλεση του μηνύματος. Αντίθετα, ένα αίτημα υπηρεσιών που λαμβάνει πρόσβαση σε μια WS-Resource-qualified endpoint αναφορά δεν πρέπει να εξετάσει ή να προσπαθήσει να ερμηνεύσει το περιεχόμενο των ιδιοτήτων αναφοράς που αντιπροσωπεύουν το stateful προσδιοριστικό πόρων. Ακόμη και μια προσπάθεια από το αίτημα υπηρεσίας να συγκρίνει δύο stateful προσδιοριστικά πόρων θεωρείται άκυρη. Από την πλευρά του αιτήματος υπηρεσίας, το περιεχόμενο του τμήματος ιδιοτήτων μιας WS-Resource endpoint αναφοράς είναι αδιαφανές. Η σημασιολογική έννοια της ταυτότητας του τμήματος stateful πόρων ενός WS-Resource, και τα μέσα με τα οποία καθορίζεται και εκτίθεται σε έναν αιτούντα υπηρεσιών, εξαρτάται από την υλοποίηση του WS-Resource. Αυτή τη στιγμή, δεν υπάρχουν προδιαγραφές Υπηρεσιών Διαδικτύου που να παρέχουν τον ορισμό της ταυτότητας των stateful πόρων. Ούτε υπάρχει οποιοσδήποτε καθορισμός των μέσων με τα οποία ένα αίτημα υπηρεσιών λαμβάνει την ταυτότητα ενός stateful πόρου.

Εάν η ταυτότητα ενός τμήματος stateful των πόρων ενός WS- Resource εκτίθεται ή όχι σε ένα αίτημα υπηρεσίας εξαρτάται από το σχεδιασμό του WS-Resource. Η ταυτότητα θα πρέπει να είναι μια φορητή, namespace-scoped τιμή. Η φορητότητα είναι σημαντική δεδομένου ότι επιτρέπει μια εφαρμογή να περάσει την ταυτότητα σε μια άλλη. Το Namespace scoping είναι σημαντικό καθώς επιτρέπει την αποσαφήνιση των πολλαπλών ταυτοτήτων που μπορούν να δημιουργηθούν από τις διαφορετικές πηγές.

Ένα αίτημα Υπηρεσιών Διαδικτύου που λαμβάνει μια WS-Addressing endpoint αναφορά μπορεί να περάσει την endpoint αναφορά σε άλλες υπηρεσίες, με τη διαβεβαίωση ότι ο δέκτης μπορεί καλέσει διαδικασίες που περιλαμβάνουν την WS-Resource instance. Αυτό είναι θεμελιώδες στο WSAddressing. Προβλέπεται ότι μια κοινή προσέγγιση για την έκθεση της ταυτότητας του τμήματος stateful πόρων ενός WS- Resource θα είναι να αντιμετωπίζει την ταυτότητα ως μια ή περισσότερες ιδιότητες πόρων που εκφράζονται στο WS- Resource έγγραφο ιδιοτήτων. Αυτή η προσέγγιση θα επέτρεπε σε ένα αίτημα υπηρεσιών να απευθύνει μια ερώτηση στο έγγραφο, απευθυνόμενο στις ιδιότητες που γίνονται κατανοητές ότι αντιπροσωπεύουν την ταυτότητα του stateful πόρου. Εάν η ταυτότητα εκτίθεται ως μια ή περισσότερες ιδιότητες πόρων, το WS-Resource θα πρέπει να εξασφαλίσει μόνο πρόσβαση ανάγνωσης σε εκείνες τις ιδιότητες. Τυπικά, θα ήταν άκυρο να επιτραπεί σε ένα αίτημΑ υπηρεσιών να αλλάξει την ταυτότητα ενός stateful πόρου.

9.2.3. Καταστροφή WS-Resource

Ένα αίτημα που ζητά από ένα WS-Resource εργοστάσιο να δημιουργήσει ένα νέο WS-Resource τυπικά θα ενδιαφέρεται μόνο για εκείνο το νέο WS-Resource για κάποια πεπερασμένη περίοδο. Μετά από εκείνο τον χρόνο, πρέπει να είναι δυνατό να καταστραφεί το WS-Resource έτσι ώστε οι πόροι του σχετικού συστήματος ή της εφαρμογής να μπορούν να ανακτηθούν. Το πλαίσιο WS-Resource τυποποιεί δύο προσεγγίσεις διαχείρισης της διάρκειας ζωής για την καταστροφή ενός WSResource: άμεση (immediate) και σχεδιασμένη (scheduled) καταστροφή.

9.2.3.1. Άμεση καταστροφή

Σε πολλές περιπτώσεις, είναι κατάλληλο στις εφαρμογές που χρησιμοποιούν ένα WS-Resource να ζητούν να καταστραφεί αμέσως. Ένα αίτημα υπηρεσιών που επιθυμεί να καταστρέψει ένα WS-Resource άμεσα πρέπει να χρησιμοποιήσει την κατάλληλη WS-Resource-qualified endpoint αναφορά για να στείλει ένα μήνυμα καταστροφής αιτήματος στο WS-Resource που προσδιορίζεται από την endpoint αναφορά. Το προσδιοριστικό stateful πόρων μέσα στην endpoint αναφορά χρησιμοποιείται για να προσδιορίσει το τμήμα stateful πόρων που καταστρέφεται. Η παραλαβή της απάντησης στο μήνυμα καταστροφής αιτήματος αντιπροσωπεύει ένα σημείο συγχρονισμού μεταξύ του αιτήματος υπηρεσιών και του WS-Resource που λαμβάνει το μήνυμα καταστροφής αιτήματος. Κατά την λήψη του μηνύματος απάντησης, κάθε περαιτέρω απόπειρα ανταλλαγής μηνυμάτων με το WS-Resource πρέπει να καταλήξει στο μήνυμα λάθους *Unknown Resource*, εφόσον δεν προηγηθούν οποιεσδήποτε άλλες συνθήκες λαθών .

9.2.3.2. Προγραμματισμένη καταστροφή

Ένα αίτημα μπορεί να είναι απρόθυμο να καταστρέψει έναν WS-Resource αμέσως και συγχρόνως, ή μπορεί πράγματι να είναι ανίκανο να το κάνει σε έναν κατανεμημένο υπολογιστικό περιβάλλον, επειδή έχει αποσυνδεθεί από το endpoint του προμηθευτή υπηρεσίας. Κατά συνέπεια, εκτός από τη δυνατότητα να καταστραφεί ένα WS-Resource άμεσα, καθορίζονται τα μέσα με τα οποία ένας WS-Resource μπορεί να σχεδιαστεί για λήξει σε μελλοντικό χρόνο. Χρησιμοποιώντας μια WS-Resource-qualified endpoint αναφορά, το αίτημα υπηρεσιών μπορεί αρχικά να καθιερώσει και στη συνέχεια να ανανεώσει το σχεδιασμένο χρόνο λήξης του WS-Resource. Όταν εκείνος ο χρόνος λήξει, το WS-Resource μπορεί να αυτοκαταστραφεί χωρίς την ανάγκη για έναν σύγχρονο αίτημα καταστροφής από ένα αίτημα υπηρεσίας. Ένα αίτημα μπορεί περιοδικά να ενημερώσει το σχεδιασμένο χρόνο λήξης για να ρυθμίσει τη διάρκεια ζωής του WS-Resource. Η προδιαγραφή WSResourceLifetime καθορίζει μια πρότυπη ανταλλαγή μηνυμάτων από την οποία ένα αίτημα υπηρεσιών μπορεί αρχικά να καθιερώσει και στη συνέχεια να αλλάξει το σχεδιασμένο χρόνο λήξης ενός WS-Resource, και ορίζει έναν τρόπο καθορισμού του τρέχοντος σχεδιασμένου χρόνου λήξης ενός WS-Resource.

Ένα WS-Resource εργοστάσιο μπορεί να υποστηρίξει τη δυνατότητα να διαπραγματευτεί ένας αρχικά σχεδιασμένος χρόνος λήξης όταν δημιουργείται ένας WS-Resource. Στη συνέχεια, οι εξουσιοδοτημένοι αιτούντες υπηρεσιών μπορούν να χρησιμοποιήσουν την WS-Resource-qualified endpoint αναφορά για να ζητήσουν να αλλάξουν αυτόν τον σχεδιασμένο χρόνο λήξης, με την αποστολή του μηνύματος αίτησης στο WS-Resource. Εάν φτάσει ο χρόνος λήξης του WS-Resource, μπορεί να καταστραφεί και οποιοιδήποτε πόροι σχετικών συστημάτων μπορούν να ανακτηθούν. Ο χρόνος λήξης των WS-Resource μπορεί να αλλάξει μη-μονοτονικά. Δηλαδή ένα αίτημα υπηρεσίας μπορεί να ζητήσει έναν χρόνο λήξης που είναι νωρίτερα από το χρόνο λήξης που είναι ήδη συνδεμένος με το WS-Resource. Εάν ο ζητούμενος χρόνος λήξης αντιπροσωπεύει έναν χρόνο πριν από τον τρέχοντα χρόνο, όπως είναι γνωστό από την WS- Resource εφαρμογή, το αίτημα πρέπει να ερμηνευθεί ως αίτημα για άμεση, αλλά όχι σύγχρονη, καταστροφή του WS-Resource.

9.3. WS-Resource Ιδιότητες

Η προδιαγραφή WS-ResourceProperties καθορίζει τον τύπο και τις τιμές εκείνων των συστατικών της κατάστασης ενός WS-Resource που μπορούν να αντιμετωπισθούν και να τροποποιηθούν από αιτούντες υπηρεσίας μέσω μιας διεπαφής Υπηρεσιών Διαδικτύου. Οι βασικές ιδέες είναι οι ακόλουθες.

- Το WS-Resource έχει ένα XML έγγραφο ιδιότητας πόρων που καθορίζεται χρησιμοποιώντας ένα XML schema.

- Τα αιτήματα υπηρεσιών μπορούν να καθορίσουν έναν τύπο WS- Resource ανακτώντας τον WSDL portType ορισμό με τα τυποποιημένα μέσα.
- Τα αιτήματα υπηρεσιών μπορούν να χρησιμοποιήσουν ανταλλαγές μηνυμάτων Υπηρεσιών Διαδικτύου για να διαβάσουν, τροποποιήσουν και να ρωτήσουν το XML έγγραφο που αναπαριστά την κατάσταση του WS-Resource.

Ο όρος resource property χρησιμοποιείται για να αναφερθεί σε ένα ανεξάρτητο συστατικό μιας κατάστασης WS-Resource. Το έγγραφο XML που περιγράφει τον τύπο ενός τμήματος stateful πόρων ενός WS-Resource καλείται WS-Resource properties έγγραφο. Κάθε ιδιότητα πόρων αντιπροσωπεύεται ως ένα στοιχείο XML μέσα στο WS- Resource έγγραφο ιδιοτήτων. Η χρήση XML είναι λογική.

9.4. Renewable References

Το WS-Renewable References πρέπει να καθορίσει τους μηχανισμούς που μπορούν να χρησιμοποιηθούν για να ανανεώσουν μια endpoint αναφορά που έχει γίνει άκυρη. Αυτοί οι μηχανισμοί μπορούν να είναι εφαρμοσμένοι σε οποιαδήποτε endpoint αναφορά, αλλά είναι ιδιαίτερα χρήσιμοι στην περίπτωση μιας endpoint αναφοράς που αναφέρεται σε ένα WS-Resource, δεδομένου ότι μπορεί να παρέχει μια σταθερή αναφορά στο WS-Resource που μπορεί να επιτρέψει στην ίδια κατάσταση και να προσπελαστεί επανειλημμένα κατά τη διάρκεια του χρόνου. Μια WS-Addressing endpoint αναφορά μπορεί να περιέχει όχι μόνο addressing αλλά και πληροφορίες πολιτικής σχετικά με τις αλληλεπιδράσεις με την υπηρεσία. Χαρακτηριστικά, endpoint αναφορές κατασκευάζεται από μια επιτακτική πηγή των πληροφοριών εξέτασης και πολιτικής. Οι WS-Renewable References πρέπει να καθορίσουν μια συγκεκριμένη WS-Policy assertion με σκοπό να εφοδιάσουν τις endpoint αναφορές με τις απαραίτητες πληροφορίες για να ανακτήσουν μια νέα endpoint αναφορά σε περίπτωση που η αναφορά γίνει άκυρη.

9.5. Ομάδες Υπηρεσιών

Η προδιαγραφή WS-ServiceGroup πρέπει να καθορίσει μέσα αναπαράστασης και διαχείρισης των ετερογενών συλλογών των Υπηρεσιών Διαδικτύου. Αυτή η προδιαγραφή μπορεί να χρησιμοποιηθεί για να οργανώσει τις συλλογές των WS-Resources, παραδείγματος χάριν για να χτίσουν τους καταλόγους, ή τις υπηρεσίες που μπορούν να εκτελέσουν τις συλλογικές διαδικασίες σε μια συλλογή WS-Resources.

Η προδιαγραφή ServiceGroup πρέπει να εκφράσει τους κανόνες ιδιότητας μέλους ServiceGroup, τους περιορισμούς ιδιότητας μέλους, και τις ταξινομήσεις που χρησιμοποιούν το πρότυπο ιδιότητας όρων από τις WS-ResourceProperties. Οι ομάδες μπορούν να οριστούν ως μια συλλογή μελών που συναντούν τους περιορισμούς της ομάδας όπως εκφράζεται μέσω των ιδιοτήτων πόρων. Η προδιαγραφή ServiceGroup πρέπει επίσης να καθορίσει τις διεπαφές για τη διαχείριση της ιδιότητας μέλους ενός ServiceGroup. Οι διεπαφές που καθορίζονται από την Ws-ServiceGroup αναμένεται να συντεθούν με άλλες Διεπαφές Υπηρεσιών Διαδικτύου, που καθορίζουν πιο εξειδικευμένη αλληλεπίδραση με την ομάδα υπηρεσίας ομάδας/ή με τις υπηρεσίες που είναι μέλη του ServiceGroup.

9.6. Base Faults

Η προδιαγραφή WS-BaseFaults πρέπει να καθορίσει έναν βασικό τύπο λαθών για να χρησιμοποιείται όταν επιστρέφονται λάθη σε μια ανταλλαγή μηνυμάτων Υπηρεσιων Διαδικτύου. Χρησιμοποιείται από όλες τις άλλες προδιαγραφές του πλαισίου WS-Resource για να προσδώσει συνέπεια στα λάθη που επιστρέφονται από διαδικασίες σε αυτές τις προδιαγραφές, συμπεριλαμβανομένης της συνεπούς υποβολής αναφορών λαθών σχετικά με την χρήση τουWS-Resource.

9.7. Ειδοποίηση

Μια χωριστή οικογένεια προδιαγραφών, αποκαλούμενη WS-Notification, καθορίζει ένα γενικό σύστημα Υπηρεσιών Διαδικτύου για αλληλεπιδράσεις δημοσίευσης και συνδρομής που στηρίζονται στο πλαίσιο WS-Resource. Η βασική μέθοδος που υιοθετείται είναι να καθορίζει μηχανισμούς και διεπαφές που επιτρέπουν στους πελάτες να γίνονται συνδρομητές σε θέματα ενδιαφέροντος, όπως αλλαγές τιμής της ιδιότητας πόρων για έναν WS-Resource. Από την πλευρά του WS-Notification, το πλαίσιο WS-Resource παρέχει τις χρήσιμες δομικές μονάδες για την αναπαράσταση και δόμηση των ανακοινώσεων. Από την πλευρά του WS-Resource πλαισίου, η οικογένεια των προδιαγραφών WS-Notification επεκτείνει τη χρησιμότητα των WS-Resources επιτρέποντας στους πελάτες να ζητήσουν να ειδοποιηθούν ασύγχρονα για τις αλλαγές των τιμές ιδιοτήτων πόρων.

Μια προδιαγραφή WS-Notification, η WS-BaseNotification, περιγράφει τους βασικούς ρόλους, έννοιες, και σχέδια που απαιτούνται για να επιτρέψουν σε έναν συνδρομητή να καταχωρήσει το ενδιαφέρον του να λαμβάνει μηνύματα ανακοίνωσης από έναν παραγωγό ανακοίνωσης. Μια ανακοίνωση μπορεί να έχει σχέση με οτιδήποτε, μια αλλαγή στην αξία μιας ιδιότητας πόρων, κάποια άλλη εσωτερική αλλαγή στην κατάσταση του παραγωγού ανακοίνωσης, ή κάποια άλλη "κατάσταση" μέσα στο περιβάλλον. Ένας συνδρομητής καταχωρεί το ενδιαφέρον του να λάβει τα μηνύματα ανακοίνωσης πάνω σε ένα ή περισσότερα θέματα με την έκδοση ενός μηνύματος “subscribe”. Σε απάντηση, ο συνδρομητής λαμβάνει μια WS-Resource-qualified endpoint αναφορά σε μια WS-Resource "συνδρομή". Η WS-Resource συνδρομή διαμορφώνει αυτήν την σχέση μεταξύ συνδρομητή και ο παραγωγού, και χρησιμοποιεί τη WS-ResourceProperties και τη WS-ResourceLifetime για να βοηθήσουν να διαχειριστεί αυτήν την σχέση.

Η δεύτερη προδιαγραφή WS-Notification, η WS-Topics, παρουσιάζει μια περιγραφή XML των θεμάτων και σχετικά meta- δεδομένα. Τα θέματα είναι ένας μηχανισμός για την διοργάνωση των μηνυμάτων ανακοίνωσης έτσι ώστε οι συνδρομητές να μπορούν να καταλάβουν, ποιοι τύποι ανακοινώσεων είναι διαθέσιμοι για συνδρομή. Τα θέματα μπορούν να οργανωθούν ιεραρχικά, δηλαδή ένα θέμα μπορεί να αποσυντεθεί περαιτέρω με θέματα παιδιά.

Η τρίτη προδιαγραφή WS-Notification, η WS-BrokeredNotification, καθορίζει τη διεπαφή σε ένα NotificationBroker που υλοποιεί μια ενδιάμεση υπηρεσία που διαχειρίζεται συνδρομές για άλλες οντότητες στο σύστημα που παράγει τα μηνύματα ανακοίνωσης.

10. Περιγραφή του μεσολαβητή διαχείρισης υπηρεσίας με κριτήρια ποιότητας.

Το WSRF (Web Services Resource Framework) όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο είναι ένα νέο σύνολο προδιαγραφών που παρέχουν ένα κοινό πλαίσιο στο οποίο μπορεί να αναπτυχθεί η μελλοντική τεχνολογία συνεργαζόμενων υπηρεσιών ή grid. Αυτό το πλαίσιο ορίζει την έννοια των "stateful πόρων" που μπορούν να ανακαλυφθούν, ερωτηθούν και διαχειριστούν μέσω Υπηρεσιών Διαδικτύου.

Στα πλαίσια αυτής της διπλωματικής υλοποιήθηκε ένα σύστημα διαχείρισης Υπηρεσιών Διαδικτύου χρησιμοποιώντας την τεχνολογία των WS Resources. Η πλατφόρμα ανάπτυξης που χρησιμοποιήθηκε είναι το WSRF.NET, το οποίο επιτρέπει τη δημιουργία WS-Resource τύπων καθώς και τη διατήρηση και ανάκτηση πόρων με τεχνολογία .NET. Χρησιμοποιούνται επίσης οι προδιαγραφές WS-Notification και WS-ServiceGroups για να διευκολύνουν την ασύγχρονη ειδοποίηση γεγονότων και την ομαδοποίηση των WS-Resources.

Ο μεσολαβητής (broker) ποιότητας υπηρεσίας διαχειρίζεται ένα σύνολο Υπηρεσιών Διαδικτύου, τις διευθύνσεις και τα χαρακτηριστικά ποιότητας των οποίων διατηρεί σε μια βάση δεδομένων. Στην πειραματική εφαρμογη, ο broker συλλέγει πληροφορίες QoS για τους παρόχους υπηρεσιών (εξυπηρετητές) και όταν λάβει αιτήσεις υπηρεσιών από πελάτες αναγνωρίζει τις υπηρεσίες που μπορούν να ικανοποιήσουν τόσο τις λειτουργικές αιτήσεις όσο και τις απαιτήσεις Ποιότητας Υπηρεσίας των πελατών.

Ο αλγόριθμος επιλογής υπηρεσίας που χρησιμοποιείται από τον broker για την επιλογή της κατάλληλης υπηρεσίας χρησιμοποιεί ως κριτήριο τη δυναμική χωρητικότητα του εξυπηρετητή (server load). Άλλα κριτήρια θα μπορούσαν να είναι το κόστος υπηρεσίας, ο χρόνος απόκρισης κλπ. Ορίζουμε ως διαθεσιμότητα εξυπηρετητή την πιθανότητα μια υπηρεσία να είναι διαθέσιμη και κλάση υπηρεσιών μια συλλογή από ανεξάρτητες Υπηρεσίες Διαδικτύου με κοινή λειτουργικότητα, αλλά διαφορετικά μη λειτουργικά χαρακτηριστικά. Κάθε υπηρεσία έχει μια μέγιστη χωρητικότητα Cmax που είναι ο μέγιστος αριθμός πελατών που μπορεί να δεχτεί και μια τρέχουσα χωρητικότητα που Ccur είναι ο τρέχον αριθμός πελατών που δέχονται την υπηρεσία.

Στέλνοντας ένα νέο αίτημα υπηρεσίας σε έναν εξυπηρετητή με ελαφρύτερο φόρτο, ένας πελάτης μπορεί να λάβει γρηγορότερα απάντηση από ότι από ένα busy εξυπηρετητή. Επιπλέον, τέτοιες αποφάσεις θα κατανέμουν το φόρτο εργασίας σε περισσότερους εξυπηρετητές και δημιουργούν ένα πιο ζυγισμένο σύστημα. Με αυτή τη λογική, καλύτερη θεωρείται η υπηρεσία με την υψηλότερη διαθεσιμότητα, όπου δηλαδή γίνεται max(Cmax- Ccur).

Απαραίτητο κριτήριο στην επιλογή υπηρεσίας είναι ένα νέο αίτημα να μην επηρεάζει της δραστηριότητες των υπάρχοντων πελατών, ώστε να μην απαιτείται επανοργάνωση των πόρων [25]. Έτσι η υπηρεσία που επιλέγεται από τον broker πρέπει να είναι ανάμεσα στις διαθέσιμες, όπου δηλαδή Cmax – Ccur >0.

Μια επιπλέον παράμετρος που προστίθεται στην εφαρμογή του broker είναι η ανατροφοδότηση από τον πελάτη, η οποία είναι κρίσιμη για να καταγράφεται η ικανοποίηση του εκ του αποτελέσματος της παροχής υπηρεσίας [26]. Αφού χρησιμοποιήσει τους πόρους της Υπηρεσίας Διαδικτύου, ένας πελάτης αθροίζει την εμπειρία ποιότητας υπηρεσίας και στέλνει τις πληροφορίες στον broker. Τα δεδομένα αυτά αποθηκεύονται στη βάση δεδομένων του μεσολαβητή ως ιστορικές πληροφορίες QoS και χρησιμοποιούνται ως κριτήριο επιλογής στα επόμενα αιτήματα υπηρεσίας.

Οι εξυπηρετητές για Υπηρεσίες Διαδικτύου έχουν δυναμικό φόρτο εργασίας καθώς οι πελάτες έρχονται και φεύγουν. Το πρότυπο άφιξης των πελατών, οι ανάγκες πόρων τους και οι τρόποι χρήσης των πόρων είναι απρόβλεπτοι. Εάν ένας εξυπηρετητής αναθέτει υπερβολικούς πόρους του συστήματος σε πελάτες, νέες αιτήσεις μπορεί να μην γίνουν αποδεκτές. Μια άλλη πιθανότητα είναι ο εξυπηρετητής να μειώσει την ποιότητα υπηρεσίας ενός ή περισσότερων πελατών για να δεχτεί νέες αιτήσεις. Η μείωση ποιότητας διαταράσσει τη σταθερότητα των

ενεργειών του πελάτη και ονομάζεται αστάθεια QoS. Το περιβάλλον του συστήματος μοντελοποιήθηκε ως ένα σύνολο από πελάτες που ζητούν υπηρεσίες τυχαία, μπαίνουν στο σύστημα για να χρησιμοποιήσουν πόρους και έπειτα βγαίνουν από το σύστημα ελευθερώνοντας τους πόρους που τους είχαν διατεθεί.

Αναλυτικά, ο broker κρατάει σε μια βάση δεδομένων τα ονόματα των Υπηρεσιών Διαδικτύου, τις διευθύνσεις τους (url), τους πόρους που μπορούν να διαθέσουν για να εξυπηρετήσουν τις κλήσεις πελατών (Cmax), τους τρέχοντες διαθέσιμους πόρους (Ccur) και ένα βαθμό αξιολόγησης που αναπαριστά το αποτέλεσμα ενός γραμμικού συνδυασμού αξιολογήσεων ποιότητας (grade). Όταν ένας πελάτης κάνει μια κλήση τότε ο broker του στέλνει το όνομα και τη διεύθυνση της Υπηρεσίας Διαδικτύου που επιλέγεται με βάση τον εκάστοτε αλγόριθμο επιλογής. Ο πελάτης δεσμεύει ένα μέρος των πόρων αυτής της Υπηρεσίας Διαδικτύου και όταν εξυπηρετηθεί αποδεσμεύει τους πόρους που χρησιμοποίησε και ενημερώνει τον broker τόσο για το τρέχον υπόλοιπο διαθέσιμων πόρων όσο και για την ποιότητα υπηρεσίας που του παρείχε η Υπηρεσία Διαδικτύου.

Καθώς οι υπηρεσίες μπορεί να παρέχονται από διαφορετικούς παρόχους, επικοινωνούν στέλνοντας αιτήματα και δεδομένα στο δίκτυο. Η μεταφορά των αποτελεσμάτων από μια υπηρεσία σε μια άλλη περιλαμβάνει κάποια καθυστέρηση. Στην εφαρμογή αυτή θεωρείται ότι η καθυστέρηση μεταφοράς και το κόστος μεταξύ δυο υπηρεσιών είναι προκαθορισμένο και σταθερό.

Για να επιβεβαιωθεί και να εξεταστεί η λειτουργικότητα του broker πραγματοποιήθηκε σειρά πειραμάτων με τέσσερεις πολιτικές επιλογής και αξιολογήθηκαν τα αποτελέσματα. Συγκεκριμένα, για την επιλογή της κατάλληλης Υπηρεσίας Διαδικτύου υλοποιήθηκαν και δοκιμάστηκαν οι παρακάτω πολιτικές:

- Πολιτική Επιλογής 1: Επιλογή με βάση τη διαθεσιμότητα

Η Υπηρεσία Διαδικτύου που θα εξυπηρετήσει τον πελάτη επιλέγεται με βάση τη διαθεσιμότητα των Υπηρεσιών Διαδικτύου που διαχειρίζεται ο broker, επιλέγεται δηλαδή η Υπηρεσία Διαδικτύου με τη μεγαλύτερη τιμή Cmax – Ccur. Σε περίπτωση που μια κλήση αποτύχει τότε δεν επαναλαμβάνεται η προσπάθεια καθώς θεωρείται ότι δεν υπάρχει καμία Υπηρεσία Διαδικτύου που να μπορεί να εξυπηρετήσει τον πελάτη, είτε λόγο του μεγάλου φόρτου των διαθέσιμων Υπηρεσιών Διαδικτύου είτε λόγω των υψηλών απαιτήσεων του πελάτη.

- Πολιτική Επιλογής 2: Επιλογή με βάση τη διαθεσιμότητα και την ποιότητα υπηρεσίας

Στην δεύτερη πολιτική επιλογής οι πελάτες όταν χρησιμοποιήσουν μια Υπηρεσία Διαδικτύου και απελευθερώσουν τους πόρους που δέσμευσαν, στέλνουν στον broker μια αξιολόγηση της Υπηρεσίας Διαδικτύου με βαθμούς που κυμαίνονται από το 1 έως το 10. Οι βαθμοί αυτοί αθροίζονται στη βάση δεδομένων του broker. Όταν γίνεται μια κλήση, η Υπηρεσία Διαδικτύου που θα εξυπηρετήσει τον πελάτη επιλέγεται με βάση τη βαθμολογία των διαθέσιμων Υπηρεσιών Διαδικτύου που έχουν αρκετούς πόρους ώστε να εξυπηρετήσουν τον πελάτη. Σε περίπτωση που η κλήση ενός πελάτη δεν εξυπηρετηθεί απλά σημειώνεται ως αποτυχημένη.

- Πολιτική Επιλογής 3: Διαχείριση λαθών της πολιτικής 2 με μείωση των απαιτήσεων του πελάτη

Η τρίτη πολιτική επιλογής είναι μια επέκταση της 2 που διαχειρίζεται την περίπτωση λάθους επιστρέφοντας τις Υπηρεσίες Διαδικτύου που μπορούν να καλύψουν το 75% των διαθέσιμων πόρων που απαιτεί ο πελάτης. Με αυτόν τον τρόπο μειώνεται η ποιότητα υπηρεσίας που λαμβάνει ο πελάτης.

- Πολιτική Επιλογής 4: Διαχείριση λαθών της πολιτικής 2 με πολιτική κρατήσεων

Η τέταρτη πολιτική επιλογής διαχειρίζεται την περίπτωση λάθους της πολιτικής 2 ακολουθώντας μια μέθοδο κρατήσεων. Όταν η κλήση ενός πελάτη αποτύχει, δηλαδή δε βρεθεί διαθέσιμη Υπηρεσία Διαδικτύου που να μπορεί να τον εξυπηρετήσει, τότε επιλέγεται η Υπηρεσία Διαδικτύου που έχει τη μεγαλύτερη βαθμολογία και τη μεγαλύτερη διαθεσιμότητα από τις Υπηρεσίες Διαδικτύου που οι συνολικοί διαθέσιμοι πόροι τους είναι περισσότεροι από αυτούς

που απαιτεί ο πελάτης. Αυτή η Υπηρεσία Διαδικτύου δεσμεύεται υπό την έννοια ότι δε δίνεται σε άλλους πελάτες εωσότου τις επιστραφούν πόροι και μπορέσει να εξυπηρετήσει τον πελάτη.

10.1. Υλοποίηση με χρήση του WSRF.NET

Σε μεγάλο βαθμό, η αλληλεπίδραση με το σύστημα WSRF.NET έγινε μέσω των .NET ιδιοτήτων. Αυτές οι ιδιότητες επιτρέπουν στο WSRF.NET να ελέγξει διάφορες πλευρές των component Υπηρεσιών Διαδικτύου όπως η δημιουργία WSDL, η ονομασία, η διαχείριση καταστάσεων και η συνάθροιση των port types. Για παράδειγμα, η ιδιότητα [WsdlBaseName] περιγράφει στο WSRF.NET το base name που θα χρησιμοποιήσει όταν δημιουργεί WSDL ονόματα όπως επίσης και το namespace στο οποίο τοποθετεί τα στοιχεία WSDL.

Το WSRF.NET αποθηκεύει τα WS Resources σε μια βάση δεδομένων και οι σχετικοί πόροι φορτώνονται αυτόματα από μια WSRF.NET Υπηρεσία Διαδικτύου όταν φτάνει μια κλήση. Η προδιαγραφή WS-ResourceProperties περιλαμβάνει ένα σύνολο από port types που μπορούν να χρησιμοποιηθούν από πελάτες για να διαβάσουν και να διαχειριστούν Resource Properties σε μια υπηρεσία. Συγκεκριμένα είναι δυνατό τα δεδομένα αυτών των port types να ανακτηθούν, να ενημερωθούν, να προστεθούν, να διαγραφούν τιμές και να ερωτηθεί μια υπηρεσία για το όποιες ιδιότητες ταιριάζουν με συγκεκριμένα πρότυπα.

Η ειδοποίηση (Notification) είναι ο τρόπος με τον οποίο οι πελάτες μπορούν να εγγραφούν για να λαμβάνουν ασύγχρονα μηνύματα σχετικά με επιθυμητά θέματα. Η προδιαγραφή WS-Notification ορίζει το port type NotificationProducer για τη διαχείριση της διαδικασίας subscription. Ο broker γράφεται συνδρομητής στον Notification Producer όταν γίνεται μια νέα επιτυχής κλήση πελάτη για να λάβει ειδοποίηση σχετικά με τους διαθέσιμους πόρους μιας υπηρεσίας. Η υλοποίηση του broker περιλαμβάνει ένα server thread που λαμβάνει ασύγχρονα μηνύματα ειδοποίησης χρησιμοποιώντας την κλάση AsynchronousNotificationListener του WSRF.NET.

Τα ServiceGroups στο WSRF είναι ένας μηχανισμός μέσω του οποίου πολλαπλά WS-Resources μπορούν να ομαδοποιηθούν σε λογικές συλλογές. Αυτές οι ομάδες μπορούν να ερωτηθούν μέσω διάφορων XPath δηλώσεων για την ανακάλυψη μελών (ή πληροφοριών σχετικά με μέλη) της ομάδας. Στην εφαρμογή αυτή ο μεσολαβητής επιλέγει από μια κλάση υπηρεσιών με την ίδια λειτουργικότητα τις κατάλληλες Υπηρεσίες Διαδικτύου που θα εξυπηρετήσουν τα αιτήματα των πελατών. Αυτό υλοποιείται με τη χρήση των Service Groups που δημιουργούν συλλογές από ενεργές Υπηρεσίες Διαδικτύου.

Στην πειραματική εφαρμογή, ο broker επιλέγει την κατάλληλη Υπηρεσία Διαδικτύου με βάση τον αλγόριθμο επιλογής. Τα WS-Resources του broker αναπαριστούν τις Υπηρεσίες Διαδικτύου. Κάθε WS-Resource περιέχει πληροφορίες για την τρέχουσα διαθεσιμότητα της Υπηρεσία Διαδικτύου, πληροφορίες για τον τρέχον πελάτη και τους πόρους που δέσμευσε από την Υπηρεσία Διαδικτύου. Η καταστροφή ενός WS-Resource αναπαριστά την αφαίρεση της Υπηρεσίας Διαδικτύου από την κλάση υπηρεσιών.

10.2. Πειραματικά αποτελέσματα

Στη συνέχεια παρουσιάζονται τα πειραματικά αποτελέσματα σε μορφή γραφημάτων. Τα γραφήματα αυτά ανήκουν σε δύο κατηγορίες :

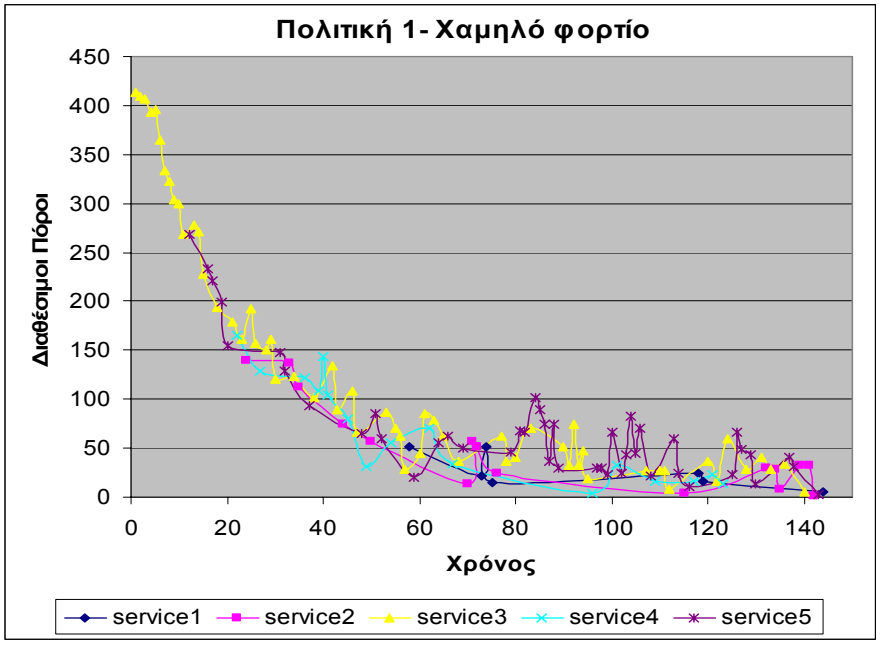
- Γραφήματα χωρητικότητας (διαθέσιμων πόρων συστήματος) των Υπηρεσιών Διαδικτύου που διαχειρίζεται ο broker ως προς το χρόνο, για κάθε μια από τις πολιτικές επιλογής υπηρεσίας
- Γραφήματα σύγκρισης των πολιτικών επιλογής, όπου παρουσιάζεται ο ρυθμός εξυπηρέτησης πελατών, δηλαδή ο αριθμός των πελατών που εξυπηρετήθηκαν ως προς το χρόνο

Η αρχική διαθεσιμότητα των Υπηρεσιών Διαδικτύου φαίνεται στον παρακάτω πίνακα

Όνομα Υπηρεσίας Διαδικτύου	Μέγιστη Χωρητικότητα
service1	61
service2	166
service3	423
service4	194
service5	289

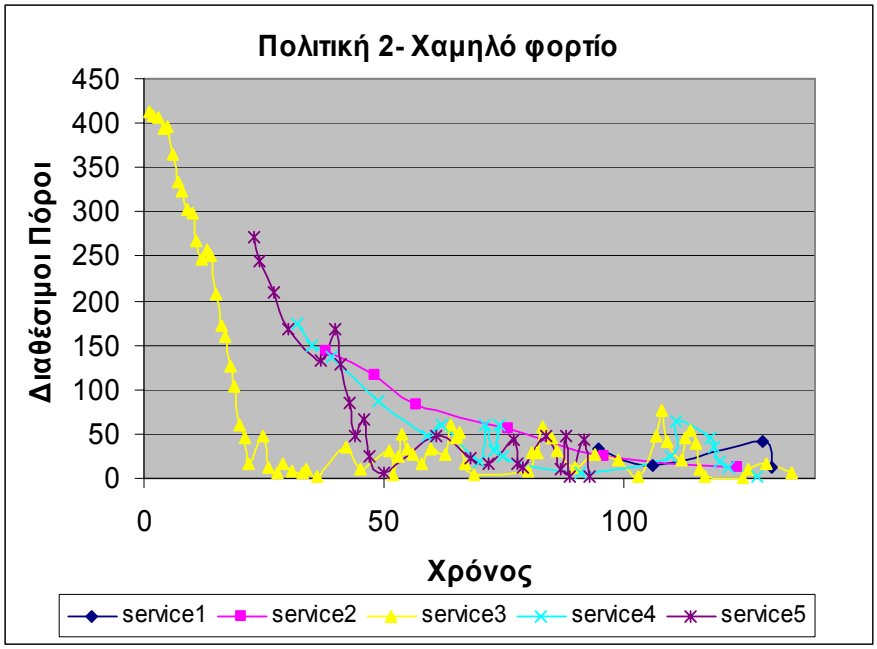
10.2.1. Γραφήματα πρώτου συνόλου πειραμάτων με δεδομένα εισόδου για εξυπηρέτηση πελατών με χαμηλές απαιτήσεις χωρητικότητας.

Το Γράφημα 1 παρουσιάζει τη διαθεσιμότητα των υπηρεσιών καθώς οι πελάτες δεσμεύουν και απελευθερώνουν πόρους με βάση την πολιτική επιλογής 1 (Επιλογή με βάση τη διαθεσιμότητα). Η Υπηρεσία Διαδικτύου service1 που έχει τη μεγαλύτερη διαθεσιμότητα πόρων χρησιμοποιείται αρχικά για όλες τις κλήσεις πελατών έως ότου εξισωθεί με τη δεύτερη μεγαλύτερη. Στη συνέχεια ανάλογα με τις δεσμεύσεις και αποδεσμεύσεις πόρων από τους πελάτες επιλέγεται κάθε φορά η υπηρεσία με τη μεγαλύτερη διαθεσιμότητα.



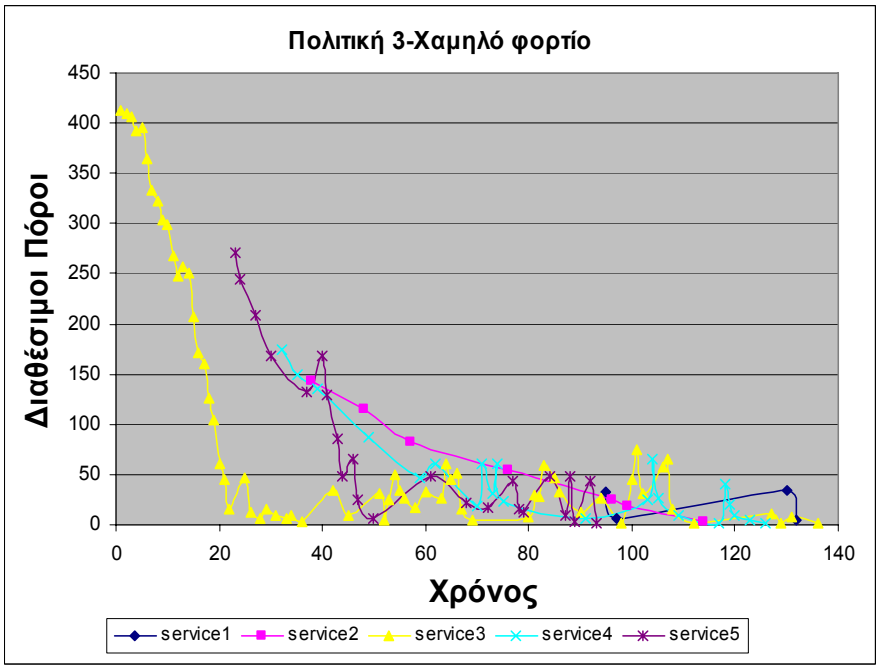
Γράφημα 1:Πολιτική 1-Χαμηλό φορτίο

Το Γράφημα 2 παρουσιάζει τη διαθεσιμότητα των υπηρεσιών καθώς οι πελάτες δεσμεύουν και απελευθερώνουν πόρους με βάση την πολιτική επιλογής 2 (Επιλογή με βάση τη διαθεσιμότητα και την ποιότητα υπηρεσίας). Η γραφική παράσταση σε αυτήν την περίπτωση διαφέρει από την προηγούμενη καθώς στον αλγόριθμο επιλογής συμβάλει και η παράμετρος αξιολόγησης (grade)



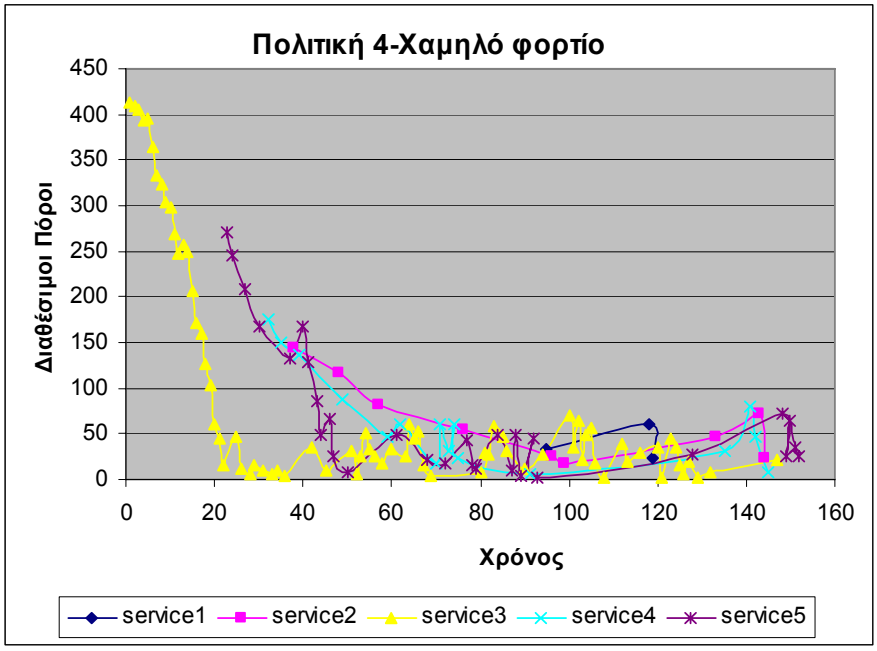
Γράφημα 2:Πολιτική 2-Χαμηλό φορτίο

Το Γράφημα 3 παρουσιάζει τη διαθεσιμότητα των υπηρεσιών καθώς οι πελάτες δεσμεύουν και απελευθερώνουν πόρους με βάση την πολιτική επιλογής 3 (Επιλογή με βάση τη διαθεσιμότητα και την ποιότητα υπηρεσίας και διαχείριση λαθών με μείωση των απαιτήσεων του πελάτη) .



Γράφημα 3: Πολιτική 3-Χαμηλό φορτίο

Το Γράφημα 4 παρουσιάζει τη διαθεσιμότητα των υπηρεσιών καθώς οι πελάτες δεσμεύουν και απελευθερώνουν πόρους με βάση την πολιτική επιλογής 4 (Επιλογή με βάση τη διαθεσιμότητα και την ποιότητα υπηρεσίας και διαχείριση λαθών με πολιτική κρατήσεων).



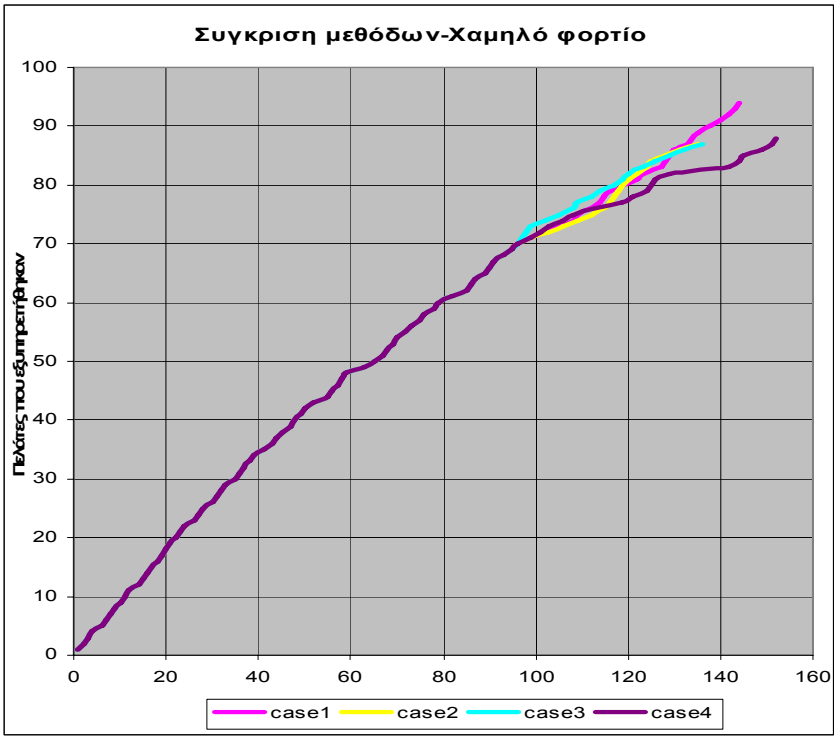
Γράφημα 4: Πολιτική 4-Χαμηλό φορτίο

Συγκεντρωτικά τα αποτελέσματα των πειραμάτων με δεδομένα πελατών χαμηλών απαιτήσεων χωρητικότητας είναι τα εξής:

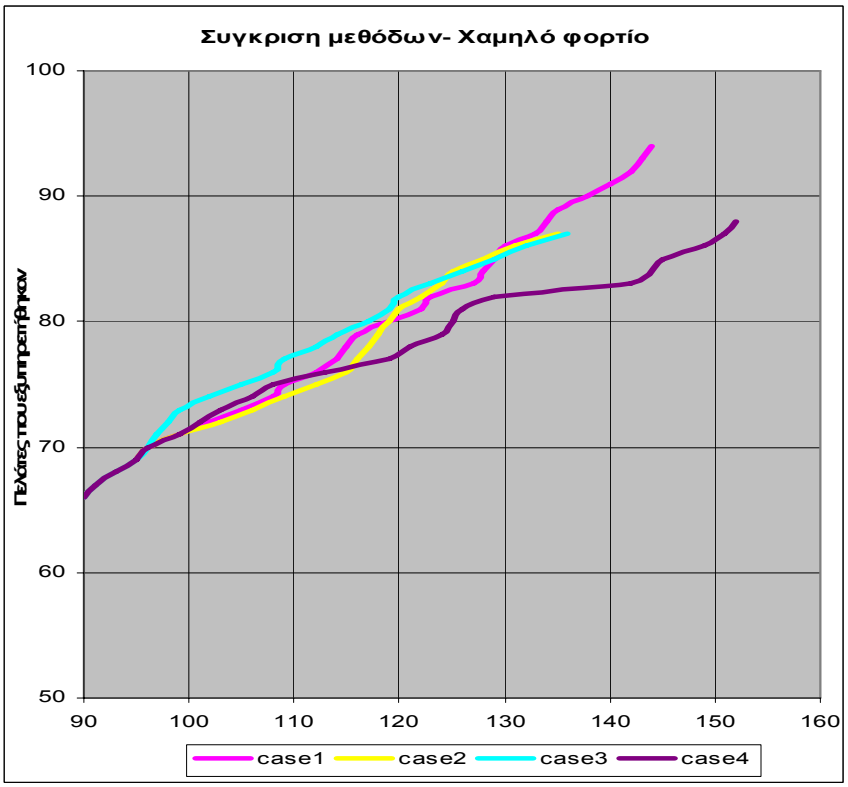
Πολιτική επιλογής	Αποτυχημένες κλήσεις	Επιτυχείς κλήσεις	Χρόνος	Μειωμένη ποιότητα	Αριθμός αποτυχιών
Case1	6	94	144		6
Case2	13	87	135		13
Case3	13	87	136	5	13
Case4	12	88	152		19

Η στήλη «Μειωμένη ποιότητα» περιέχει τον αριθμό των πελατών που εξυπηρετήθηκαν με μειωμένη ποιότητα (πολιτική επιλογής 3), η στήλη «Αριθμός αποτυχιών» είναι ο αριθμός των αποτυχημένων προσπαθειών για εξυπηρέτηση.εξυπηρετήθηκαν με μειωμένη ποιότητα (πολιτική επιλογής 3) και η στήλη «Χρόνος» είναι οι κύκλοι ρολογιού. Παρατηρούμε ότι όταν εισάγεται η παράμετρος αξιολόγησης της ποιότητας υπηρεσίας από τον πελάτη, τότε ικανοποιούνται λιγότεροι πελάτες καθώς υπάρχει συμφόρηση κίνησης στις υπηρεσίες με τη μεγαλύτερη βαθμολόγηση.

Στα γραφήματα 5 και 6 βλέπουμε ότι αρχικά και οι τέσσερις πολιτικές έχουν την ίδια συμπεριφορά μέχρι τη χρονική στιγμή «96» όπου και αρχίζουν να διαφοροποιούνται.



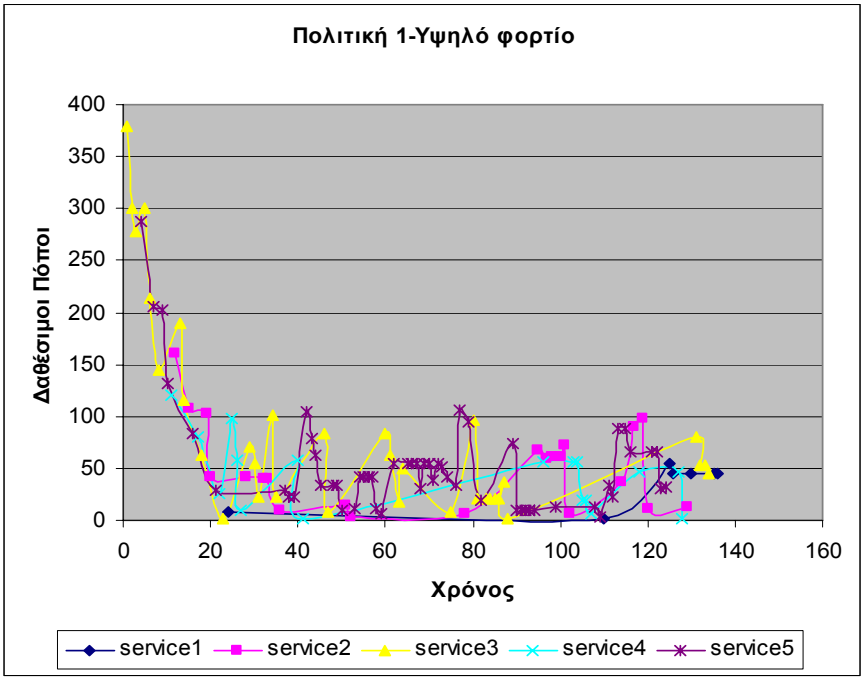
Γράφημα 5: Σύγκριση μεθόδων –Χαμηλό φορτίο



Γράφημα 6:Σύγκριση μεθόδων-Χαμηλό φορτίο(μεγαλύτερη ανάλυση)

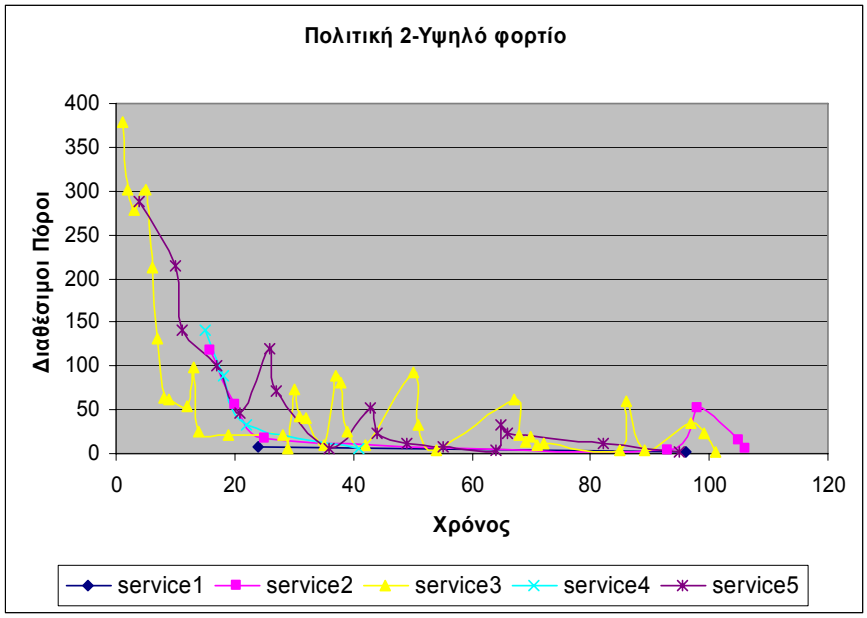
10.2.2. Γραφήματα του δεύτερου συνόλου πειραμάτων με δεδομένα εισόδου για εξυπηρέτηση πελατών με υψηλές απαιτήσεις χωρητικότητας.

Το Γράφημα 7 παρουσιάζει τη διαθεσιμότητα των υπηρεσιών καθώς οι πελάτες δεσμεύουν και απελευθερώνουν πόρους με βάση την πολιτική επιλογής 1 (Επιλογή με βάση τη διαθεσιμότητα).



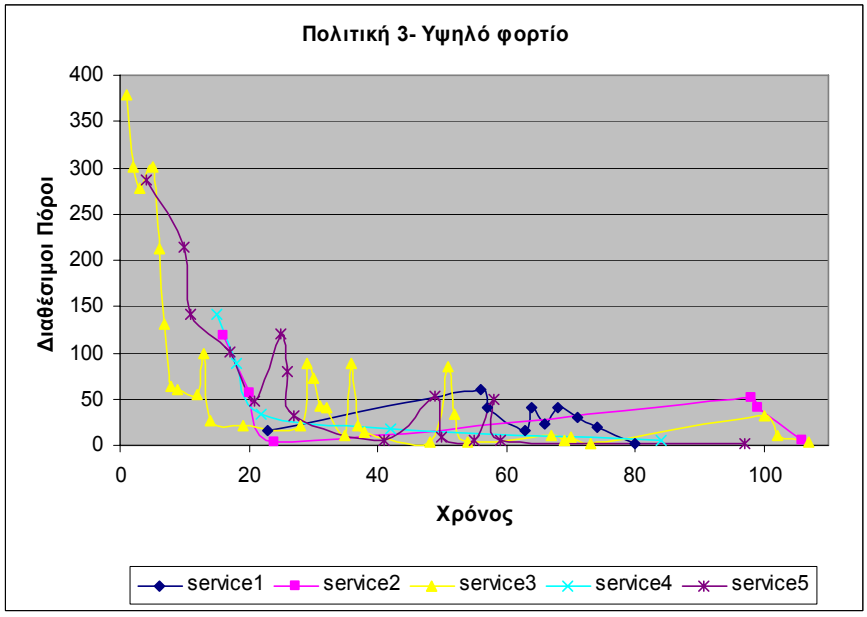
Γράφημα 7: Πολιτική 1-Υψηλό φορτίο

Το Γράφημα 8 παρουσιάζει τη διαθεσιμότητα των υπηρεσιών καθώς οι πελάτες δεσμεύουν και απελευθερώνουν πόρους με βάση την πολιτική επιλογής 2 (Επιλογή με βάση τη διαθεσιμότητα και την ποιότητα υπηρεσίας).



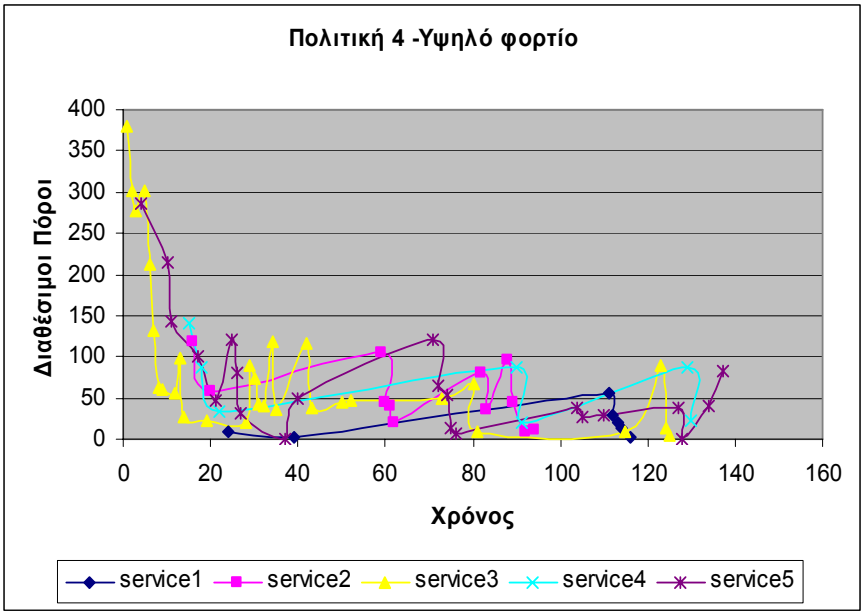
Γράφημα 8:Πολιτική 2-Υψηλό φορτίο

Το Γράφημα 9 παρουσιάζει τη διαθεσιμότητα των υπηρεσιών καθώς οι πελάτες δεσμεύουν και απελευθερώνουν πόρους με βάση την πολιτική επιλογής 3 (Επιλογή με βάση τη διαθεσιμότητα και την ποιότητα υπηρεσίας και διαχείριση λαθών με μείωση των απαιτήσεων του πελάτη) .



Γράφημα 9: Πολιτική 3-Υψηλό φορτίο

Το Γράφημα 10 παρουσιάζει τη διαθεσιμότητα των υπηρεσιών καθώς οι πελάτες δεσμεύουν και απελευθερώνουν πόρους με βάση την πολιτική επιλογής 4 (Επιλογή με βάση τη διαθεσιμότητα και την ποιότητα υπηρεσίας και διαχείριση λαθών με πολιτική κρατήσεων).



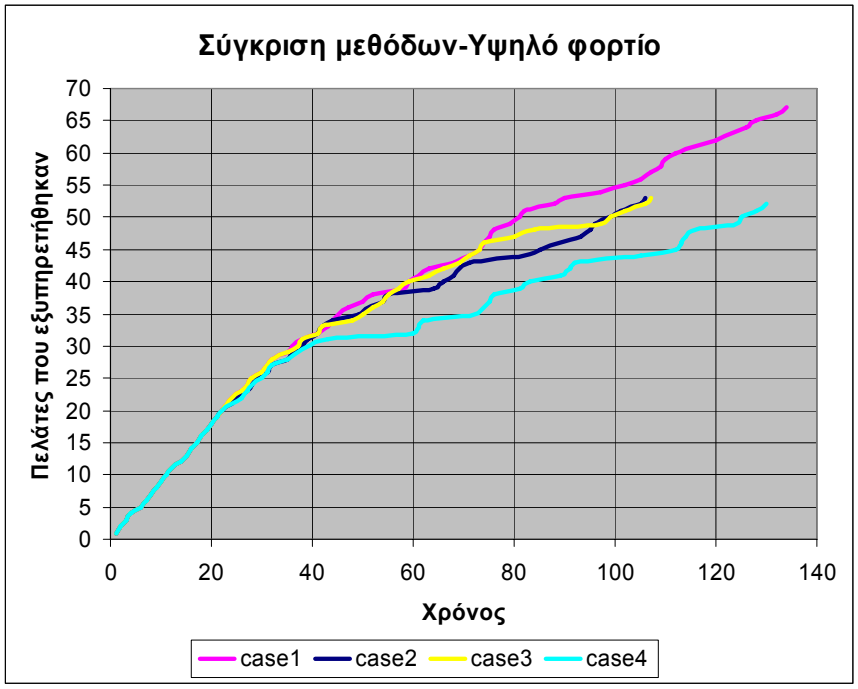
Γράφημα 10: Πολιτική 4-Υψηλό φορτίο

Συγκεντρωτικά τα αποτελέσματα των πειραμάτων με δεδομένα πελαών υψηλών απαιτήσεων χωρητικότητας είναι τα εξής:

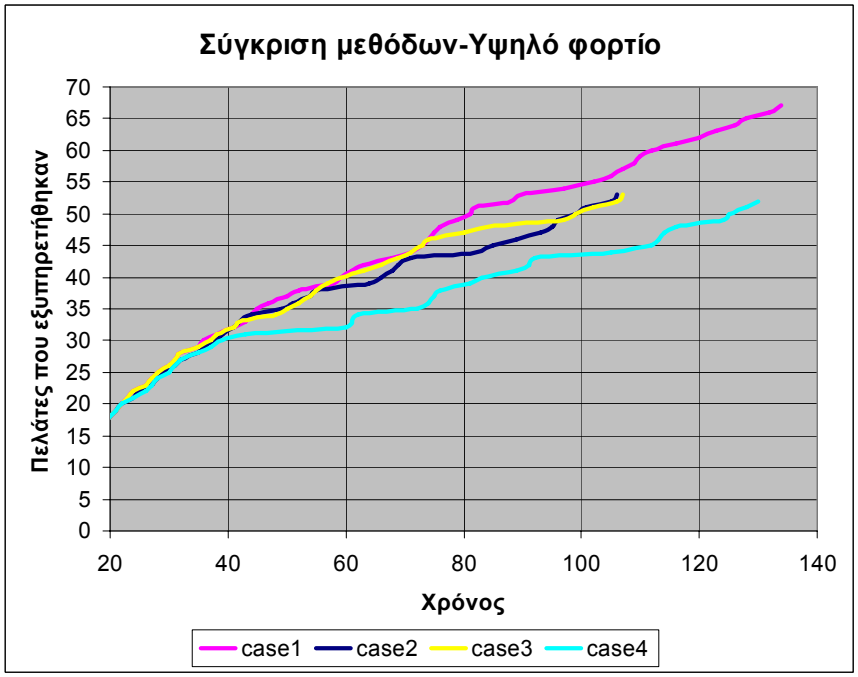
Πολιτική επιλογής	Αποτυχημένες κλήσεις	Επιτυχείς κλήσεις	Χρόνος	Μειωμένη ποιότητα	Αριθμός αποτυχιών
Case1	33	67	136		33
Case2	47	53	114		47
Case3	47	53	115	5	47
Case4	48	52	139		62

Η στήλη «Μειωμένη ποιότητα» περιέχει τον αριθμό των πελατών που εξυπηρετήθηκαν με μειωμένη ποιότητα (πολιτική επιλογής 3), η στήλη «Αριθμός αποτυχιών» είναι ο αριθμός των αποτυχημένων προσπαθειών για εξυπηρέτηση.εξυπηρετήθηκαν με μειωμένη ποιότητα (πολιτική επιλογής 3) και η στήλη «Χρόνος» είναι οι κύκλοι ρολογιού.

Στα γραφήματα 11 και 12 βλέπουμε ότι αρχικά και οι τέσσερις πολιτικές έχουν την ίδια συμπεριφορά μέχρι τη χρονική στιγμή «20» όπου και αρχίζουν να διαφοροποιούνται, πολύ πιο γρήγορα δηλαδή από ότι στα πειράματα με χαμηλό φορτίο.



Γράφημα 11: Σύγκριση μεθόδων-Υψηλό φορτίο



Γράφημα 12: Σύγκριση μεθόδων-Υψηλό φορτίο(μεγαλύτερη ανάλυση)

Συμπερασματικά, τα πειραματικά αποτελέσματα επιβεβαιώνουν τη λειτουργικότητα του WSRF.NET Broker. Οι διάφορες πολιτικές επιλογής που εφαρμόστηκαν λειτούργησαν αναμενόμενα όπως φάνηκε στην ανάλυση των γραφικών παραστάσεων. Σε πραγματικό περιβάλλον επιλογής η εφαρμογή του μεσολαβητή επηρεάζεται αρκετά από τη διαδικασία υπολογισμού της ποιότητας υπηρεσίας μιας Υπηρεσίες Διαδικτύου. Τέτοιες διαδικασίες παρουσιάστηκαν σε προηγούμενα κεφάλαια μαζί με τα πειραματικά τους αποτελέσματα.

11. Βιβλιογραφία

[1] W3C (2003) Simple Object Access Protocol (SOAP), http://www.w3.org/TR/soap
[2] W3C (2003) Web Services Description Language (WSDL), http://www.w3.org/TR/wsdl
[3] W3C (2003) Universal description, discovery, and integration (UDDI). http://www.uddi.org
[4] W3C (2003) SemanticWeb. http://www.w3.org/2001/sw
[5] UDDI Organization (ed.): UDDI Technical White Paper. UDDI Standards Organization Public Draft (2000) http://www.uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf
[6] On Specifying Web Services Using UDDI Improvements, Sven Overhage
[7] Councill, B., Heineman, G. T.: Definition of a Software Component and Its Elements. In: Councill, W. T., Heineman, G. T. (eds.): Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey (2001): 5–19
[8] A Model for Web Services Discovery With QoS, SHUPING RAN, ACM 2003
[9] CLARK,2001.UDDI weather report. http://www.webservicesarchitect.com/content/articles/clark04.asp
[10] UDDI COMMITTEE. 2002. UDDI Version 2.03 Data Structure Reference, http://www.uddi.org/pubs/DataStructure_v2.htm.
[11] Algorithm for Web Services Matching, Atul Sajjanhar, Jingyu Hou, and Yanchun Zhang
[12] Composing Web services on the SemanticWeb, Brahim Medjahed, Athman Bouguettaya, Ahmed K. Elmagarmid
[13] Medjahed B, Benatallah B, BouguettayaA, NguA, Elmagarmid, Business-to-business interactions: issues and enabling technologies. VLDB J 12(1):59–85
[14] Berners-LeeT(2001) Services and semantics: Web architecture, http://www.w3.org/2001/04/30-tbl
[15] The Design of QoS Broker Algorithms for QoS-Capable Web Services, Tao Yu and Kwei-Jay Lin
[16] J. P. Hansen, J. Lehoczky and R. Rajkumar"Optimization of Quality of Service in DynamicSystems" in Proc. of the 9th InternationalWorkshop on Parallel and Distributed Real-Time Systems, April 2001
[17] High Availability with Clusters of Web Services, Julio F.Vilas, José Pazos Arias, Ana F.Vilas
[18] Rajkumar Buyya, “High Performance Cluster Computing: Architectures and Systems”,Prentice Hall, 1999
[19] Quality-Oriented Handling of Exceptions in Web-Service-Based Cooperative Processes, Ulrike Greiner, Erhard Rahm
[20] Keidl, M., Seltzsam, S., Kemper, A.: Reliable Web Service Execution and Deployment in Dynamic Environments. In: Benatallah, B., Shan, M-C. (eds.): TES 2003. Springer-Verlag, Berlin Heidelberg (2003) 104-118
[21] Efficient Access to Web Services, Mourad Ouzzani , Athman Bouguettaya, IEEE INTERNET COMPUTING
[22] B. Medjahed, A. Bouguettaya, and A. Elmagarmid, “Composing Web Services on the Semantic Web,” J. Very Large Databases, vol. 12, no. 4, Nov. 2003, pp. 333–351.

[23] The WS-Resource Framework

[24] WSRF.NET 1.1 Programmer's Reference , Glenn Wasson

[25] Service Selection Algorithms for Web Services with End –to End Constraints, TaoYu, Kwei , Jay Lin, IEEEInternational Conference on E-Commerce Technology, 2004

[26] Liu, Y., Ngu, A., and Zeng, L. QoS Computation and Policing in Dynamic Web Service Selection. In proc. World Wide Web Conference (WWW2004), pp. 66 - 73, 2004

12. ΠΑΡΑΡΤΗΜΑ Α

Τα κύρια businessEntity πεδία δεδομένων

businessKey	Αυτή η ιδιότητα είναι το μοναδικό προσδιοριστικό για ένα στιγμιότυπο μιας δομής businessEntity.
authorizedName	Αυτή η ιδιότητα είναι το καταγραμμένο όνομα του ατόμου που δημοσίευσε τα στοιχεία businessEntity. Η τιμή του υπολογίζεται από τον λειτουργικό κατάλογο και δεν πρέπει να παρέχεται από τη συνάρτηση save_business API.
operator	Αυτή η ιδιότητα είναι το όνομα του καταλόγου UDDI που διαχειρίζεται τα στοιχεία businessEntity. Ο κατάλογος καταγράφει αυτά τα δεδομένα όταν αποθηκεύονται και δεν πρέπει να παρέχεται μέσα στη συνάρτηση save_business.
discoveryURLs	Αυτό το προαιρετικό στοιχείο είναι μια λίστα των URLs που δείχνουν σε έναν μηχανισμό ανακαλύψεων βασισμένο σε αρχεία. Σε κάθε καταγραμμένο businessEntity ορίζεται αυτόματα ένα URL που επιστρέφει τη δομή businessEntity σε μια XML αναπαράσταση.
name	Αυτό το απαραίτητο στοιχείο είναι το όνομα που καταγράφεται για την οντότητα. Η κλήση find_business API παρέχει μια αναζήτηση businessEntity με το όνομά της.
description	Αυτό είναι ένα προαιρετικό στοιχείο επανάληψης για την παροχή μηδενικών ή πιο μικρών κειμένων περιγραφής της επιχείρησης.
contacts	Αυτό το προαιρετικό στοιχείο είναι ένας κατάλογος πληροφοριών επικοινωνίας.
businessServices	Αυτό το προαιρετικό στοιχείο είναι ένας κατάλογος μιας ή περισσότερων περιγραφών επιχειρησιακών υπηρεσιών. Εάν δεν εγγράφεται καμία υπηρεσία με το businessEntity για ένα δεδομένο χρονικό διάστημα, ο κατάλογος μπορεί να διαγράψει το businessEntity κατά την εκτέλεση μιας λειτουργίας καθαρισμού.
identifierBag	Αυτό το προαιρετικό στοιχείο είναι ένας κατάλογος ζευγαριών ονόματος-τιμής που μπορεί να χρησιμοποιηθεί για την αποθήκευση αριθμών αναγνώρισης για ένα businessEntity. Αυτοί οι αριθμοί μπορούν να χρησιμοποιηθούν για την αναζήτηση μέσω της συνάρτησης find_business API.
categoryBag	Αυτό το προαιρετικό στοιχείο είναι ένας κατάλογος ζευγαριών ονόματος-τιμής που χρησιμοποιείται για την επικόλληση του businessEntity με συγκεκριμένες πληροφορίες ταξονομίας. Αυτά μπορούν να χρησιμοποιηθούν για έρευνα μέσω της συνάρτησης find_business API.

Τα businessService πεδία δεδομένων

businessKey	Αυτή η ιδιότητα αναφέρεται στο businessEntity που παρέχει την υπηρεσία.
serviceKey	Αυτή η ιδιότητα είναι το μοναδικό προσδιοριστικό για ένα δεδομένο businessService. Αυτή η τιμή δεν πρέπει να παρασχεθεί όταν αποθηκεύεται μια νέα δομή businessService, η οποία δηλώνει ότι πρέπει να παραχθεί ένα UUID για εκείνη την οντότητα. Κατά την ενημέρωση

	μιας υπάρχουσας δομής businessService, πρέπει να παρασχεθεί η ήδη υπάρχουσα τιμή κλειδι.
name	Αυτή η τιμή είναι ένα αναγνώσιμο από τον άνθρωπο όνομα για την υπηρεσία.
description	Αυτό είναι ένα προαιρετικό στοιχείο επανάληψης που παρέχει μηδενικές ή περισσότερες περιγραφές κειμένου για την υπηρεσία.
bindingTemplates	Αυτό το στοιχείο είναι ένας κατάλογος μιας ή περισσότερων πληροφοριών για τεχνική περιγραφή των υπηρεσιών που αναπαριστούνται από τις δομές bindingTemplate.
categoryBag	Αυτό το προαιρετικό στοιχείο είναι ένας κατάλογος ζευγαριών ονόματος-τιμής που μπορεί να χρησιμοποιηθεί για την εγγραφή συγκεκριμένων πληροφοριών ταξονομίας για ένα businessService. Μπορεί επίσης να χρησιμοποιηθεί για αναζήτηση μέσω της συνάρτησης find_service API.

Τα bindingTemplate πεδία δεδομένων

bindingKey	Αυτό είναι το μοναδικό κλειδί για ένα δεδομένο bindingTemplate. Κατά την αποθήκευση ενός νέου bindingTemplate, δεν πρέπει να παρασχεθεί μια τιμή bindingKey συνεπώς αυτό παράγεται αυτόματα από τον κατάλογο. Για την ενημέρωση μιας υπάρχουσας bindingTemplate, πρέπει να παρασχεθεί το υπάρχον κλειδί του.
serviceKey	Αυτή η ιδιότητα είναι το κλειδί για την αντίστοιχη υπηρεσία και ενεργεί ως αναφορά σε αυτήν.
description	Αυτό το προαιρετικό στοιχείο επανάληψης παρέχει μηδενικές ή περισσότερες περιγραφές κειμένου για τα χαρακτηριστικά του bindingTemplate.
accessPoint	Αυτή η ιδιότητα είναι ένα πεδίο κειμένου που χρησιμοποιείται για να μεταβιβάσει τη διεύθυνση του σημείου εισόδου που είναι κατάλληλο για να πάρει πληροφορίες για μια συγκεκριμένη υπηρεσία. Αυτό μπορεί να είναι ένα URL, αλλά και μια διεύθυνση ηλεκτρονικού ταχυδρομείου ή ακόμα και ένας αριθμός τηλεφώνου. Στην περίπτωση που είναι διαθέσιμη μια περιγραφή WSDL για την υπηρεσία, η θέση διευθύνσεών του (όπως ένα URL) μπορεί να παρασχεθεί σε αυτήν την θέση.
hostingRedirector	Αυτό είναι ένα απαραίτητο στοιχείο εάν δεν παρέχεται η ιδιότητα accessPoint. Είναι μια επαναπροσανατολισμένη αναφορά σε ένα διαφορετικό bindingTemplate. Αυτό το στοιχείο χρησιμοποιείται για να υποδείξει ότι μια είσοδος bindingTemplate είναι ένας δείκτης σε μια διαφορετική είσοδο bindingTemplate. Αυτό θα ήταν χρήσιμο ώστε να ωφεληθούν πολλές περιγραφές υπηρεσιών από μια ενιαία περιγραφή υπηρεσιών. Το στοιχείο hostingRedirector περιέχει μια ενιαία ιδιότητα που κρατά την τιμή bindingKey από το διαφορετικό bindingTemplate στο οποίο αναφέρεται.
tModelInstanceDetails	Αυτό το προαιρετικό στοιχείο είναι ένας κατάλογος στοιχείων tModelInstanceInfo. Κάθε tModelInstanceInfo δείχνει οποιαδήποτε προδιαγραφή (που αντιπροσωπεύεται από ένα tModel) στην οποία αντιστοιχεί αυτό το bindingTemplate. Κατά τη διάρκεια της αναζήτησης για μια υπηρεσία, ένα ενδιαφερόμενο συμβαλλόμενο

	μέρος θα μπορούσε να χρησιμοποιήσει αυτές τις πληροφορίες για να ψάξει για ένα συγκεκριμένο bindingTemplate που προσαρμόζεται στις απαιτούμενες προδιαγραφές, οι οποίες εκφράζονται μέσω του στοιχείου tModelInstanceDetails από την ύπαρξη της αντίστοιχης αναφοράς tModel.

Τα πεδία δεδομένων tModel

tModelKey	Αυτή η απαραίτητη ιδιότητα είναι το μοναδικό προσδιοριστικό για μια δεδομένη δομή tModel. Όταν αποθηκεύεται μια νέα δομή tModel, αυτή η τιμή δεν πρέπει να παρασχεθεί για να δείξει ότι πρόκειται να παραχθεί μια τιμή UUID από τον κατάλογο. Για την ενημέρωση ενός υπάρχοντος tModel, πρέπει να περαστεί η ήδη υπάρχουσα τιμή κλειδί.
authorizedName	Αυτή η ιδιότητα είναι το καταγραμμένο όνομα του συμβαλλόμενου μέρους που δημοσίευσε τα δεδομένα tModel. Η τιμή της υπολογίζεται από τον τρέχων κατάλογο και δεν πρέπει να παρέχεται από τη συνάρτηση save_tModel API.
operator	Αυτή η ιδιότητα είναι το όνομα του καταλόγου UDDI που διαχειρίζεται τα δεδομένα tModel. Ο κατάλογος καταγράφει αυτόματα μια τιμή στο χρόνο που σώζεται το δεδομένο tModel.
name	Αυτό το απαραίτητο στοιχείο αντιπροσωπεύει το όνομα που καταγράφεται για το tModel. Η αναζήτηση ονόματος παρέχεται από την κλήση της find_tModel API.
description	Αυτό το προαιρετικό στοιχείο επανάληψης μπορεί να χρησιμοποιηθεί για να παρέχει μηδενικές ή πιο μικρές περιγραφές κειμένου για το tModel.
overviewDoc	Αυτό το προαιρετικό στοιχείο χρησιμοποιείται για να κρατήσει τις αναφορές στις περιγραφές ή στις οδηγίες για τη χρήση του tModel .
identifierBag	Αυτό το προαιρετικό στοιχείο είναι ένας κατάλογος ζευγαριών ονόματος-τιμής που μπορεί να χρησιμοποιηθεί για να καταγράφει αριθμούς προσδιορισμού για ένα tModel. Αυτοί μπορούν να χρησιμοποιηθούν κατά τη διάρκεια της αναζήτησης μέσω της συνάρτησης find_tModel API.
categoryBag	Αυτό το προαιρετικό στοιχείο είναι ένας κατάλογος ζευγαριών ονόματος-τιμής που μπορεί να χρησιμοποιηθεί για την επικόλληση ενός tModel με συγκεκριμένες πληροφορίες ταξονομίας. Αυτοί μπορούν να χρησιμοποιηθούν για αναζήτηση μέσω της συνάρτησης find_tModel API.

Namespaces (με τα αντίστοιχα προθέματά τους) που χρησιμοποιούνται αυτήν την περίοδο στα έγγραφα WSDL.

Πρόθεμα	Namespace URI	Ορισμός
wsdl	http://schemas.xmlsoap.org/wsdl/	WSDL namespace για το πλαίσιο εργασίας WSDL
soap	http://schemas.xmlsoap.org/wsdl/soap/	WSDL namespace για WSDL SOAP σύνδεση
http	http://schemas.xmlsoap.org/wsdl/http/	WSDL namespace για WSDL HTTP σύνδεση
mime	http://schemas.xmlsoap.org/wsdl/mime/	WSDL namespace για WSDL MIME

		σύνδεση
soapenc	http://schemas.xmlsoap.org/wsdl/encoding/	Κωδικοποίηση του namespace
soapenv	http://schemas.xmlsoap.org/soap/envelope/	Φακέλωμα του namespace όπως ορίζεται στο [SOAP11]
xsi	http://www.w3.org/1999/XMLSchema-instance	XML namespace σχηματικά παραδείγματα
	http://www.w3.org/2000/10/XMLSchema-instance	
xsd	http://www.w3.org/1999/XMLSchema	XML namespace σχηματικοί τύποι δεδομένων
	http://www.w3.org/2000/10/XMLSchema	
tns	various	Το 'this namespace' πρόθεμα χρησιμοποιείται ως δρομολόγηση για παραπομπή στο τρέχων έγγραφο.
other	various	Όλα τα υπόλοιπα προθέματα είναι μόνο δείγματα ή ειδικά για ιδιόμορφα έγγραφα.

Namespaces για WSDL έγγραφα

Πρόθεμα	Namespace URI	Καθορισμός
SOAP-ENC	http://schemas.xmlsoap.org/soap/encoding/	namespace κωδικοποίησης SOAP
SOAP-ENV	http://schemas.xmlsoap.org/soap/envelope/	namespace φακέλου SOAP
xsi	http://www.w3.org/1999/XMLSchemainstance	namespace περιπτώσεων σχημάτων XML
xsd	http://www.w3.org/1999/XMLSchema	namespace τύπων δεδομένων του σχήματος XML
tns	various	Το "this namespace" (tns) χρησιμοποιείται ως σύμβαση για να αναφερθεί το τρέχον έγγραφο.
various	some-URI	Όλα τα προθέματα namespace αυτής της μορφής είναι μόνο δείγματα ή αντιπροσωπεύουν μερικά εξαρτημένα από εφαρμογές ή εξαρτημένα από πλαίσια URI.

Μηνύματα απάντησης SOAP

authToken	Αυτό το μήνυμα επιστρέφει τις πληροφορίες πιστοποίησης ως απάντηση στο μήνυμα get_authToken. Το μήνυμα καλύπτει ένα ενιαίο στοιχείο authInfo που περιέχει ένα σημείο πρόσβασης το οποίο πρόκειται να περαστεί πίσω σε όλα τα μηνύματα έκδοσης API που αλλάζουν τα δεδομένα.
bindingDetail	Αυτό το μήνυμα επιστρέφει μια ή περισσότερες συγκεκριμένες δομές πληροφοριών bindingTemplate ως απάντηση σε ένα get_bindingDetail και

	ένα find_binding μήνυμα καθώς και ως απάντηση σε ένα save_binding μήνυμα δημοσίευσης για να δείξει την επιτυχή επεξεργασία.
businessDetail	Αυτό το μήνυμα επιστρέφει μια ή περισσότερες πλήρεις δομές businessEntity ως απάντηση στο μήνυμα get_businessDetail, και ως απάντηση στο save_business μήνυμα δημοσίευσης.
businessDetailExt	Αυτό το μήνυμα επιστρέφει ένα ή περισσότερα σύνολα δεδομένων businessEntityExt ως απάντηση στο get_businessDetailExt μήνυμα έρευνας. Αυτή η δομή επιτρέπει στους συμβατούς καταλόγους UDDI να καθορίζουν τις πρόσθετες πληροφορίες για ένα businessEntity. Εάν καμία εκτεταμένη πληροφορία δεν είναι διαθέσιμη, η δομή businessEntityExt περιλαμβάνει μόνο τα τυποποιημένα στοιχεία businessEntity.
businessList	Αυτή η δομή επιστρέφει ένα ή περισσότερα σύνολα δεδομένων businessInfo ως απάντηση σε ένα find_business μήνυμα έρευνας. Μια δομή businessInfo είναι μια συντομευμένη έκδοση της businessEntity δομής για την παροχή των πληροφοριών επισκόπησης για μια επιχείρηση.
registeredInfo	Αυτό το μήνυμα περιέχει μια ή περισσότερες δομές businessInfo και μια ή περισσότερες δομές tModelInfo ως απάντηση σε ένα μήνυμα get_registeredInfo. Το μήνυμα παρέχει συντομευμένα δεδομένα επισκόπησης όλων των πληροφοριών businessEntity και tModel που καταχωρούνται από τον εκδότη.
serviceDetail	Αυτό το μήνυμα επιστρέφει τα πλήρη στοιχεία για μια ή περισσότερες δομές businessService ως απάντηση σε ένα μήνυμα έρευνας get_serviceDetail και ως απάντηση στο save_service μήνυμα δημοσίευσης για να δείξει την επιτυχή εναποθήκευση.
serviceList	Αυτό το μήνυμα επιστρέφει μηδενικές ή περισσότερες δομές serviceInfo ως απάντηση σε ένα find_service μήνυμα έρευνας. Μια δομή serviceInfo περιέχει τα συντομευμένα δεδομένα για καταχωρημένες businessService πληροφορίες.
tModelDetail	Αυτό το μήνυμα επιστρέφει μηδενικές ή περισσότερο πλήρεις δομές tModel ως απάντηση στο get_tModelDetail μήνυμα έρευνας και ως απάντηση στο save_tModel μήνυμα δημοσίευσης για να επιβεβαιώσει την επιτυχή εναποθήκευση.
tModelList	Αυτό το μήνυμα επιστρέφει μηδενικές ή περισσότερο δομές tModelInfo ως απάντηση σε ένα find_tModel μήνυμα έρευνας. Μια δομή tModelInfo είναι μια συντομευμένη έκδοση των καταχωρημένων tModel πληροφοριών.

13. ΠΑΡΑΡΤΗΜΑ Β

13.1. Δεδομένα πειράματος

Τα δεδομένα πουυ χρησιμοποιήθηκαν στα πειράματα προέκυψαν από τον παρακάτω sql κώδικα:

```
truncate table webservice
declare @i int
set @i=1
while @i<=5
begin
        insert into webservice (name,url,resources,current_resources,grade)
        values ('service'+ cast ( @i as varchar),'url'+ cast ( @i as
        varchar),rand()*100+rand()*400,0,0)
        set @i=@i+1
end

truncate table clients
declare @j int
set @j=1
while @j<=100
begin
        insert into clients (name,email,request,lifetime,grade)
        values ('client'+ cast ( @j as varchar),'email'+ cast ( @j as
        varchar),1+rand()*49,rand()*100,rand()*10)
        set @j=@j+1
end
```

13.2. C# κώδικας του WSBroker

```
using System;
using System.Xml;
using System.Xml.Serialization;
using System.Data;
using System.Data.SqlClient;
using UVa.GCG.WSRF.Common.HttpServer;

using UVa.GCG.WSRF.Common.WS;
using UVa.GCG.WSRF.Common.WS.Addressing;
using UVa.GCG.WSRF.Common.WS.Grid;
using UVa.GCG.WSRF.Common.WS.ResourceLifetime;
using UVa.GCG.WSRF.Common.WS.ResourceLifetime.Proxies;

using UVa.GCG.WSRF.Common.WS.ResourceProperties;

using UVa.GCG.WSRF.Common.WS.Notification;
using UVa.GCG.WSRF.Common.WS.Notification.Proxies;
using UVa.GCG.WSRF.Common.WS.Notification.Topics;

using UVa.GCG.WSRF.Common.WS.ServiceGroup;

using AuctionCommon;

namespace BrokerAdmin
{
      class Class5
      {
            [STAThread]
            static void Main(string[] args)
            {

                  TopicExpression bidTopic
=WellknownDialects.SIMPLE.createExpression
                        (new XmlQualifiedName("new-
bid","http://wsrfnet.cs.virginia.edu/auction-tutorial"));

                  bidTopic.addHandler(new
TopicExpressionListener(notifyCallback));

                  AsynchronousNotificationListener listener =     new
AsynchronousNotificationListener(5432);

                  listener.registerExpression(bidTopic);

                  listener.start();
                  EndpointReferenceType manager = createManager(

      "http://localhost/AuctionServices/AuctionManager.asmx");

                  SqlConnection myConnection = new
SqlConnection("Initial Catalog=broker; Data
Source=150.140.142.30;Integrated Security=SSPI;");
                  myConnection.Open();

                  SqlDataAdapter DAdapt = new SqlDataAdapter();
```

```
                    SqlCommand CmdInit=new SqlCommand();
                    CmdInit=new SqlCommand("Initialize_Data",
myConnection);
                    CmdInit.CommandType = CommandType.StoredProcedure;
                    CmdInit.ExecuteNonQuery();


                    SqlDataAdapter Adapter = new SqlDataAdapter("Select *
from webservice", myConnection);
                    DataSet myDataSet = new DataSet();
                    Adapter.Fill(myDataSet,"WebServices");




                    foreach (DataRow services in
myDataSet.Tables["WebServices"].Rows)
                    {

      createAuction( manager,"http://localhost/AuctionServices/Auction.a
smx",services[0].ToString(),services[1].ToString(),services[2].ToString(
));
                    }

                    int ii=0;
                    while( ii<100)
                    {
                          unbid(manager);

                          DataRow client=chooseclient(ii);
                          string name = client[0].ToString();
                          string email = client[1].ToString();
                          int client_bid=Convert.ToInt32(client[2]);
                          int lifetime=Convert.ToInt32(client[3]);
                          int grade=Convert.ToInt32(client[4]);
                          Console.WriteLine("Client Requesting resources
{0}", name.TrimEnd(null) );


                          EntryType []entries = retrieveEntries(manager);
                          SqlCommand BestServicecmd=new SqlCommand("select
a.name FROM webservice AS a INNER JOIN (SELECT * FROM webservice WHERE
current_resources >'"+client_bid+"') AS b ON a.id = b.id where a.name
not in (select name from webservice_reserved)ORDER BY a.grade desc,
a.current_resources desc",myConnection);
                          object serv;
                          if ((serv=BestServicecmd.ExecuteScalar())!=null)
                          {
                                      string service=serv.ToString();
                                int i, index=0;
                                int length=entries.Length-1;
                                for (i=0;i<=length;i++ )
                                {
                                      if
(entries[i].Content[0].InnerText==service)
                                            index=i;
                                }
```

```
                              Console.WriteLine("Choosing Web Service
\"{0}\".", entries[index].Content[0].InnerText);
                              EndpointReferenceType auction=
entries[index].MemberServiceEPR;
                              EndpointReferenceType consumer =
createConsumer("http://localhost/AuctionServices/Consumer.asmx");

                              EndpointReferenceType consoleSubscription
=     createSubscription(bidTopic,
                                    auction, new EndpointReferenceType(
      new AttributedURI(null, "http://localhost:5432"),

      WSUtilities.createReferencePropertiesType(
      Guid.NewGuid().ToString()),   null, null, null, null));

                              EndpointReferenceType serviceSubscription
= createSubscription(bidTopic, auction, consumer);

                              BrokerAdmin.Auction.AuctionServiceWse
bidProxy =new BrokerAdmin.Auction.AuctionServiceWse();
                              WSUtilities.setEPR(bidProxy, auction);
                              //getProperties(auction);


      makeBid(name,email,client_bid,lifetime,grade,bidProxy, auction);


                              destroyEndpoint(consoleSubscription);
                              destroyEndpoint(serviceSubscription);
                              destroyEndpoint(consumer);
                              ii++;

                        }

                        else
                        {
                              String Failed= "Insert into resources
(service,resources,bidder,bid,failed)Values
('NotAvailable',0,'"+name+"','"+client_bid+"',1)";
                              SqlCommand FailedCmd = new
SqlCommand(Failed, myConnection);
                              FailedCmd.ExecuteNonQuery();


                              SqlDataAdapter Adapt = new
SqlDataAdapter("Select name,current_resources from  webservice where
name not in (select name from webservice_reserved)and
resources>'"+client_bid+"'order by current_resources desc",
myConnection);
                              DataSet RSDataSet = new DataSet();
                              Adapt.Fill(RSDataSet,"MaxWebServices");
                              DataRowCollection
MaxServices=RSDataSet.Tables[0].Rows;
                              if (MaxServices.Count>0)
                              {
                                    DataRow MaxService=MaxServices[0];
```

```
                                    string
servicename=MaxService[0].ToString();
                                    int
maxresources=Convert.ToInt32(MaxService[1].ToString());

                                    String Reserved= "Insert into
webservice_reserved
(name,resources,bidder,bid,lifetime,grade,reserved)Values
('"+servicename+"','"+maxresources+"','"+name+"','"+client_bid+"','"+lif
etime+"','"+grade+"',1)";
                                    SqlCommand ReservedCmd = new
SqlCommand(Reserved, myConnection);
                                    ReservedCmd.ExecuteNonQuery();
                              }

                              String lifetimeCmd =  "Update lifetime set
lifetime=lifetime-1";
                              SqlCommand lifetimeCommand = new
SqlCommand(lifetimeCmd, myConnection);
                              lifetimeCommand.ExecuteNonQuery();

                              ii++;
                        }

                  }


                  destroyAuctions(manager);

                  destroyEndpoint(manager);

                  listener.stop();
            }


            static private void unbid(EndpointReferenceType manager)
            {

                  SqlConnection myConnection = new
SqlConnection("Initial Catalog=broker; Data
Source=150.140.142.30;Integrated Security=SSPI;");
                  myConnection.Open();
                  SqlDataAdapter unbidAdapter = new
SqlDataAdapter("Select * from lifetime where lifetime=0", myConnection);
                  DataSet unbidDataSet = new DataSet();
                  unbidAdapter.Fill(unbidDataSet,"UnbidServices");

                  foreach (DataRow unbid in
unbidDataSet.Tables["UnbidServices"].Rows)
                  {
                        string unbid_name=unbid[0].ToString();
                        string unbid_email=unbid_name+"@mail";
                        int
client_unbid=Convert.ToInt32(unbid[1].ToString());
                        string service=unbid[2].ToString();
                        int
unbid_lifetime=Convert.ToInt32(unbid[3].ToString());
```

```
                        int
unbid_grade=Convert.ToInt32(unbid[4].ToString());
                        EntryType []entries = retrieveEntries(manager);
                        int j, index2=0;
                        int length=entries.Length-1;
                        for (j=0;j<=length;j++ )
                        {
                              if
(entries[j].Content[0].InnerText==service)
                              {
                                    index2=j;
                              }
                        }

                        EndpointReferenceType
auction2=entries[index2].MemberServiceEPR;
                        BrokerAdmin.Auction.AuctionServiceWse bidProxy2
=new BrokerAdmin.Auction.AuctionServiceWse();
                        WSUtilities.setEPR(bidProxy2, auction2);
                        makeBid(unbid_name,unbid_email,-
client_unbid,unbid_lifetime,unbid_grade,bidProxy2, auction2);

                        String UpdateRes = "update webservice_reserved
set resources=resources+"+client_unbid+" where name='"+service+"'";
                        SqlCommand UpdateReserved = new
SqlCommand(UpdateRes, myConnection);
                        UpdateReserved.ExecuteNonQuery();


                        SqlDataAdapter SelReserv = new
SqlDataAdapter("Select * from webservice_reserved where resources>=bid",
myConnection);
                        DataSet SrDataSet = new DataSet();
                        SelReserv.Fill(SrDataSet,"ReservedFull");

                        foreach (DataRow servicesRS in
SrDataSet.Tables["ReservedFull"].Rows)
                        {
                              int k, index3=0;
                              int length3=entries.Length-1;
                              for (k=0;k<=length3;k++ )
                              {
                                    if
(entries[k].Content[0].InnerText==servicesRS[0].ToString())
                                    {
                                          index3=k;
                                    }
                              }

                              EndpointReferenceType
auction3=entries[index3].MemberServiceEPR;
                              BrokerAdmin.Auction.AuctionServiceWse
bidProxy3 =new BrokerAdmin.Auction.AuctionServiceWse();
                              WSUtilities.setEPR(bidProxy3, auction3);
```

```
      makeBid(servicesRS[2].ToString(),servicesRS[2].ToString(),Convert.
ToInt32(servicesRS[3].ToString()),Convert.ToInt32(servicesRS[6].ToString
()),Convert.ToInt32(servicesRS[5].ToString()),bidProxy3, auction3);

                        }

                        String DelReserv = "Delete from
webservice_reserved where resources>=bid";
                        SqlCommand DelReserved = new
SqlCommand(DelReserv, myConnection);
                        DelReserved.ExecuteNonQuery();

                  }

                  String DelLife = "Delete from lifetime where
lifetime=0";
                  SqlCommand DelLifetime = new SqlCommand(DelLife,
myConnection);
                  DelLifetime.ExecuteNonQuery();



                  //unbid//


            }


            static private System.Data.DataRow chooseclient(int i)
            {
                  SqlConnection myConnection = new
SqlConnection("Initial Catalog=broker; Data
Source=150.140.142.30;Integrated Security=SSPI;");
                  myConnection.Open();
                  SqlDataAdapter Adapter2 = new SqlDataAdapter("Select *
from clients order by id", myConnection);
                  DataSet myDataSet2 = new DataSet();
                  Adapter2.Fill(myDataSet2,"Clients");
                  DataRowCollection clients=myDataSet2.Tables[0].Rows;
                  DataRow client=clients[i];
                  return client;

            }




            static private EndpointReferenceType createManager(string
url)
            {
                  GCGResourceFactoryBinding proxy =
                        new GCGResourceFactoryBinding(url);
                  ServiceGroupPortTypeInit sgInit = new
ServiceGroupPortTypeInit();
                  MembershipContentRule rule = new
MembershipContentRule();
                  rule.ContentElements = new XmlQualifiedName[]
```

```
                        {
                              new XmlQualifiedName("AuctionDescription",
                              "http://wsrfnet.cs.virginia.edu/auction-
tutorial")
                        };
                  sgInit.Rules.Add(rule);
                  Create create = new Create();
                  create.PortTypeInitializers = new XmlElement[]
                        {
                              WSUtilities.Serialize(sgInit)
                        };

                  return proxy.Create(create).ResourceEndpoint;
            }

            static private EndpointReferenceType createConsumer(string
serviceURL)
            {
                  GCGResourceFactoryBinding proxy =
                        new GCGResourceFactoryBinding(serviceURL);
                  Create creationParms = new Create();

                  CreateResponse response =
                        proxy.Create(creationParms);

                  return response.ResourceEndpoint;
            }

            static private EndpointReferenceType createSubscription(
                  TopicExpression bidTopic,EndpointReferenceType
producer,EndpointReferenceType consumer)
            {
                  TopicExpressionType topic =
bidTopic.TopicExpressionType;

                  NotificationProducerProxy proxy =
                        new NotificationProducerProxy(producer);
                  return proxy.Subscribe( new
SubscribeRequest(consumer,topic)).SubscriptionReference;
            }

            static private void notifyCallback(Topic topic,
NotificationMessageHolderType message)
            {
                  Console.WriteLine(
                        "Received notification that the bid has changed
to {0}.",
                        message.Message.InnerText);
            }

            static private string getProperties(EndpointReferenceType
auction)
            {
                  GetResourcePropertyBinding proxy =new
GetResourcePropertyBinding(auction);

                  XmlElement val = proxy.GetResourceProperty(     new
```

```
XmlQualifiedName("AuctionDescription",
                        "http://wsrfnet.cs.virginia.edu/auction-
tutorial")).Any[0]      as XmlElement;
                  return(val.InnerText);
            }

            static private bool makeBid(string name, string email, int
client_bid,int lifetime,int grade, BrokerAdmin.Auction.AuctionServiceWse
bidProxy, EndpointReferenceType auction )
            {
                  BrokerAdmin.Auction.BidderInformation info =new
BrokerAdmin.Auction.BidderInformation();

                  int newbid=client_bid;
                  string auctionname= getProperties(auction);
                  info.Name = name;
                  info.Email = email;


                  if (bidProxy.bid(info, newbid,lifetime,grade,
auctionname))
                  {
                        Console.WriteLine("Bid  succeeded for {0}",
info.Name.TrimEnd(null));
                        return true;
                  }
                  else
                        Console.WriteLine("Bid  failed for {0}.",
info.Name);
                  return false;
            }


            static private EndpointReferenceType
createAuction(EndpointReferenceType manager,
                  string serviceURL, string description, string seller,
string resources)
            {
                  AuctionConstructionParameters cons =
                        new AuctionConstructionParameters(description,
seller, resources);

                  GCGResourceFactoryBinding proxy =
                        new GCGResourceFactoryBinding(serviceURL);
                  Create creationParms = new Create();

                  creationParms.PortTypeInitializers = new XmlElement[]
                        {
                              WSUtilities.Serialize(cons)
                        };

                  CreateResponse response =
                        proxy.Create(creationParms);

                  AddType add = new AddType();
                  XmlDocument doc = new XmlDocument();
```

```
      doc.AppendChild(doc.CreateElement("AuctionDescription",
                        "http://wsrfnet.cs.virginia.edu/auction-
tutorial"));
                  doc.DocumentElement.InnerText = description;
                  add.Content = new XmlElement[]
                        {
                              doc.DocumentElement
                        };
                  add.MemberEPR = response.ResourceEndpoint;
                  add.InitialTerminationTimeSpecified = false;
                  ServiceGroupRegistrationBinding sg = new
ServiceGroupRegistrationBinding(manager);
                  sg.Add(add);

                  return response.ResourceEndpoint;
            }

            static private void destroyEndpoint(EndpointReferenceType
epr)
            {
                  ImmediateResourceTerminationProxy proxy =
                        new ImmediateResourceTerminationProxy(epr);
                  proxy.Destroy(new Destroy());
            }

            static private EntryType[]
retrieveEntries(EndpointReferenceType manager)
            {
                  int lcv;
                  XmlRootAttribute rootAttr = new
XmlRootAttribute("Entry");
                  rootAttr.Namespace =
WSConstants.WS_SERVICE_GROUPS_XSD_NS;

                  QueryResourcePropertiesRequest query = new
QueryResourcePropertiesRequest();
                  query.QueryExpression = new QueryExpressionType();
                  query.QueryExpression.dialect =
"http://www.w3.org/TR/1999/REC-xpath-19991116";

                  string xpathQuery = string.Format("/*/*[namespace-
uri()='{0}' and local-name()='Entry' "
                        + "and *[namespace-uri()='{0}' and" +
                        " local-name()='Content' and *[namespace-
uri()='{1}' and local-name()='AuctionDescription']]]",
                        WSConstants.WS_SERVICE_GROUPS_XSD_NS,
"http://wsrfnet.cs.virginia.edu/auction-tutorial");
                  XmlDocument doc = new XmlDocument();
                  query.QueryExpression.Any = new XmlNode[]
{ doc.CreateTextNode(xpathQuery) };

                  QueryResourcePropertiesBinding tsgService = new
QueryResourcePropertiesBinding(manager);
                  QueryResourcePropertiesResponse response =
tsgService.QueryResourceProperties(query);
                  EntryType []entries = new
EntryType[response.Any.Length];
```

```
                  for (lcv = 0; lcv < response.Any.Length; lcv++)
                  {
                        entries[lcv] =
(EntryType)WSUtilities.Deserialize(response.Any[lcv],
                              typeof(EntryType), rootAttr);

                  }

                  return entries;
            }



            static private void destroyAuctions(EndpointReferenceType
manager)
            {
                  EntryType []entries = retrieveEntries(manager);

                  foreach (EntryType entry in entries)
                  {
                        destroyEndpoint(entry.MemberServiceEPR);
                  }
            }
      }
}
```

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Diagnostics;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Xml;
using System.Xml.Serialization;
using UVa.GCG.WSRF.Common.Attributes;
using UVa.GCG.WSRF.Service.BaseTypes;
using UVa.GCG.WSRF.Service.Grid;
using UVa.GCG.WSRF.Service.ResourceLifetime;
using UVa.GCG.WSRF.Service.ResourceProperties;
using UVa.GCG.WSRF.Service.WS.Notification;
using UVa.GCG.WSRF.Service.WS.Notification.PortTypes;
using Common;

using UVa.GCG.WSRF.Common.HttpServer;
using UVa.GCG.WSRF.Common.WS;
using UVa.GCG.WSRF.Common.WS.Addressing;
using UVa.GCG.WSRF.Common.WS.ResourceLifetime;
```

```
using UVa.GCG.WSRF.Common.WS.ResourceLifetime.Proxies;
using UVa.GCG.WSRF.Common.WS.ResourceProperties;
using UVa.GCG.WSRF.Common.WS.Notification;
using UVa.GCG.WSRF.Common.WS.Notification.Proxies;
using UVa.GCG.WSRF.Common.WS.Notification.Topics;

using UVa.GCG.WSRF.Common.WS.ServiceGroup;


namespace MyService
{
      [WsdlBaseName("Auction",
            "http://wsrfnet.cs.virginia.edu/auction-tutorial")]
      [WebService]
      [WebServiceBinding]
      [WSRFPortType(typeof(ImmediateResourceTerminationPortType))]
      [WSRFPortType(typeof(GCGResourceFactoryPortType))]
      [WSRFPortType(typeof(GetResourcePropertyPortType))]
      [WSRFPortType(typeof(NotificationProducerPortType))]
      [ResourceInitializerType(typeof(AuctionConstructionParameters))]
      public class Service1 : ServiceSkeleton
      {
            [Resource]
            [ResourceProperty("CurrentBid", typeof(int), false)]
            private int _currentBid;

            [Resource]
            private BidderInformation _currentBidder;

            [Resource]
            [ResourceProperty]
            public string AuctionDescription;

            [Resource]
            [ResourceProperty]
            private string AuctionOwner;

            [Resource]
            [ResourceProperty]
            private string AuctionResources;

            public  int mybid;

            public Service1()
            {
                  InitializeComponent();
            }

            public override void InitResource(Hashtable parms)
            {
                  AuctionConstructionParameters cons =
                   parms[this.GetType().FullName] as
                   ConstructionParameters;

                  _currentBid = Convert.ToInt32(cons.Resources);
                  _currentBidder = null;
```

```
                  Description = cons.Description;
                  Owner = cons.Owner;
                  Resources=cons.Resources;
            }

            protected override void portInit()
            {
                  ServiceBase.Topics.addTopicHierarchy(
                        new XmlQualifiedName("new-bid",
                              "http://wsrfnet.cs.virginia.edu/auction-
tutorial"),true);
            }


            [WebMethod]

      [SoapDocumentMethod("http://wsrfnet.cs.virginia.edu/auction-
tutorial/bid",
                  ParameterStyle=SoapParameterStyle.Bare)]
            [return: XmlElement("bidSuccessful",
                  Namespace="http://wsrfnet.cs.virginia.edu/auction-
tutorial")]




            public bool bid(BidderInformation bidder, int newBid,int
lifetime, int grade, string auction)
            {
                  SqlConnection myConnection = new
SqlConnection("Initial Catalog=broker; Data
Source=150.140.142.30;Integrated Security=SSPI;");
                  myConnection.Open();
                  string myauction=auction.ToString();
                  string mybidder=bidder.Name.ToString();

                  if (newBid<0)
                  {

                        _currentBid=_currentBid-newBid;
                        mybid=_currentBid;
                        String Cmd2 = "INSERT INTO resources
(service,bidder,resources,bid,
failed)VALUES('"+myauction+"','"+mybidder+"',"+ _currentBid+","+newBid
+",2)";
                        SqlCommand CurrentCommand2 = new
SqlCommand(Cmd2, myConnection);
                        CurrentCommand2.ExecuteNonQuery();

                        String CmdUpdate = "Update webservice set
current_resources="+ _currentBid+",grade=grade+'"+grade+"' where
(name='"+myauction+"')";
                        SqlCommand Update = new SqlCommand(CmdUpdate,
myConnection);
                        Update.ExecuteNonQuery();
                        myConnection.Close();
```

```
                        ServiceBase.Topics[
                        new XmlQualifiedName("new-
bid","http://wsrfnet.cs.virginia.edu/auction-tutorial")].notify(mybid);
                        return true;


                  }

                  else
                  {

                        _currentBid=_currentBid-newBid;


                        if (_currentBid< 1)
                        {
                              _currentBid=_currentBid+newBid;

                              String Cmd =  "INSERT INTO resources
(service,bidder,resources,bid,
failed)VALUES('"+myauction+"','"+mybidder+"',"+ _currentBid+","+newBid
+",1)";
                              SqlCommand CurrentCommand = new
SqlCommand(Cmd, myConnection);
                              CurrentCommand.ExecuteNonQuery();
                              String lifetimeCmd =  "Update lifetime
set lifetime=lifetime-1";
                              SqlCommand lifetimeCommand = new
SqlCommand(lifetimeCmd, myConnection);
                              lifetimeCommand.ExecuteNonQuery();
                              myConnection.Close();


                              return false;
                        }

                        _currentBidder = bidder;

                        mybid=_currentBid;
                        String Cmd2 = "INSERT INTO resources
(service,bidder,resources,bid,
failed)VALUES('"+myauction+"','"+mybidder+"',"+ _currentBid+","+newBid
+",0)";
                        String CmdUpdate = "Update webservice set
current_resources="+ _currentBid+" where (name='"+myauction+"')";
                        SqlCommand CurrentCommand2 = new
SqlCommand(Cmd2, myConnection);
                        CurrentCommand2.ExecuteNonQuery();
                        SqlCommand Update = new SqlCommand(CmdUpdate,
myConnection);
                        Update.ExecuteNonQuery();
                        String lifetimeCmd2 =  "Update lifetime set
lifetime=lifetime-1";
                        SqlCommand lifetimeCommand2 = new
SqlCommand(lifetimeCmd2, myConnection);
                        lifetimeCommand2.ExecuteNonQuery();
```

```
                        String lifeCmd =  "INSERT INTO lifetime
(service,bidder,bid,
lifetime,grade)VALUES('"+myauction+"','"+mybidder+"',"+newBid
+","+lifetime+","+grade+")";
                        SqlCommand lifeCommand = new
SqlCommand(lifeCmd, myConnection);
                        lifeCommand.ExecuteNonQuery();

                        myConnection.Close();

                        ServiceBase.Topics[
                              new XmlQualifiedName("new-bid",
                              "http://wsrfnet.cs.virginia.edu/auction-
tutorial")].notify(mybid);

                        return true;
                  }

            }

            #region Component Designer generated code
            //Required by the Web Services Designer
            private IContainer components = null;

            private void InitializeComponent()
            {
            }
            protected override void Dispose( bool disposing )
            {
                  if(disposing && components != null)
                  {
                        components.Dispose();
                  }
                  base.Dispose(disposing);
            }

            #endregion
      }
}
```