Πανεπιστήμιο Πειραιώς
Τμήμα Ψηφιακών Συστημάτων
Π.Μ.Σ. Διδακτική της Τεχνολογίας και Ψηφιακά Συστήματα
Κατεύθυνση Δικτυοκεντρικά Συστήματα

«Ανάπτυξη Εφαρμογής Με Την Χρήση Service-oriented Architecture»

Διπλωματική Εργασία

της

Μπούρχας Παναγιώτας (ΜΕ10091)

Επιβλέπων: Μαρίνος Θεμιστοκλέους

Αναπληρωτής Καθηγητής

Πειραιάς 2015

Συντομογραφίες

API	application program interface
App(s)	Εφαρμογή/ές
HTTP	Hypertext Transfer Protocol
JSON	Javascript Object Notation
REST	Representational State Transfer
SOA	Service Oriented Architecture
XML	EXtensible Markup Language
SOAP	Simple Object Access protocol
WSDL	Web Services Description Language

Ευχαριστίες

Θα ήθελα να εκφράσω τις πιο θερμές μου ευχαριστίες προς τον Αναπληρωτή Καθηγητή κ. Μαρίνο Θεμιστοκλέους για την επίβλεψη, την υπομονή, τις πολύτιμες συμβουλές του και την καθοδήγηση στην διάρκεια της διπλωματικές μου εργασίες.

Περιεχόμενα

Ευρετήριο Εικόνων

Ευρετήριο Σχημάτων

Ευρετήριο Πινάκων

Ευρετήριο Διαγραμμάτων

Πρόλογος

Η παρούσα διπλωματική εργασία στοχεύει στην αυτοματοποίηση επιχειρηματικών διαδικασιών στα πλαίσια της υπηρεσιοστρεφής αρχιτεκτονικής SOA με την χρήση της τεχνολογίας υπηρεσίες διαδικτύου (Web Services). Η διαδικασία που αυτοματοποιείται είναι αυτή της συλλογής δεδομένων προγνωστικών από πολλαπλές πηγές. Σε αυτή την διαδικασία λαμβάνεται πληροφορία μέσω διαφορετικών υπηρεσιών διαδικτύου (Web Services) ανεξαρτήτου μορφής (format), επεξεργάζεται και διατίθεται σε όποια μορφή επιθυμεί ο χρήστης, xml ή json. Τέλος, παρουσιάζεται η εφαρμογή που αναπτύχθηκε και προσημειώνει την διαδικασία αυτή, αναλύεται η αρχιτεκτονική της και περιγράφονται οι τεχνολογίες που χρησιμοποιήθηκαν που επί το πλείστων αποτελούν τεχνολογίες αιχμής.

Λέξεις κλειδιά

Service Oriented Architecture (SOA), Web Service, XML, SOAP, WSDL

1 Κεφάλαιο 1 (Ποιο είναι το ερευνητικό πρόβλημα)

1.1 Περίληψη

Μετά τη Μεσολυμπιάδα του 1906 οι Έλληνες άρχισαν να ενδιαφέρονται συλλογικά για το ποδόσφαιρο. Ως άθλημα αναγνωρίστηκε επίσημα από τον Σύνδεσμο Ελληνικών Γυμναστικών Αθλητικών Σωματείων (Σ.Ε.Γ.Α.Σ.) το 1910, ενώ το 1919 το ποδόσφαιρο αυτονομείται δημιουργώντας την Ένωση Ποδοσφαιρικών Σωματείων Αθηνών Πειραιώς (Ε.Π.Σ.Α.Π.), η οποία τον επόμενο χρόνο μετονομάστηκε σε Ένωση Ποδοσφαιρικών Σωματείων Ελλάδος (Ε.Π.Σ.Ε.).

Η δυναμική τού αθλήματος ενισχύθηκε τα χρόνια μετά τη Μικρασιατική Καταστροφή, με την έλευση των προσφύγων και νέου αθλητικού «αίματος», κατακτώντας όλο και περισσότερους οπαδούς.

Το 1923, η Ε.Π.Σ.Ε. γίνεται μέλος της Παγκόσμιας Ποδοσφαιρικής Ομοσπονδίας (FIFA). Το 1926, η Ε.Π.Σ.Ε. δίνει τη θέση της στην Ελληνική Ποδοσφαιρική Ομοσπονδία (Ε.Π.Ο.), η οποία μέχρι τις μέρες μας αποτελεί τον ανώτερο ποδοσφαιρικό θεσμό της χώρας.

Το πρωτάθλημα της εποχής εμφανίζει ιδιαίτερο ενδιαφέρον και οι λάτρεις του αθλήματος πολύ σύντομα επιδόθηκαν σε μια συνήθεια, η οποία επρόκειτο να κυριαρχήσει σε ολόκληρο το περιβάλλον του ποδοσφαίρου μέχρι σήμερα: την πρόγνωση των αποτελεσμάτων των αγώνων.

Σε μια απόπειρα οργάνωσης της αθλητική ζωή της χώρας, η οποία βρίσκονταν ακόμα σε εμβρυακή κατάσταση, το 1957 συγκροτείται η Γενική Γραμματεία Αθλητισμού, το 1958 με το Βασιλικό Διάταγμα «Περί Συστάσεως Οργανισμού Προγνωστικών Αγώνων Ποδοσφαίρου» θεσμοθετείται ο ΟΠΑΠ και το 1959 καθιερώνεται στο ποδόσφαιρο το Σύστημα της Εθνικής Κατηγορίας.

Την 1η Μαρτίου του 1959 κάνει την εμφάνισή του το πρώτο παιχνίδι που αφορά το ποδόσφαιρό και βασίζεται σε προβλέψεις και προγνωστικά επικείμενων αγώνων , το ΠΡΟ-ΠΟ .

Έκτοτε έχει δημιουργηθεί από τον ΟΠΑΠ πληθώρα παιχνιδιών που σχετίζονται με το ποδόσφαιρό, καθώς και άλλα αθλήματα, κάθε ένα από τα οποία έχει σαν κοινό παρονομαστή την έγκαιρη και έγκυρη ενημέρωση σχετικά με στατιστικά των ομάδων που συμμετέχουν στο Ελληνικό πρωτάθλημα.

Την ανάγκη αυτή αναγνώρισε και επιχείρησε να καλύψει πρώτος ο Γιώργος Παρασκευάς, ο οποίος στα μέσα της δεκαετίας του 90, και ενώ πολλές από τις σημερινές εταιρίες στοιχημάτων δεν υπήρχαν, ίδρυσε στην Ηλιούπολη μια εταιρία πληροφόρησης η οποία πουλούσε τα αποτελέσματα των αγώνων μέσω τηλέφωνου, παρέχοντας στους συνδρομητές της τα live scores.

Στις μέρες μας η ανάγκη αυτή καλύπτεται από την πληθώρα ιστοσελίδων προγνωστικών μέσω των οποίων, είτε με συνδρομή, είτε δωρεάν, ο χρήστη μπορεί να ανασύρει ιστορικές πληροφορίες και στατιστικά ανά ομάδα, καθώς και προγνωστικά για μελλοντικά παιχνίδια.

Παρατηρείται όμως ότι αυτή η υπερ-πληθώρα πληροφόρησης μπορεί να παρουσιάζει μικρές ή και μεγάλες αποκλείσεις από ιστοσελίδα σε ιστοσελίδα, ενώ δεν καθίσταται σαφές ποια είναι η πηγή πληροφόρησης και πόσο αξιόπιστη είναι.

Τέλος, αν και στον αλγόριθμο υπολογισμού το τυχερού παιχνιδιού "Πάμε Στοίχημα" εκτός από τα στατιστικά της κάθε ομάδας λαμβάνονται υπόψη και τα στατιστικά των παιχτών, δεν υπάρχει αντίστοιχη πληροφόρηση στις υπάρχουσες ιστοσελίδες παροχής στατιστικών και προγνωστικών.

1.2 Ερευνητικό πρόβλημα

Στην παρούσα διπλωματική εργασίες γίνεται μια προσπάθεια ανάπτυξης μιας υπηρεσία διαδικτύου (web service) η οποία θα στοχεύει στην βελτίωση της ποιότητας των δεδομένων που θα διατίθενται στους παίκτες προγνωστικών. Στοχεύει σε έναν συνδυαστικό ρόλο σε αντίθεση με τις υπάρχουσες ιστοσελίδες προγνωστικών που η κάθε μία φέρεται να έχει την δική της πηγή δεδομένων. Ποιο συγκεκριμένα:

1. Θα αναγράφονται με σαφήνεια οι πηγές

2. Θα συλλέγεται και θα διατίθεται πληροφορία από δύο ανεξάρτητες πηγές με δυνατότητα επέκτασης
3. Θα συλλέγεται και θα διατίθεται πληροφορία που έχει να κάνει με τα παιχνίδια και τις αποδόσεις αυτών
4. Ενώ μελλοντικά, συνδυάζοντας και συγκρίνοντας τις πηγές αυτές, θα μπορεί να προτείνει την εγκυρότερη.

1.3 Σκοπός και αντικειμενικοί στόχοι

Σκοπός της παρούσας εργασίας είναι η διερεύνηση της υπάρχουσας κατάστασης όσον αφορά την παροχή πληροφοριών σχετικά με το ποδόσφαιρο από υπάρχουσες ιστοσελίδες προγνωστικών, ενώ προτείνεται και υλοποιείται μια υπηρεσία διαδικτύου (web service) για την παροχή και διανομή σχετικής πληροφορίας από πολλαπλές πηγές.

Ο ανωτέρω σκοπός εξειδικεύεται στους εξής επιμέρους αντικειμενικούς στόχους:

1. Βιβλιογραφική Ανασκόπηση με στόχο την κατανόηση και τον προσδιορισμό των βασικών θεματικών περιοχών προς διερεύνηση.
2. Ανάλυση των πρακτικών που ακολουθούν οι ιστοσελίδες προγνωστικών.
3. Καταγραφή και μοντελοποίηση των δεδομένων που απαιτούνται από τους χρήστες για έγκυρε προβλέψεις
4. Να καταγραφεί η μεθοδολογία που θα ακολουθηθεί στην παρούσα πτυχιακή εργασία
5. Ανάπτυξη υπηρεσίας διαδικτύου (web service) που θα παρέχει συνδυαστική πληροφορία

1.4 Δομή εργασίας

Η παρούσα εργασία αποτελείται από 5 κεφάλαια:

- **Κεφάλαιο 1ο**: Στο πρώτο κεφάλαιο, γίνεται η εισαγωγή στο θέμα της διπλωματικής εργασίας, διατυπώνοντας τον ορισμό του προβλήματος και τέλος περιγράφεται η δομή της εργασίας.
- **Κεφάλαιο 2ο**: Στο δεύτερο κεφάλαιο γίνεται ανασκοπεί τη βιβλιογραφία και τις βασικές θεματικές περιοχές η οποίες είναι η αρχιτεκτονική SOA και η διακυβέρνηση της, τα Web Services, η ενορχήστρωση των Web Services και η χορογραφία τους.
- **Κεφάλαιο 3ο**: Στο τρίτο κεφάλαιο, γίνεται αναφορά στις τεχνολογίες και στα πρωτόκολλα ανάπτυξης που χρησιμοποιηθήκαν κατά την ανάπτυξη της εφαρμογής.
- **Κεφάλαιο 4ο**: Στο τέταρτο κεφάλαιο, πραγματοποιείται η παρουσίαση της χρήσης της εφαρμογής.
- **Κεφάλαιο 5ο**: Στο πέμπτο κεφάλαιο, παρουσιάζονται τα κυριότερα συμπεράσματα της εργασίας.

2 Κεφάλαιο 2 (Γιατί είναι πρόβλημα)

2.1 Περίληψη

Σε αυτό το κεφάλαιο θα παρουσιαστή το θεωρητικό υπόβαθρο και η βιβλιογραφία σχετικά με τις τεχνολογίες που χρησιμοποιήθηκαν. Στην πρώτη ενότητα του κεφαλαίου αυτού θα γίνει μια σύντομη ιστορική αναδρομή στον τρόπο με τον οποίο λειτουργούσαν οι επιχειρήσεις πριν την εμφάνιση των υπηρεσιοστρεφών αρχιτεκτονικών (Service Oriented Architecture).

Στην δεύτερη ενότητα θα δοθεί ο ορισμός των υπηρεσιοστρεφών αρχιτεκτονικών (SOA) δεύτερη ενότητα, καθώς και τα πλεονεκτήματα και τα μειονεκτήματά τους. Στην τρίτη και τέταρτη ενότητα θα παρατεθεί η σχετική θεωρία σημαντικών δομικών στοιχείων των υπηρεσιοστρεφών αρχιτεκτονικών (SOA), όπως είναι τα Web Services που ανήκουν, η χορογραφία και η ενορχήστρωσή υπηρεσιών.

2.2 Ιστορική Αναδρομή

Όταν η ιδέα των υπηρεσιοστρεφών αρχιτεκτονικών (SOA) πρωτοεμφανίστηκε το 2005 προκάλεσε αναστάτωση. Οι υποστηρικτές της προέβλεψαν ότι θα αντικαταστήσει την αρχιτεκτονική της παραδοσιακής τεχνολογία της πληροφορίας (IT), ενώ εκείνοι απάντησαν ότι η υπηρεσιοστρεφής αρχιτεκτονικη (SOA) δεν ήταν κάτι νέο, αλλά μια επανάληψη των παλαιών – και καλών –ιδεών περί ενθυλάκωσης (encapsulation) και χαλαρής σύνδεσης (loose coupling).

Υπάρχει κάποια αλήθεια και στις δύο αυτές θέσεις, αλλά στην πραγματικότητα είναι και οι δύο εσφαλμένες. Παρά το γεγονός ότι η υπηρεσιοστρεφής αρχιτεκτονικη (SOA) περιλαμβάνει παλαιότερες αρχιτεκτονικές ιδέες, είναι ένα ξεχωριστό ύφος που σηματοδοτεί ένα σημαντικό βήμα προς τα εμπρός. Παρόλα αυτά, για να εκλάβει μια επιχείρηση το μέγιστο όφελος από μια υπηρεσιοστρεφής αρχιτεκτονικη (SOA), χρειάζεται πρακτικές και μεθόδους της παραδοσιακής αρχιτεκτονικής.

Προάγγελος των υπηρεσιοστρεφών αρχιτεκτονικών (SOA) υπήρξαν τα συνδεδεμένα ή κατανεμημένα συστήματα. Μέχρι τα μέσα της δεκαετίας του '80, οι υπολογιστές χαρακτηρίζονταν από μεγάλο μέγεθος και υψηλό κόστος. Το γεγονός αυτό είχε ως αποτέλεσμα, οι περισσότεροι οργανισμοί να επένδυαν στην προμήθεια μικρού αριθμού υπολογιστών που λειτουργούσαν συνήθως ανεξάρτητα, λόγω της ανυπαρξίας τρόπου διασύνδεσης μεταξύ τους.

Αργότερα, δύο τεχνολογικά επιτεύγματα άρχισαν να διαφοροποιούν αυτήν την κατάσταση:

- η ανάπτυξη ισχυρών μικροεπεξεργαστών (microprocessors)
- η δημιουργία τοπικών δικτύων υψηλών ταχυτήτων (high speed Local Area Networks Local Area Networks)

Το τελικά αποτέλεσμα των δύο αυτών επιτευγμάτων, είναι η δυνατότητα να λειτουργήσουν μαζί υπολογιστικά συστήματα που αποτελούνται από μεγάλο αριθμό CPU συνδεδεμένων μέσω δικτύων υψηλών ταχυτήτων.

Τα συστήματα αυτά αποκαλούνται συνήθως κατανεμημένα συστήματα (distributed systems).

Τα προηγούμενα παραδοσιακά συστήματα καλούνται κεντρικοποιημένα συστήματα (centralized systems) αποτελούμενα από μία μοναδική CPU, την κεντρική μνήμη, τα περιφερειακά και τα τερματικά.

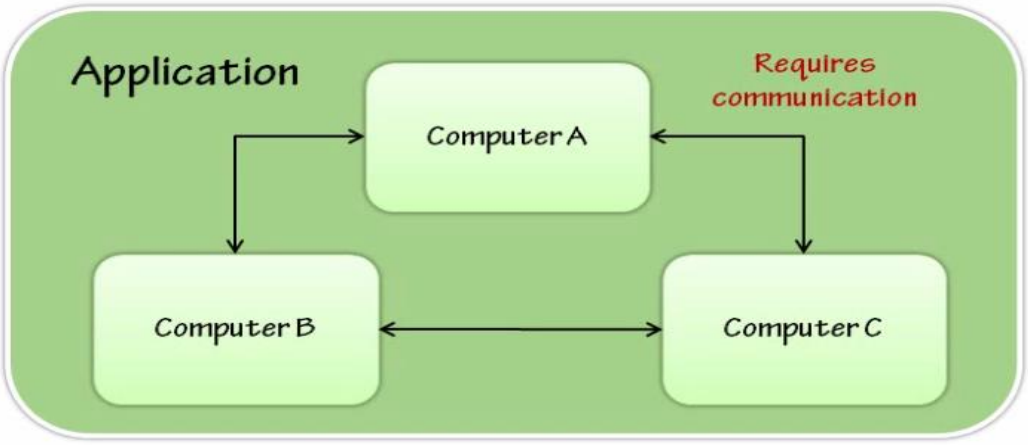
2.2.1 Ορισμός κατανεμημένων συστημάτων

Στην πληροφορική, παράλληλα, κατανεμημένα ή ταυτόχρονα συστήματα ονομάζονται υπολογιστές οι οποίοι επιτρέπουν την ταυτόχρονη εκτέλεση πολλαπλών συνεργαζόμενων προγραμμάτων σε μία ή περισσότερες επεξεργαστικές μονάδες. Οι διαφορές μεταξύ αυτών των όρων είναι λεπτές, με την έμφαση να δίνεται άλλοτε στον σχεδιασμό και ανάλυση αλγορίθμων, άλλοτε στην κατασκευή υποστηρικτικού λογισμικού και άλλοτε στη σχεδίαση των υποδομών υλικού που απαιτούνται για την επίτευξη του ταυτοχρονισμού.

Πιο συγκεκριμένα, ως κατανεμημένο σύστημα ορίζεται μια εφαρμογή η οποία διανέμετε σε πολλούς διαφορετικούς κόμβους υπολογιστών (distributed across multiple computer nodes) με την χρήση κάποιου δικτυακού πρωτοκόλλου επικοινωνίας.

Όπως φαίνεται στο Σχήμα 1 στα συνδεδεμένα συστήματα υπάρχει μια εφαρμογή η οποία αναπτύσσεται κατά μήκος τριών διαφορετικών υπολογιστών. Κάθε ένας από τους υπολογιστές αυτούς εκτελεί μέρος της επιχειρηματικής λογικής της εφαρμογής αυτής, ενώ και οι τρις πρέπει να συνεργαστούν ώστε να εκτελεστεί η λειτουργία της εφαρμογής στο σύνολό της.

Για να συνεργαστούν οι τρις αυτοί υπολογιστές απαραίτητη προϋπόθεση είναι ότι υπάρχει κάποιος τρόπος να επικοινωνήσουν μεταξύ τους, δηλαδή έχει προβλεφθεί κάποιος τρόπος ώστε να ανταλλάσσουν πληροφορία με την χρήση κάποιου δικτυακού πρωτοκόλλου (network protocol) .



Εικόνα 1 Αναπαράσταση συνδεδεμένου συστήματος

Η αυξανόμενη ανάγκη για ενσωμάτωση (integrate) πολύ μεγάλων, χαλαρά συνδεδεμένων συστημάτων με σκοπό την αυτοματοποίηση διαδικασιών μεταξύ επιχειρήσεων (business-to-business), έχει τονίσει το πόσο δύσκολη και πολύπλοκη είναι η επέκταση των ήδη υπάρχοντων τεχνολογιών πέρα από τα όρια μιας επιχείρησης. Η υπηρεσιοστρεφής αρχιτεκτονική (SOA) έρχεται να γεφυρώσει το

χάσμα αυτό, σκοπός της είναι να παρέχει υπηρεσίες που καλύπτουν τόσο τις ενδο-επιχειρησιακές, όσο και τις δια-επιχειρησιακές ανάγκες.

2.3 Υπηρεσιοστρεφής Αρχιτεκτονική (Service Oriented Architecture)

Ένα από τα χαρακτηριστικά των υπηρεσιοστρεφών αρχιτεκτονικών (SOA) είναι ο κατακερματισμός μεγάλων προβλημάτων σε μικρότερα επιμέρους προβλήματα Βάση της λογικής αυτής, θα εξεταστούν οι όροι υπηρεσία (service) και υπηρεσιοστρεφής (service-oriented) πριν την περαιτέρω ανάλυση της θεωρίας των υπηρεσιοστρεφών αρχιτεκτονικών.

2.3.1 Υπηρεσία (service) και Υπηρεσιοστρεφής (service-oriented)

Στον χώρο των υπολογιστών υπάρχουν πολλών διαφορετικών ειδών υπηρεσίες (services), όταν όμως αναφερόμαστε στις υπηρεσιοστρεφής αρχιτεκτονικές (SOA) εστιάζουμε στις υπηρεσίες που εμπλέκονται στην επιχειρηματική λογική (business logic) ενός οργανισμού

Σύμφωνα με το The Open Group, ως υπηρεσία (service) ορίζεται:

Μια λογική αναπαράσταση μιας επαναλαμβανόμενης επιχειρηματικής δραστηριότητάς η οποία έχει ένα συγκεκριμένο αποτέλεσμα.

Είναι αυτόνομη

Μπορεί να αποτελείται από άλλες υπηρεσίες

Για τους καταναλωτές της η υπηρεσία αποτελεί ένα «μαύρο κουτί», δηλαδή δεν γνωρίζουν τις ακριβής λειτουργίες που εκτελεί, παρά μόνο ποια είναι τα δεδομένα που πρέπει να παρέχουν στην υπηρεσία και ποια είναι τα αποτελέσματα που θα λάβουν.

Επίσης, το The Open Group ορίζει ως υπηρεσιοστρεφής (service-oriented) τον τρόπο σκέψης που εστιάζει στις υπηρεσίες, την ανάπτυξη εφαρμογών που βασίζονται σε υπηρεσίες και τα αποτελέσματα των υπηρεσιών.

Αν και ο όρος υπηρεσιοστρεφής (service-oriented) χρησιμοποιείται εδώ και καιρό μέσα σε διαφορετικά τεχνολογικά πλαίσια για διαφορετικούς σκοπούς, αντικατοπτρίζει ένα νέο τρόπο σκέψης σχετικά με τις επιχειρηματικές διαδικασίες που ενισχύει την αξία της εμπορευματοποίησης, της επαναχρησιμοποίησης, της σημασιολογίας και της πληροφορίας, και δημιουργεί νέες επιχειρηματικές αξίες.

Αρχή της υπηρεσιοστρεφής (service-oriented) λογικής είναι η κατάτμηση μεγάλων και περίπλοκων προβλημάτων σε πολλά μικρότερα και πιο εύκολα διαχειρίσιμα.

Οι περισσότερες επιχειρήσεις δεν έχουν μια συνολική εικόνα των διεργασιών τους, καθώς επίσης και των τεχνολογικών και επιχειρησιακών πολιτικών τους, οι οποίες δεν είναι καλά τεκμηριωμένες (well documented) και τείνουν να μην τηρούνται πάντοτε. Επίσης, δεν διαθέτουν επιχειρηματικές στρατηγικές οι οποίες να είναι άμεσα συνδεδεμένες με το τμήμα του IT. Σε μια εποχή που οι απαιτήσεις έχουν αυξηθεί λόγω της παγκοσμιοποίησης, της αύξησής εσωτερικών και εξωτερικών κανονισμών και των απαιτήσεων συμμόρφωσης (compliance requirements) οι οργανισμοί οφείλουν να προβούν σε διαθρωτικές, λειτουργικές και πολιτιστικές αλλαγές με σκοπό να ενσωματώσουν τα οφέλη των υπηρεσιοστρεφών αρχιτεκτονικών.

2.3.2 Ορισμός Υπηρεσιοστρεφών Αρχιτεκτονικών (SOA)

Τόσο ο όμιλος OASIS όσο και το the Open Group έχουν δώσει επίσημους ορισμούς για τις υπηρεσιοστρεφής αρχιτεκτονικές

2.3.2.1 Ο ορισμός του OASIS είναι ο εξής:

Αποτελεί ένα παράδειγμα για την οργάνωση και τη χρησιμοποίηση κατανεμημένων δυνατοτήτων, οι οποίες μπορεί να βρίσκονται υπό τον έλεγχο διαφόρων ιδιοκτησιακών τομέων (ownership domains). Παρέχει ένα ενιαίο μέσο για

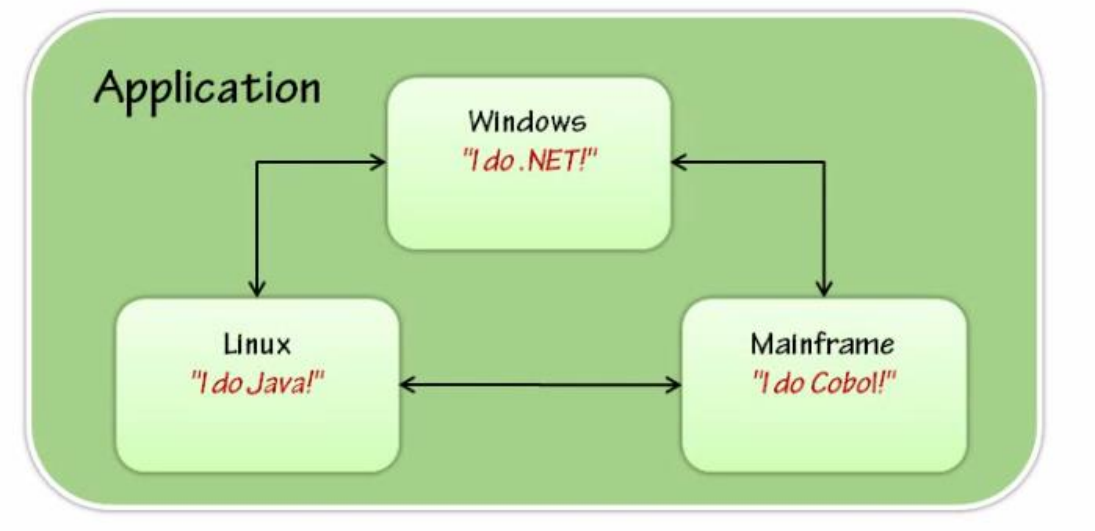
την προσφορά, ανακάλυψή, αλληλεπίδραση και χρήση των δυνατοτήτων με σκοπό την παραγωγή επιθυμητών αποτελεσμάτων τα οποία θα είναι σύμφωνα με μετρήσιμες προϋποθέσεις και προσδοκίες.

2.3.2.2 Ο ορισμός του the Open Group είναι ο εξής:

Οι υπηρεσιοστρεφής αρχιτεκτονικές (Service- Oriented Architecture, SOA) αποτελούν ένα αρχιτεκτονικό ύφος που υποστηρίζει τον προσανατολισμό στην υπηρεσία (service-orientation). Ο προσανατολισμός στην υπηρεσία (service-orientation) είναι ένας τρόπος σκέψης που εστιάζει στις υπηρεσίες, την ανάπτυξη εφαρμογών που βασίζονται σε υπηρεσίες και τα αποτελέσματα των υπηρεσιών. Ενώ, μια υπηρεσία (service) είναι μια λογική αναπαράσταση μια επαναλαμβανόμενης επιχειρηματική δραστηριότητα που έχει ένα συγκεκριμένο αποτέλεσμα

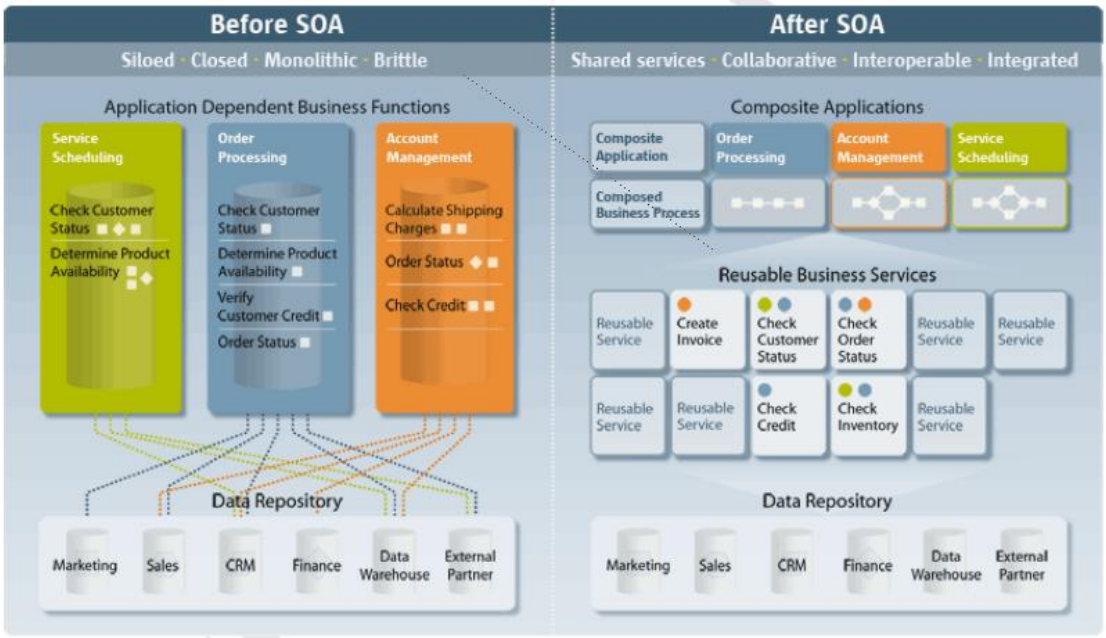
2.3.3 Γιατί Υπηρεσιοστρεφής Αρχιτεκτονική (SOA)

Η υπηρεσιοστρεφής αρχιτεκτονική (SOA) προσφέρει μια πρακτική και βιώσιμη προσέγγιση όσον αφορά την διερεύνησή των υπηρεσιών (services) σε σχέση με τις ανάγκες των επιχειρήσεων. Όσον αφορά την τεχνολογία της πληροφορίας (Information Technology, IT), παρέχει ένα πλαίσιο εργασίας (framework) για την εμπορευματοποίηση του υλικού, του λογισμικού, και τις επιχειρηματικής διαδικασίας. Ο προσανατολισμός στις υπηρεσίες έχει εξελιχθεί σε ένα πρωταρχικό αρχιτεκτονικό μοντέλο της πληροφορικής το οποίο χρησιμοποιείται από πληθώρα οργανισμών με στόχο να μετατρέψουν τις υπάρχουσες αρχιτεκτονικές σε πιο χαλαρά συνδεδεμένες, ώστε να υποστηρίζουν ευέλικτες διαδικτυακές υπηρεσίες (IT Services), όπως φαίνεται στο Σχήμα 2.



Εικόνα 2 Δια-λειτουργικότητα μεταξύ των συνδεδεμένων συστημάτων

- **Το πριν**: Τα συμβατικά αρχιτεκτονικά μοντέλα οι επιχειρηματικές δραστηριότητες, οι εφαρμογές και τα δεδομένα είναι “κλειδωμένα” σε ανεξάρτητα και ασύμβατα μεταξύ τους συστήματα, τα οποία είναι δαπανηρά στην συντήρηση τους και υποχρεώνουν τους χρήστες να περιηγηθούν σε χωριστά δίκτυα, εφαρμογές και βάσεις δεδομένων προκειμένου να εκτελέσουν συγκεκριμένες εργασίες. Επιπλέον, τα συστήματα αυτά έχουν φτιαχτεί για να καλύψουν συγκεκριμένες ανάγκες, αν στην πορεία προκύψουν ανάγκες αλλαγής ή επεκτάσεις αυτό θα είναι εξαιρετικά κοστοβόρο και χρονοβόρο ή ακόμα και αδύνατο.
- **Το μετά**: Οι χρήστες πλέον δεν χρειάζεται να αναζητούν δεδομένα σε πολλά και ξεχωριστά δίκτυα, εφαρμογές, βάσεις δεδομένων κ.τ.λ., τα οποία μετά θα συνθέσουν μόνοι τους για την εξαγωγή συμπερασμάτων, καθώς αυτά ενοποιούνται σε μια καθολική εφαρμογή, ευέλικτη και επεκτάσιμη, μέσω του δικτύου της επιχείρησης ή του διαδικτύου, η οποία παρέχει ευκολία και ταχύτητα χωρίς να καταργεί τα ήδη υπάρχοντα συστήματα.



Εικόνα 3 Πριν και μετά την SOA (Πηγή: SUN, n.d)

Για την επίτευξη της δια-λειτουργικότητας κατά μήκος των συνδεδεμένων συστημάτων μιας ή πολλών διαφορετικών επιχειρήσεων, εφαρμόζονται οι γνωστές σε όλους διαδικτυακές υπηρεσίες (web services ή services).

2.3.4 Τα χαρακτηριστικά των υπηρεσιοστρεφών υπηρεσιών (SOA)

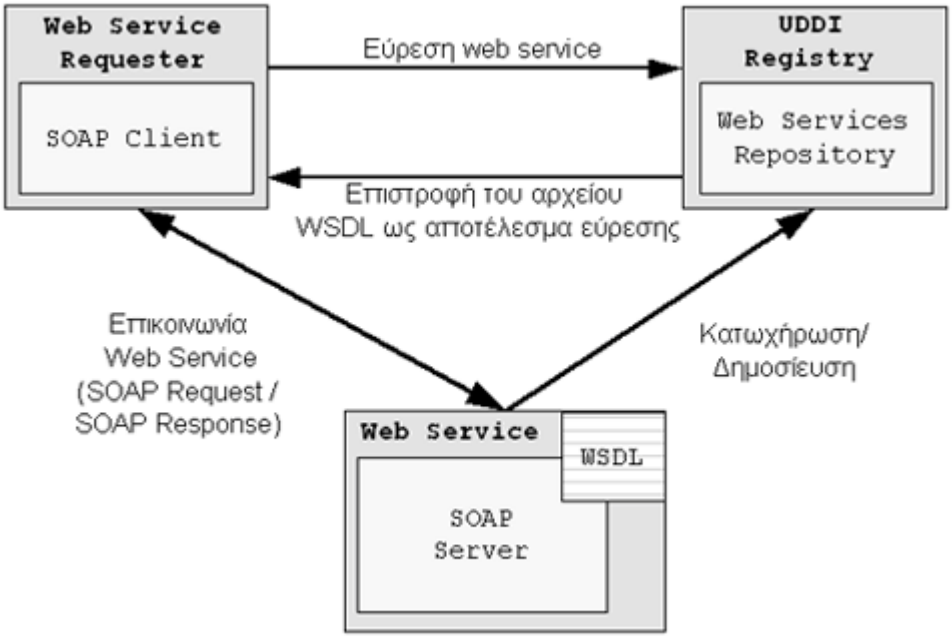
Η υπηρεσιοστρεφής αρχιτεκτονική υποστηρίζει την χαλαρή διασύνδεση μεταξύ των μονάδων που την αποτελούν (components), ώστε να επιτευχθεί η επαναχρησιμοποίησή τους. Οι τρις βασικοί ρόλοι που διακρίνονται στις υπηρεσιοστρεφής αρχιτεκτονικές είναι οι εξής:

Πάροχος Υπηρεσίας (Service provider): Ο πάροχος μιας υπηρεσίας εφαρμόζει μία διεργασία και την καθιστά διαθέσιμη στο διαδίκτυο.

Αιτών Υπηρεσίας (Service requestor): Ο αιτών μίας υπηρεσίας στην ουσία είναι ένας πελάτης (client) ή μία υπηρεσία που χρησιμοποιεί μία άλλη στέλνοντάς της ένα αίτημα.

Μητρώο Υπηρεσίας (Service registry): Μία υπηρεσία μητρώου είναι ένας κοινός κεντρικός χώρος που περιέχει μία λίστα από υπηρεσίες ιστού που έχουν δημιουργήσει εταιρίες ή οργανισμοί.

Ο ρόλος του αιτών υπηρεσίας (service requestor) βρίσκεται ενδιάμεσα στους άλλους δύο έχοντας την ικανότητα να αιτείται δεδομένα και πληροφορίες από ένα πάροχο υπηρεσίας (service provider), αλλά και να καλεί μία υπηρεσία από ένα μητρώο υπηρεσίας (registry service), όπως φαίνεται στο Σχήμα 1.



Σχήμα 1 Αρχιτεκτονική των SOA

Αναλύοντας το παραπάνω σχήμα, παρατηρείται ότι οι υπηρεσιοστρεφής αρχιτεκτονικές ακουσθούν το μοντέλο αναζήτηση (find) – δέσμευση (bind) - εκτέλεση (execute). Για να καταστεί μια υπηρεσία (service) διαθέσιμη προς εκτέλεση από τους πελάτες (clients) θα πρέπει πρώτα να έχει δηλωθεί από τον πάροχο της υπηρεσίες (service provider) σε ένα δημόσιο μητρώο (service registry). Το μητρώο αυτό αποτελεί μια συλλογή από υπηρεσίες οι οποίες συνοδεύονται από έναν αφηρημένο (abstract) προσδιορισμός, οποίος υποστηρίζεται από την προδιαγραφή WSDL και διασφαλίζει ότι οποιαδήποτε υπηρεσία ή client θελήσει να χρησιμοποιήσει ή να δεσμεύσει (bind) την υπηρεσία, να μπορεί να το πράξει με τον καταλληλότερο τρόπο, καθώς και μια ηλεκτρονική διεύθυνση

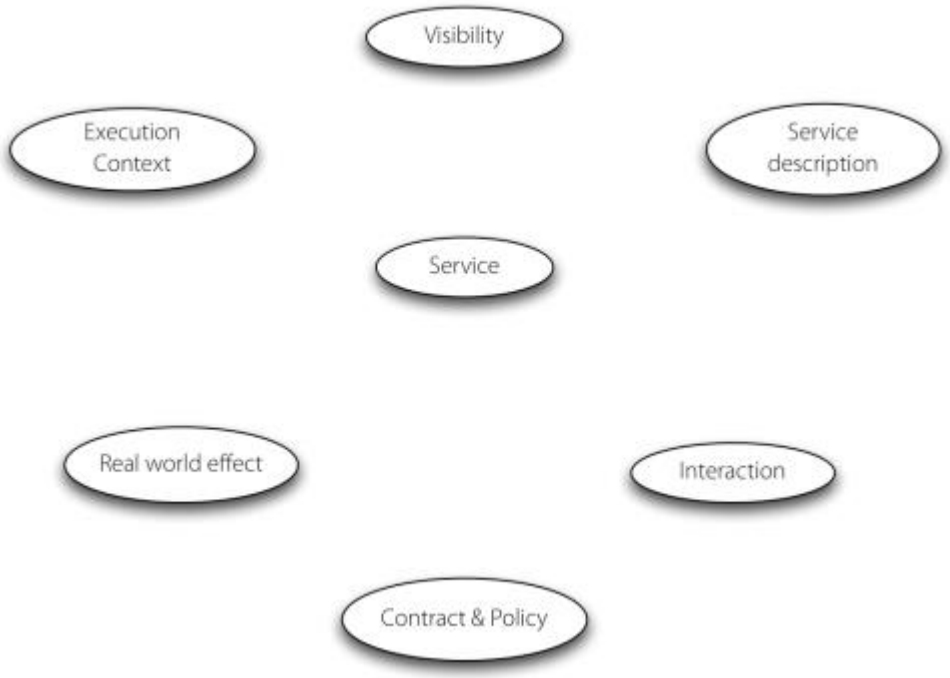
Το μητρώο χρησιμοποιείται από τους καταναλωτές προς αναζήτηση υπηρεσιών (services) που καλύπτουν τα κριτήρια απαιτήσεων που αυτοί θέτουν. Αν το μητρώο περιέχει την υπηρεσία που αναζητά ένας πελάτης τότε του παρέχεται η ηλεκτρονική διεύθυνση στην οποία βρίσκεται. Τότε ο καταναλωτής έχει πλέον πρόσβαση στην εν λόγο υπηρεσία και μπορεί να καλέσει τις μεθόδους και να πάρει τα αποτελέσματα που επιθυμεί.

Το μοντέλο αναφοράς που ορίζει το OASIS για τις υπηρεσιοστρεφής υπηρεσίες (SOA) προσδιορίζει έξι κύριες έννοιες που πρέπει να πληροί μια υπηρεσία ώστε να καταναλωθεί:

- **Ορατότητα (Visibility)**: Για να υπάρξει αλληλεπίδραση μεταξύ του καταναλωτή (service consumer) και του πάροχου τις υπηρεσίας (service provider) πρέπει να “βλέπουν” ο ένας τον άλλο. Αυτό ισχύει για κάθε σχέση καταναλωτή/παρόχου όμως, στην περίπτωση της SOA είναι σημαντικό να δοθεί ιδιαίτερη έμφαση καθώς δεν είναι απαραίτητα εμφανές πως οι συμμετέχοντες στην διαδικασία μπορούν να “δουν” ο ένας τον άλλο. Οι προϋποθέσεις για την ορατότητα είναι οι εξής:
- **Επίγνωση (awareness)**: ο εκκινητής μιας αλληλεπίδρασης πρέπει να γνωρίζει τα άλλα μέρη
- **Προθυμία (willingness)**: οι συμμετέχοντες πρέπει να είναι προδιατεθειμένοι να αλληλοεπιδράσουν
- **Πρόσβαση (reachability)**: οι συμμετέχοντες πρέπει να είναι σε θέση να αλληλοεπιδράσουν
- **Ρεαλιστικό αποτέλεσμα (Real World Effect)**: Είναι το φυσικό αποτέλεσμα από την χρήση μιας υπηρεσίας. Ο καταναλωτής της υπηρεσίας προσπαθεί να έχει κάποιο θεμιτό αποτέλεσμα από την χρήση μιας υπηρεσίας. Ένα ρεαλιστικό αποτέλεσμα μπορεί να είναι η επιστροφή της απάντησης στην αναζήτηση μιας πληροφορίας ή η αλλαγή κατάστασης κάποιων ορισμένων οντοτήτων που διαμοιράζονται μεταξύ των συμμετεχόντων στην υπηρεσία.
- **Αλληλεπίδραση (Interaction)**: Είναι η διαδικασία της εκμετάλλευσης μιας προσφερόμενης δυνατότητας, συνήθως πέραν των ορίων ιδιοκτησίας, με σκοπό την επίτευξη κάποιου συγκεκριμένου ρεαλιστικού αποτελέσματος. Η

αλληλεπίδραση συνήθως γίνεται με την αποστολή και την λήψη μηνυμάτων, δεν είναι όμως ο μοναδικός τρόπος.

- **Περιγραφή υπηρεσίας (Service Description)**: Είναι η απαραίτητη πληροφορία για να μπορέσει κάποιος καταναλωτής να ανακαλύψει και να χρησιμοποιήσει μια υπηρεσία. Σκοπός της περιγραφής είναι η διευκόλυνση της αλληλεπίδρασης και της προβολής της υπηρεσίας, κυρίως όταν οι καταναλωτές της προέρχονται από διαφορετικές διαχειριστικές περιοχές.
- **Περιβάλλον εκτέλεσης (Execution Context)**: Είναι το σύνολο των τεχνικών και επιχειρησιακών στοιχείων που δημιουργούν ένα μονοπάτι μεταξύ αυτών που διαθέτουν τους πόρους και αυτών που τους καταναλώνουν, επιτρέποντας έτσι την αλληλεπίδραση. Οι συμμετέχοντες πρέπει να συμφωνήσουν και να αναγνωρίσουν ένα σύνολο συμφώνων για να έχουν μια επιτυχημένη αλληλεπίδραση μεταξύ των υπηρεσιών, δηλαδή ένα ρεαλιστικό αποτέλεσμα.
- **Πολιτικές και συμβόλαια (Policies & Contracts)**: Μια πολιτική αντιπροσωπεύει ένα είδος περιορισμού και προϋποθέσεων στην χρήση, εγκατάσταση ή περιγραφή μιας υπηρεσίας μεταξύ των συμμετεχόντων. Οι πολιτικές συνήθως εφαρμόζονται σε αρκετές πτυχές της SOA, όπως η ασφάλεια, η ιδιωτικότητά, η διαχειρισιμότητα, η ποιότητα των υπηρεσιών και ου το καθεξής. Το συμβόλαιο από την άλλη μεριά αντιπροσωπεύει την συμφωνία μεταξύ δύο ή περισσοτέρων συμβαλλόντων. Όπως και στις πολιτικές οι συμφωνίες πάλι αφορούν την χρήση μιας υπηρεσίας, επίσης μπορεί να περιορίσουν το αναμενόμενο ρεαλιστικό αποτέλεσμα που προκύπτει από την χρήση μιας υπηρεσίας.



Σχήμα 2 Κύριες έννοιες του μοντέλου αναφοράς

2.3.5 Ισχυρή και χαλαρή συνδεσιμότητα

Στην τεχνολογία λογισμικού (software engineering), ως συνδεσιμότητα (coupling) ορίζεται ο τρόπος και ο βαθμός αλληλεξάρτησης μεταξύ των ενοτήτων ενός λογισμικού (software modules), ένα μέτρο για το πόσο στενά συνδεδεμένες είναι δύο ρουτίνες ή ενότητες.

Η συνδεσιμότητα είναι συνήθως αντίθετη με την συνοχή (cohesion). Χαμηλή συνδεσιμότητα (loose coupling) συνεπάγεται υψηλή συνοχή και το αντίστροφο. Η χαμηλή συνδεσιμότητα αποτελεί ένδειξη ενός καλά δομημένου και σχεδιασμένου υπολογιστικού συστήματος και όταν συνδυάζεται με υψηλή συνοχή επιτυγχάνει υψηλής αναγνωσιμότητας και συντήρηση

Σε ένα παραδοσιακό κατανεμημένο σύστημα παρατηρούνταν προβλήματα σχετικά με την κλιμάκωση (scaling), την υποστήριξη ετερογενών εφαρμογών ,συστημάτων, κ.τ.λ., καθώς και την συντήρηση του κώδικα των υπαρχόντων εφαρμογών. Αυτά τα προβλήματα έπληττων την ευελιξία και ήταν το αποτέλεσμα της ισχυρής συνδεσιμότητας (tight coupling) μεταξύ των κατανεμημένων εφαρμογών. Μια ισχυρή εξάρτηση κάνει τις εφαρμογές λιγότερο ανεκτικές στα σφάλματα.

Η χαλαρή συνδεσιμότητα (loose coupling) είναι η προσέγγιση που συνήθως χρησιμοποιείται για την αντιμετώπιση των απαιτήσεων της κλιμάκωσης, της ευελιξίας, της ανοχής στα σφάλματα και της δυνατότητας αντικατάστασης και επέκτασης. Η χαλαρή συνδεσιμότητα οδηγεί στην χαλαρή εξάρτηση δίνοντας σε ένα σύστημα μεγαλύτερη ευελιξία και μικρότερες επιπτώσεις σε τυχόν τροποποιήσεις ή λάθη. Η υπηρεσιοστρεφής αρχιτεκτονικές και οι διαδικτυακές υπηρεσίες (Web services), προάγουν την χαλαρή συνδεσιμότητα.

Στον παρακάτω πίνακα φαίνονται συνοπτικά οι διαφορές μεταξύ της ισχυρής και της χαλαρής συνδεσιμότητας για τα κατανεμημένα συστήματα.

	Ισχυρή Συνδεσιμότητα (Tight coupling)	Χαλαρή Συνδεσιμότητα (Loose coupling)
physical connections	Point-to-point	Via mediator
Communication style	Synchronous	Asynchronous
Data model	Common complex types	Simple common types only
Type system	Strong	Weak
Interaction pattern	Navigate through complex object trees	Data-centric, self-contained message
Control of process logic	Central control	Distributed control
Binding	Statically	Dynamically
Platform	Strong platform dependencies	Platform independent
Transactionality	2PC (two-phase commit)	Compensation
Deployment	Simultaneous	At different times
Versioning	Explicit upgrades	Implicit upgrades

Table 1 Σύγκριση ισχυρής και χαλαρής συνδεσιμότητας (Πηγή: Josuttis, 2007)

Αν και τα μειονεκτήματα της ισχυρής συνδεσιμότητας είναι αρκετά δεν αποτελεί μια κακή πρακτική, εν αντιθέσει σε κάποιες περιπτώσεις η χρήση της δίνει καλύτερα αποτελέσματα, όπως όταν οι εξαρτήσεις είναι κρίσιμης σημασίας για το σχεδιασμό.

2.3.6 Αρχιτεκτονική Δομή

Μέχρι προσφάτως τα περισσότερα επιχειρησιακά συστήματα χωρίζονταν σε δύο επίπεδα: την βάση δεδομένων και τις δικτυακές Διεπαφές Προγραμματισμού Εφαρμογών (Application programming interface, API). Και τα δύο αυτά επίπεδα ήταν προσβάσιμη από τον χρήστη. Αυτή η αρχιτεκτονική δομή δεν μπορεί να υποστηρίξει ευέλικτα APIs, δεν παρέχει επαρκή απομόνωση του κώδικα της εφαρμογής από τον πελάτη με αποτέλεσμα να μην είναι ευέλικτο και εύκολα κλιμακούμενο κατά την χρήση από πολλούς ταυτόχρονους χρήστε. Κατά συνέπεια, αν και αυτή είναι μια δοκιμασμένη αρχιτεκτονική είναι αποτελεσματική κυρίως σε πρωτότυπα και μικρού μεγέθους συστήματα

Αυτή τη στιγμή το πιο διαδεδομένο μοντέλο ανάπτυξης εφαρμογών βασίζεται σε τρία επίπεδα αρχιτεκτονικής δομής. Το μοντέλο αυτό υποστηρίζει ένα συμπληρωματικό στρώμα μεταξύ των στρώματος του πελάτη και του στρώματος της βάσης δεδομένων. Το πρόσθετο αυτό στρώμα, ονομάζεται στρώμα της επιχειρηματικής λογικής και προσφέρει απομόνωση του κώδικα από τον πελάτη και διαμοιρασμό της λογικής της εφαρμογής μεταξύ των διαφόρων εφαρμογών πελάτη. Πρόκειται για μια επάξια

Οι λύσεις που εφαρμόζουν υπηρεσιοστρεφής αρχιτεκτονικές επιχειρούν να βοηθήσουν τους επιχειρησιακούς στόχους και παράλληλα να δομήσουν ένα σύστημα που να ανταποκρίνεται στις απαιτήσεις ενός οργανισμού. Οι υπηρεσιοστρεφής αρχιτεκτονικές εξετάζονται σε πέντε οριζόντια επίπεδα:

- **Επίπεδο διασύνδεσης με των καταναλωτή (Consumer Interface Layer)** – Το επίπεδο αυτό αποτελείται από το γραφικό περιβάλλον (GUI) που θα

διαχειρίζεται ο τελικός χρήστης ή/και το περιβάλλον μέσα από το οποίο θα έχουν πρόσβαση οι εφαρμογές/υπηρεσίες.

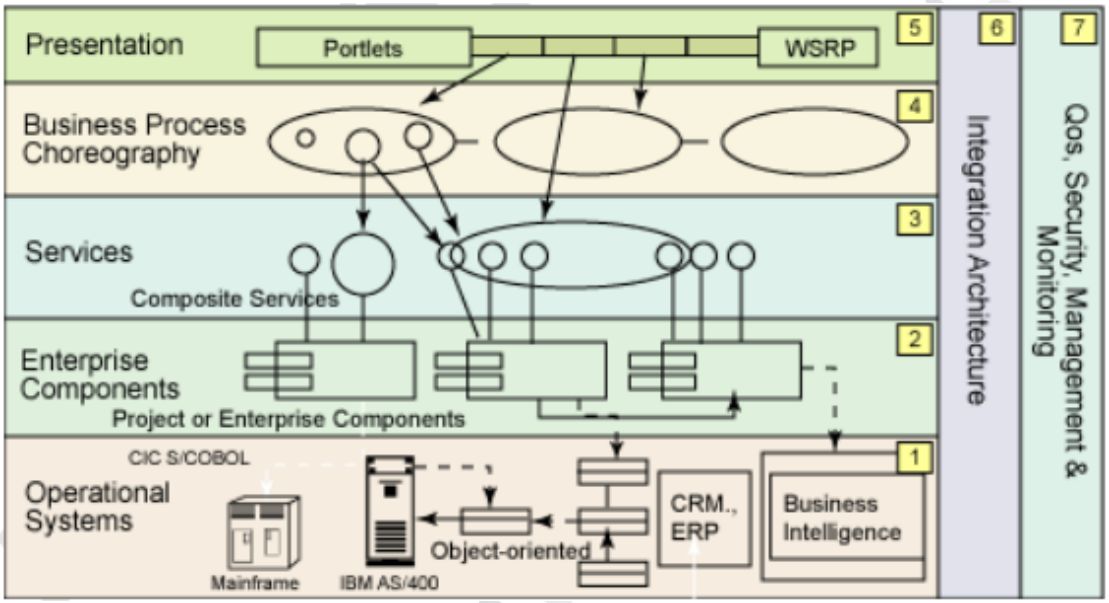
- **Επίπεδο Επιχειρησιακών Διαδικασιών (Business Process Layer)** – Οι υπηρεσίες μπορούν να ενωθούν και να αλληλοεπιδράσουν σαν να είναι μια ενιαία εφαρμογή μέσω της ενορχήστρωσης ή της χορογραφίας υπηρεσιών (services) οι οποίες αναπαριστούν επιχειρηματικές περιπτώσεις χρήσης από άποψη εφαρμογών.
- **Υπηρεσίες (Services)** – Αυτό το στρώμα περιέχει τις πραγματικές υπηρεσίες που μπορεί να είναι εντοπίσιμες και να κληθούν από άλλες εφαρμογές για να παράσχουν τις ειδικές επιχειρηματικές λειτουργίες των επιχειρήσεων. Οι υπηρεσίες αναπτύσσουν τις διασυνδέσεις των επιχειρησιακών στοιχείων καθώς οι περιγραφές τους δημοσιεύονται στο Διαδίκτυο. Συνήθως χρησιμοποιούνται τα Web Services το στρώμα.
- **Στοιχεία μιας υπηρεσίας (Service Components)** – Αυτά είναι εξειδικευμένα δομικά στοιχεία (components) που παρέχουν στις υπηρεσίες συγκεκριμένες λειτουργίες και καλύπτουν συγκεκριμένες απαιτήσεις τους, δηλαδή οι λειτουργικές και τεχνικές βιβλιοθήκες , τεχνολογικές διασυνδέσεις (interfaces) κ.ο.κ.
- **Λειτουργικά Συστήματα (Operational Systems)** – Αυτό το στρώμα περιλαμβάνει τα μοντέλα δεδομένων, αποθήκη δεδομένων των επιχειρήσεων , τεχνολογικές πλατφόρμες κ.λπ.

Υπάρχουν τέσσερα εγκάρσια κάθετα στρώματα (cross-cutting vertical layers), καθένα από τα οποία εφαρμόζονται και υποστηρίζονται από καθεμία από τις προηγούμενες οριζόντιες στρώσεις :

- **Επίπεδο Ενσωμάτωσης (Integration Layer)** – Αυτό το στρώμα επιτρέπει την ενσωμάτωση των υπηρεσιών με τη θέσπιση ενός αξιόπιστου συνόλου δυνατοτήτων όπως είναι η ευφυής δρομολόγηση, το πρωτόκολλο της διαμεσολάβησης και άλλοι μηχανισμοί μετατροπής. Το σύνολο αυτό συχνά περιγράφεται ως Enterprise Service Bus (ESB).
- **Διασφάλιση Ποιότητας (Quality of Service)** – Η ασφάλεια (Security), η διαθεσιμότητα (availability), η επίδοση (performance) συνιστούν την ποιότητα

των παρεχόμενων υπηρεσιών που έχουν διαμορφωθεί με βάση τα απαιτούμενα SLAs , OLAs.

- **Πληροφοριακό (Informational)** – Παρέχει στις επιχειρήσεις πληροφορίες.
- **Διακυβέρνηση (Governance)** – Στρατηγική του IT διέπεται σε κάθε οριζόντιο στρώμα για να επιτευχθεί απαιτούμενο λειτουργικό.

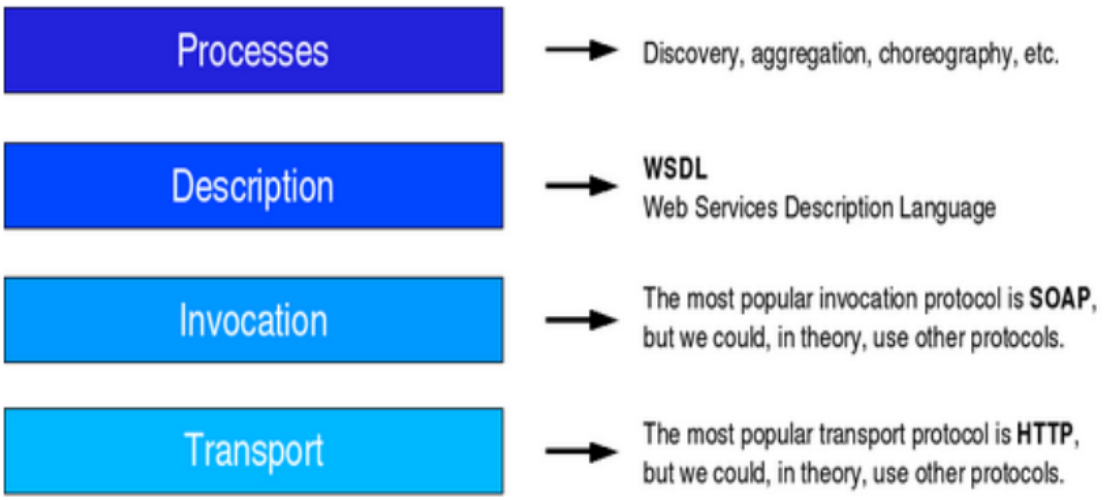


Σχήμα 3 Τα στρώματα της SOA αρχιτεκτονικής

Το δομικά στοιχεία μιας υπηρεσιοστρεφής αρχιτεκτονικής εκτείνονται από το ανώτερο επιχειρησιακό επίπεδο (top-layer business), και τις επιχειρησιακής διαδικασίες, στο μεσαίο υπηρεσιακό επίπεδο (mid-layer services) και φτάνει μέχρι τα κατώτερα επίπεδα και τις υπάρχουσες εφαρμογές και συστήματα.

- **Επίπεδο μεταφοράς (Transport layer)**: Το επίπεδο αυτό είναι υπεύθυνο για τη μεταφοράς των μηνυμάτων μεταξύ των υπηρεσιών. Τα πρωτόκολλα που χρησιμοποιούνται είναι το HTTP, το SMPT και το FTP.
- **Επίπεδο επίκλησης (Invocation layer)**: Το επίπεδο αυτό είναι υπεύθυνο για την σύνταξη των μηνυμάτων της υπηρεσίας χρησιμοποιώντας το κοινό πρότυπο της XML. Η κάψουλα μεταφοράς των μηνυμάτων όμως και η δομική μονάδα των υπηρεσιών ιστού είναι το Simple Object Access Protocol–SOAP.

- **Επίπεδο περιγραφής (Description layer)**: Στο επίπεδο αυτό περιγράφεται η λειτουργία που εφαρμόζει μία υπηρεσία ιστού μέσω της περιγραφικής γλώσσας υπηρεσιών ιστού – WSDL.
- **Επίπεδο διαδικασίας (Process layer)**: Στο επίπεδο αυτό βρίσκονται περισσότερες από μία υπηρεσίες ιστού. Μία από αυτές είναι η "ανακάλυψη" (discovery) μιας υπηρεσίας που γίνεται μέσω του προτύπου UDDI (Universal Description Discovery Integration).



Εικόνα 4 Τα επίπεδα που καλύπτουν οι υπηρεσιοστρεφής αρχιτεκτονικές

2.3.7 Πλεονεκτήματα Υπηρεσιοστρεφών Αρχιτεκτονικών

Η πρόκληση που αντιμετωπίζουν όλοι οι οργανισμοί στις μέρες μας είναι να καταφέρουν να ανταπεξέλθουν με επιτυχία σε ένα δυναμικό και συνεχώς μεταβαλλόμενο επιχειρηματικό περιβάλλον, διατηρώντας το ανταγωνιστικό τους πλεονέκτημα ως αφορά την σχέση κόστους – αποτελεσματικότητας παράλληλα με μια υψηλή Απόδοση Επενδύσεων (Return on Investment, ROI). Εδώ έρχονται να βοηθήσουν οι υπηρεσιοστρεφής αρχιτεκτονικές προσφέροντας:

- **Διαλειτουργικότητα**: Η χαλαρή σύζευξή μεταξύ των υπηρεσιών (services) οδηγεί στην ανάπτυξη ευέλικτου λογισμικού στα κατανεμημένα πληροφοριακά περιβάλλοντα.

- **Ενσωμάτωση**: Επιτρέπουν την εύκολη και γρήγορη ενσωμάτωση νέων επιχειρηματικών διαδικασιών ανάλογα με τις ανάγκες που προκύπτουν, μειώνοντας σημαντικά το κόστος ανάπτυξης λογισμικού και αυξάνοντας την εναρμόνιση του οργανισμού με τις απαιτήσεις της αγοράς. Αυτό επιτυγχάνεται διότι οι νέες υπηρεσίες επαναχρησιμοποιούν τις υφιστάμενες και τα συστατικά τους μειώνοντας έτσι τον χρόνο που χρειάζεται για τον σχεδιασμό, την ανάπτυξη, την δομική και την εγκατάσταση/υιοθέτηση.
- **Επεκτασιμότητα**: Έχουν την δυνατότητα εκσυγχρονισμού πληροφοριακών συστημάτων που έχουν δημιουργηθεί με παλαιές τεχνολογίες.
- **Επαναχρησιμοποίηση**: Οι υπηρεσίες (services) έχουν δημιουργηθεί έτσι ώστε να μπορούν να επαναχρησιμοποιηθούν και να βελτιστοποιηθούν με ευκολία ενσωματώνοντας νέα χαρακτηριστικά. Αυτό ελαχιστοποιεί την προσπάθεια που χρειάζεται για την ανάπτυξη νέων υπηρεσιών-εφαρμογών έχοντας ως φυσικό επακόλουθο την μείωση του κόστους.
- **Αναβάθμιση**: Περιέχουν την περιγραφή στο εσωτερικό τους οπότε είναι εύκολο να χρησιμοποιηθούν από πληροφοριακά συστήματα και να τα αναβαθμίσουν.
- **Μόχλευση των υπαρχόντων στοιχείων-δομών**: Οι υπάρχουσες επενδύσεις στα πληροφοριακά συστήματα μπορούν να χρησιμοποιηθούν σε συνδυασμό με την SOA, επειδή η SOA παρέχει ένα πιο αφηρημένο επίπεδο το οποίο περικλείει τα υπάρχοντα στοιχεία-δομές σαν υπηρεσίες που παρέχουν επιχειρηματικές λειτουργίες. Η χρήση των νέων αυτών υπηρεσιών είναι πολύ απλή και προϋποθέτει ο πελάτης να ξέρει μόνο την διεπαφή και το όνομα της. Αυτό έχει ως αποτέλεσμα οι οργανισμοί να μπορούν να συνεχίσουν να λαμβάνουν προστιθέμενη αξία από τους υπάρχοντες πόρους χωρίς να χρειάζεται μεγάλης κλίμακας ενός πλήρους συστήματος. Είναι λογικό λοιπόν το κόστος που χρειάζεται ένα σύστημα για συναντήσει τις τρέχουσες απαιτήσεις να είναι μειωμένο.
- **Χρήση έξω-επιχειρησιακών πόρων**: Η υπηρεσιοστρεφής αρχιτεκτονική απλοποιεί την ενσωμάτωση συστημάτων μεταξύ εταιριών (Business-to-business Integration) επιτρέποντας την σύνδεση του οργανισμού με εξωτερικούς εταίρους και δίνοντάς του την δυνατότητα να έχει πρόσβαση σε

εξειδικευμένες υπηρεσίες, μειώνοντας το κόστος και τον χρόνο υλοποίησης, αλλά διατηρώντας τον έλεγχο και την παρακολούθηση σημαντικών διαδικασιών.

- **Μετασχηματισμός των δεδομένων**: Με την υπηρεσιοστρεφής αρχιτεκτονική μπορεί να παρασχεθεί μετατροπή δεδομένων από μια μορφή σε μία άλλη, μέσω αυτοματοποιημένων αντιστοιχίσεων πεδίων. Η μετατροπή δεδομένων από εξειδικευμένο λογισμικό είναι αρκετά δαπανηρή, σε αντίθεση με την χρήση γενικών (generic) μεταφραστών δεδομένων μπορεί να επιφέρει σημαντική μείωση κόστους.
- **Καλύτερη ευθυγράμμιση μεταξύ επιχείρησης και πληροφοριακών συστημάτων**: Η ευθυγράμμιση επιτυγχάνεται όταν όλες τις λειτουργίες των επιχειρήσεων (συμπεριλαμβανομένων και των IT) υποστηρίζουν κοινούς στόχους και αποτελέσματα. Οι υπηρεσίες που παρέχονται από το IT ενός οργανισμού που χρησιμοποιούν μια προσέγγιση υπηρεσιοστρεφών αρχιτεκτονικών μπορούν και πρέπει να καθοριστούν έτσι ώστε να υποστηρίζουν άμεσα τις υπηρεσίες που ο οργανισμός παρέχει στους πελάτες, τους πολίτες και τους εταίρους. Χρησιμοποιώντας τις υπηρεσίες (services) και τη υπηρεσιοστρεφής αρχιτεκτονική για την ευθυγράμμιση των επιχειρήσεων οδηγεί στη βελτίωση του σχεδιασμού και της ανάπτυξης των υπηρεσιών. Αυτό επιτυγχάνεται με τον εξορθολογισμό της επικοινωνίας και τις μεταφοράς των αναγκών και των απαιτήσεων από τους επιχειρηματικούς χρήστες στους τεχνολόγους IT, επειδή αυτός ο τρόπος βοηθά στην ανύψωση της συζήτησης στο επιχειρησιακό επίπεδο..

2.3.8 Μειονεκτήματα Υπηρεσιοστρεφών Αρχιτεκτονικών

Αν και η υπηρεσιοστρεφής αρχιτεκτονική είναι μια εξαιρετικά δημοφιλής και ταχέως αναπτυσσόμενη αρχιτεκτονική την τελευταία δεκαετία δεν παύει να έχει και αυτή τα μειονεκτήματά της. Κατά τον σχεδιασμό ενός πληροφοριακού συστήματος βάση της υπηρεσιοστρεφής αρχιτεκτονικής υπάρχει πάντα ο κίνδυνος να προβούμε σε κάποια προφανή σφάλματα που στην πορεία μπορεί να μας κοστίσουν.

Η πιο προφανής παγίδα στην οποία μπορεί να πέσει κάποιος που επιχειρεί να σχεδιάσει ένα πληροφοριακό σύστημα βάση των υπηρεσιοστρεφών αρχιτεκτονικών είναι να ακολουθήσει τις πεπατημένες των κατανεμημένων συστημένων Αυτό είναι πιθανότητα το πιο συνηθισμένο λάθος που αντιμετωπίσουν οι οργανισμοί. Στην προσπάθεια τους για την επίτευξη μιας υπηρεσιοστρεφής αρχιτεκτονικής κατασκευάζουν υπηρεσιοστρεφής υπηρεσίες με τον ίδιο τρόπο με τον οποίο κατασκευάζουν παραδοσιακά κατανεμημένα συστήματα. Η υπηρεσιοστρεφής αρχιτεκτονικής δεν είναι, όμως, ίδια με άλλες προσεγγίσεις, ούτε μπορεί τα εφαρμοστούν σε αυτή τα χαρακτηριστικά ισχυρής συνδεσιμότητας.

Δεν υπάρχει προτυποποίηση των υπηρεσιοστρεφών αρχιτεκτονικών. Όπως και σε κάθε άλλη αρχιτεκτονική απαιτείται η επιβολή εσωτερικής προτυποποίησης, δεν μπορεί να σχεδιάζεται και να υλοποιείται μια υπηρεσιοστρεφής αρχιτεκτονική τμηματικά χωρίς να λαμβάνεται υπόψη τα υπόλοιπα τμήματα ενός οργανισμού με τα οποία θα αλληλοεπιδράσει.

Παραλείπεται το σχέδιο μετάβασής. Η επιτυχής μετάβαση σε μια υπηρεσιοστρεφής αρχιτεκτονική εξαρτάται από το αν έχουν μελετηθεί και σχεδιαστεί τα στάδια από τα οποία πρέπει να περάσει ένας οργανισμός ώστε να προσαρμοστεί στα καινούργια δεδομένα καθώς ενδέχεται να χρειαστεί αναθεώρηση του περιβάλλοντος του IT της.

Δεν ξεκινούν με την θεμελιώδη XML αρχιτεκτονική: Αυτή την στιγμή τα πάντα στις υπηρεσιοστρεφής αρχιτεκτονικές ξεκινούν με τα Web services. Η δήλωση αυτή έχει καθιερωθεί σε ορισμένους οργανισμούς, αλλά δεν είναι εντελώς αληθής. Στον κόσμο των υπηρεσιοστρεφών αρχιτεκτονικών σήμερα τα πάντα ξεκινούν με την XML. Είναι το πρότυπο από το οποίο πολλά συμπληρωματικά πρότυπα έχουν εξελιχθεί και έχουν σχηματίσει μια καινούργια αρχιτεκτονική αναπαράσταση δεδομένων. Είναι το βασικό σύνολο προτύπων που έχει τροφοδοτήσει την δημιουργία πολλών βασικών προδιαγραφών (βλέπε WSDL, SOAP) για Web services οι οποίες οδηγούν τώρα στην υπηρεσιοστρεφή αρχιτεκτονική. Έτσι, δίνεται μεγάλη προσοχή στο τρόπο με τον οποίο τα δεδομένα μεταφέρονται μεταξύ των υπηρεσιών, ο τρόπος όμως με τον οποίο αυτά τα ίδια τα δεδομένα είναι δομημένα και επικυρωμένα πίσω από τις γραμμές των υπηρεσιών συχνά παραβλέπεται. Η παράλειψη αυτή μπορεί να

οδηγήσει σε εσφαλμένη εφαρμογή του μόνιμου στρώματος αναπαράστασης XML δεδομένων. Αυτό το στρώμα είναι θεμελιώδες για την υπηρεσιοστρεφή αρχιτεκτονική και οι αδυναμίες του θα έχουν επιπτώσεις σε οποιοδήποτε λύση κατασκευαστεί πάνω σε αυτό.

Δεν αντιλαμβάνονται τις απαιτήσεις επιδόσεων των υπηρεσιοστρεφών αρχιτεκτονικών. Η χαλαρή συνδεσιμότητα έχει το τίμημα όταν κάποιος επιλέγει υπηρεσιοστρεφής αρχιτεκτονικές. Όταν η εφαρμογή της γίνεται με τα Web services η SOA εισάγει νέα στρώματα επεξεργασίας δεδομένων, καθώς και το επιπλέον φορτίο που σχετίζεται με αυτά. Όταν ξεκινάει κάποιος μικρός, είναι εύκολο να κατασκευάσει υπηρεσιοστραφείς λύσεις οι οποίες λειτουργούν και ανταποκρίνονται όπως είχαν σχεδιαστεί. Όσο αυξάνει το εύρος και περισσότερες λειτουργίες, προστίθεται, ο όγκος των μηνυμάτων επικοινωνίας μεγαλώνει. Τότε είναι που τα απροετοίμαστα περιβάλλοντα μπορούν να αρχίσουν αντιμετωπίζουν σημαντικές καθυστερήσεις στην επεξεργασία. Είναι κρίσιμης σημασίας η πρόβλεψη μελλοντικών καταστάσεων, η πλήρης κατανόηση των απαιτήσεων του συστήματος και τέλος η στρεσογόνα δοκιμή με ανάλογα τέτοια σενάρια.

Δεν έχουν σχεδιασμό για την ασφάλεια των Web services. Μια SOA ξεκινάει να αναπτύσσεται σιγά-σιγά μέσα σε ένα οργανισμό, όταν αρχίζει και επεκτείνεται είναι εύκολο για κάποιον να βασίσει την ασφάλεια της ανταλλαγής απλών μηνυμάτων στο οικείο Secure Sockets Layer(SSL). Την στιγμή όμως που οι υπηρεσίες ξεκινήσουν να λαμβάνουν μεγαλύτερες ευθύνες επεξεργασίας το SSL δεν επαρκεί, η ασφάλεια πλέον πρέπει να εφαρμοστεί στο επίπεδο ανταλλαγής μηνυμάτων. Το πλαίσιο WS-Security παρέχει την απαιτούμενη ασφάλεια και ευελιξία που χρειάζεται η SOA. Όταν κάποιος θέλει οι υπηρεσίες του να είναι ασφαλείς και δεν λαμβάνει υπόψητου, το προαναφερθέν πλαίσιο και τις επεκτάσεις του μπορεί να οδηγηθεί σε ακριβούς επαναπρογραμματισμούς των υπηρεσιών του, καθώς και σε ακριβή εκ των υστέρων τοποθέτηση του. Καλό είναι ο προγραμματισμός για την εφαρμογή ασφάλειας να γίνεται εξαρχής.

Δεν διατηρούν επαφή με τις πλατφόρμες προϊόντων και τα πρότυπα ανάπτυξης. Η μετάβαση προς μια SOA που βασίζεται σε Web services ανοίγει το χώρο των προϊόντων και των πλατφορμών μέσα από τον οποίο οι οργανισμοί

μπορούν να επιλέξουν την κατάλληλη λύση. Ενώ η τάση είναι η διεύθυνση πληροφορικής να συνεχίσει με αυτό που ξέρει καλύτερα, η δυνατότητα να κοιτάξει αλλού είναι πάντα παρούσα. Η αρχιτεκτονική SOA ακριβώς επειδή είναι ανεξάρτητη κατασκευαστών-πλατφορμών μετατρέπει σε επιλέξιμη οποιαδήποτε διαθέσιμη λύση υπάρχει και υποστηρίζει Web services. Άλλο ένα χαρακτηριστικό που πρέπει να επισημανθεί είναι ότι στην επιλογή της λύσης από κάποιον κατασκευαστή να έχει υποστήριξη στις προδιαγραφές WS-* που βρίσκονται στο στάδιο της ανάπτυξης.

2.3.9 Διακυβέρνηση SOA

Ένας οργανισμός θεσπίζει την διακυβέρνηση για να αμβλύνει τους κινδύνους και για να συντείνει στην προώθηση της στρατηγικής, των στόχων και τον προτεραιοτήτων του. Περιμένει να υλοποιηθεί το περιεχόμενο της έννοιας της διακυβέρνησης. Όταν υπάρχει διακυβέρνηση υπάρχουν πλαίσια και περιορισμοί και διάφοροι άλλοι παράμετροι για την καθοδήγηση, τον έλεγχο και τον επηρεασμό των αποφάσεων, υπάρχει ανάθεση καθηκόντων, και εποπτεύουσα αρχή για την παρακολούθηση και την επιβολή μέτρων σε ό,τι και όσους δεν συμμορφώνονται. Ένα καλό σύστημα διακυβέρνησης βοηθάει τον οργανισμό και τα μέλη του να πετύχουν τους στόχους τους και το όραμα τους.

Όταν ένας οργανισμός επενδύει στην SOA περιμένει να αποκτήσει οφέλη που αξίζουν περισσότερο από το κόστος υλοποίησης της αρχιτεκτονικής. Το ROI μετράται από τα αποτελέσματα της επιχείρησης και προφανώς αυτά τα αποτελέσματα αντικατοπτρίζουν την στρατηγική, τις προτεραιότητες και τους στόχους της. Ένα καλό σύστημα διακυβέρνησης μπορεί να βοηθήσει στην επίτευξη αυτών των αποτελεσμάτων καθιστώντας προσοδοφόρα μια τέτοια επένδυση.

Υπάρχουν πολλά άρθρα και συστάσεις για την ανάγκη διακυβέρνησης όταν γίνεται εισαγωγή στην SOA, π.χ ο Carter (2007) τονίζει ότι η διακυβέρνηση είναι τόσο σημαντική και απαραίτητη που πρέπει να εμπεριέχεται στον σχεδιασμό και την ανάπτυξη της SOA από την πρώτη μέρα.

Οι ορισμοί για την Διακυβέρνηση της SOA είναι αρκετοί και προέρχονται, από κατασκευαστές λογισμικού, φίρμες αναλυτών, σεβαστούς συγγραφείς και από

σώματα προτυποποίησης. Το The Open Group δίνει τον παρακάτω ορισμό για την Διακυβέρνηση SOA:

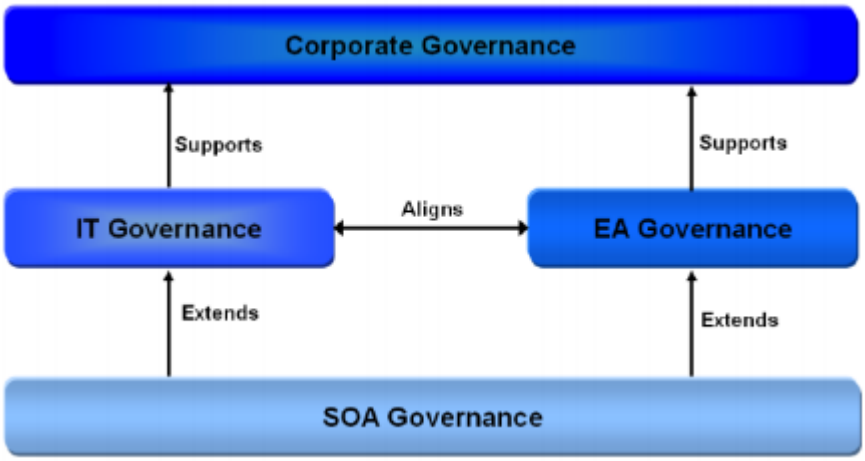
> *"Η Διακυβέρνηση SOA θα πρέπει να θεωρείται ως η εφαρμογή της Εταιρικής Διακυβέρνησης της Διακυβέρνηση IT και της Διακυβέρνηση της EA προς τη Service Oriented Architecture. Στην πραγματικότητα η SOA Διακυβέρνηση επεκτείνετε στην Διακυβέρνηση IT και στην Διακυβέρνηση EA, διασφαλίζοντας ότι πληρούνται τα οφέλη τα οποία εκθειάζει η SOA. Αυτό απαιτεί τη διακυβέρνηση όχι μόνο των πτυχών εκτέλεσης της SOA αλλά και των δραστηριοτήτων του στρατηγικού προγραμματισμού της."*
>
> *The Open Group, 2009 (14)*

Παραθέτει την (Εικόνα 5) και διευκρινίζει πώς:

Η Διακυβέρνηση της Επιχειρησιακή Αρχιτεκτονικής (EA) είναι η πρακτική και ο προσανατολισμός με τον οποίον επιχειρησιακές αρχιτεκτονικές και άλλες αρχιτεκτονικές διαχειρίζονται και ο ελέγχονται σε ένα ευρύ επιχειρησιακό επίπεδο.

Η Διακυβέρνηση των IT περιλαμβάνει τα δικαιώματα αποφάσεων, το πλαίσιο λογοδότησης και τις διαδικασίες, για την ενθάρρυνση της επιθυμητής συμπεριφοράς στην χρήση του IT.

Η Εταιρική Διακυβέρνηση είναι το σύνολο των διαδικασιών, των πολιτικών, των νόμων και των θεσμών που επηρεάζουν τον τρόπο που μια εταιρεία κατευθύνεται, διοικείται ή ελέγχεται.



Εικόνα 5 Οι σχέσεις μιας Διακυβέρνησης SOA (Πηγή: The Open Group, 2009)

2.3.9.1 Ένα μοντέλο Κύκλου Ζωής της Διακυβέρνησης SOA

Για να εφαρμοστεί επιτυχώς η SOA Διακυβέρνηση χρειάζεται ένα μοντέλο κύκλου ζωής. Υπάρχουν πολλοί κύκλοι ζωής για την SOA Διακυβέρνηση καθώς δεν υπάρχει κάποιος που να είναι κοινά αποδεκτός. Ένας κύκλος ζωής σύμφωνα με τον Carter(2007) μπορεί να περιέχει τα στάδια του σχεδιασμού (Plan), του ορισμού (Define), της ενεργοποίησης (Enable) και τέλος το στάδιο των μετρήσεων (Measure).Αναλυτικά:

Στο στάδιο του σχεδιασμού καθορίζεται η ανάγκη για διακυβέρνηση. Το στάδιο αυτό περιλαμβάνει δραστηριότητες όπως:

- Η τεκμηρίωση και επικύρωση της επιχειρηματικής στρατηγικής για τη SOA και το IT.
- Η αξιολόγηση των τρεχουσών δυνατοτήτων της SOA και του IT.
- Ο καθορισμός/τελειοποίηση του οράματος και της στρατηγικής της SOA.
- Η επανεξέταση των τρεχουσών δυνατοτήτων και η οργάνωση της Διακυβέρνησης.
- Το στήσιμο του πλάνου της διακυβέρνησης.

Εν ολίγοις, αυτό το στάδιο θα πρέπει να οδηγήσει στον καθορισμό ενός συνολικού σχεδίου διακυβέρνησης.

Η προσέγγιση που θα προκύψει από το παραπάνω στάδιο θα αντιστοιχιστεί στην συνέχεια στο στάδιο του ορισμού. Το στάδιο αυτό περιλαμβάνει δραστηριότητες όπως:

- Ο ορισμός/αλλαγή των διαδικασιών διακυβέρνησης.
- Ο σχεδιασμός των πολιτικών και των μηχανισμών επιβολής τους.
- Ο εντοπισμός των επιτυχημένων παραγόντων και μετρήσεων.
- Ο εντοπισμός των ιδιοκτητών και ενός μοντέλου χρηματοδότησης.
- Η Ανακήρυξης/τελειοποίηση ενός Κέντρου Αριστείας (Centre of Excellence) SOA.
- Ο σχεδιασμός της διακυβέρνησης της υποδομής του IT.

Στο στάδιο της ενεργοποίησης, αναπτύσσετε βαθμιαία το μοντέλο διακυβέρνησης. Αυτό περιλαμβάνει:

- την ανάπτυξη πολιτικών,
- την ανάπτυξη μηχανισμών διακυβέρνησης,
- και την ανάπτυξη της διακυβέρνησης της υποδομής του IT, που ορίστηκαν στο προηγούμενο στάδιο. Επιπλέον, σε αυτό το στάδιο γίνεται εκπαίδευση και ανάπτυξη πάνω σε αναμενόμενες συμπεριφορές, πρακτικές.

Στο στάδιο μετρήσεων ρυθμίζονται η παρακολούθηση και η διαχείριση της διαδικασίας διακυβέρνησης. Είναι σημαντικό να εξασφαλισθεί η συμμόρφωση με τις πολιτικές και τις ρυθμίσεις της διακυβέρνησης και να ελέγχεται η αποτελεσματικότητα των μετρήσεων του IT. Αυτό το πλαίσιο σχετίζεται με τη θέσπιση των διαδικασιών διακυβέρνησης SOA από άκρη σε άκρη — το πώς οι αποφάσεις και οι πολιτικές λαμβάνονται για τον κύκλο ζωής του SOA.

2.3.10 Κύκλος Ζωής SOA

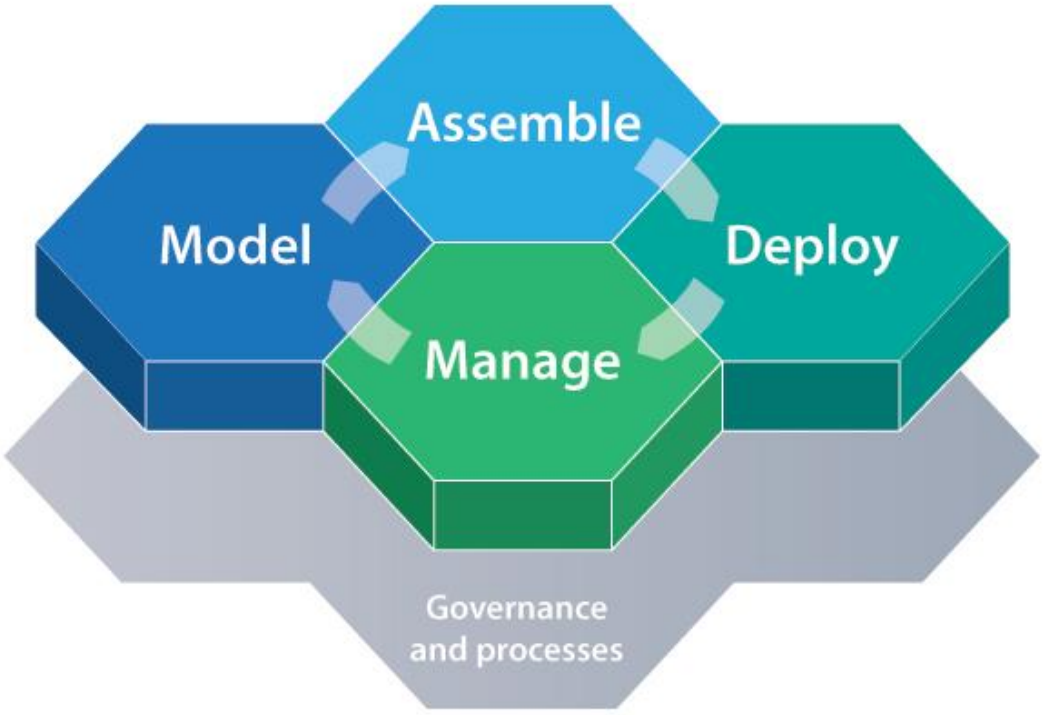
Ο κύκλος ζωής μιας υπηρεσιοστρεφής αρχιτεκτονικής χωρίζεται στα παρακάτω γενικά στάδια

- **Μοντελοποίηση (Model)**: Η φάση της μοντελοποίησης ξεκινά με τη συγκέντρωση και την ανάλυση των επιχειρησιακών απαιτήσεων ενός οργανισμού, τις οποίες κατόπιν χρησιμοποιεί για την δημιουργία μοντέλων, τη διενέργεια προσομοιώσεων και τη βελτιστοποίηση των επιχειρηματικών διαδικασιών. Οι επιχειρηματικές διαδικασίες που προκύπτουν, χρησιμοποιούνται για το σχεδιασμό των σχετικών υπηρεσιών λογισμικού και επιπέδων υπηρεσίας και την υποστήριξη αυτών των διαδικασιών. Κατά τη διάρκεια αυτής της φάσης χρησιμοποιείται ένα μοντέλο για να δημιουργήσει μια κοινή βάση επικοινωνίας μεταξύ της επιχειρησιακής και της πληροφορικής διάστασης επιχειρηματικών διαδικασιών, στόχων και αποτελεσμάτων, αλλά και για να διασφαλιστεί ότι η εφαρμογή που θα προκύψει, ανταποκρίνεται σε όλες τις προκαθορισμένες επιχειρηματικές απαιτήσεις της. Το μοντέλο αυτό παρέχει, επίσης, μια βάση αναφοράς για την αξιολόγηση της επιχειρηματικής απόδοσης.
- **Συγκρότηση (Assemble):** Κατά τη φάση της συγκρότησης η επιχείρηση μπορεί να δημιουργήσει υπηρεσίες, αξιοποιώντας τους υφιστάμενους πόρος, όπως προγραμματισμός επιχειρησιακών πόρων (ERP) και συστήματα οικονομικής διαχείρισης και άλλες λύσεις. Σε πολλές περιπτώσεις μπορεί να χρησιμοποιηθεί μια βιβλιοθήκη για την εύρεση των υπηρεσιών που ήδη υπάρχουν στον οργανισμό. Εάν δεν υπάρχουν τέτοιες λειτουργίες, μπορεί να δημιουργήσει και να δοκιμάσει τις λειτουργίες που απαιτούνται για μια συγκεκριμένη επιχειρηματική διαδικασία. Μόλις οι απαιτούμενες υπηρεσίες γίνουν διαθέσιμες, εναρμονίζονται, ώστε να υλοποιηθεί η επιχειρηματική διαδικασία. Κατά την φάση της συγκρότησης αξιοποιείται πλήρως το μοντέλο των διαδικασιών που δημιουργήθηκε στο στάδιο της μοντελοποίησης. Στο στάδιο αυτό αρχιτεκτονικά εργαλεία επιτρέπουν τη δημιουργία μοντέλων δεδομένων, λειτουργικών ροών, τρόπων αλληλεπίδρασης μεταξύ συστημάτων. Οι ειδικοί στα θέματα ενοποίησης χρησιμοποιούν αυτά τα

εξειδικευμένα εργαλεία για να για να διαμορφώνουν τις αλληλεπιδράσεις μεταξύ των υπηρεσιών σε επιχειρηματικές διαδικασίες.

- **Εφαρμογή (Deploy):** Κατά τη διάρκεια της φάσης της εφαρμογής ο οργανισμός μπορεί να ρυθμίσει και να επεκτείνει το περιβάλλον εκτέλεσης (run-time environment), έτσι ώστε να ανταποκρίνεται στα επίπεδα εξυπηρέτησης που οι επιχειρηματικές διαδικασίες απαιτούν. Αφού μια επιχειρησιακή διαδικασία ρυθμιστεί, μπορεί να κλιμακωθεί σε ένα ισχυρό, με δυνατότητες επέκτασης και υψηλής ασφάλειας, περιβάλλον υπηρεσιών. Αυτό, λοιπόν, το περιβάλλον υπηρεσιών βελτιστοποιείται ώστε να εκτελεί με αξιοπιστία κρίσιμες επιχειρησιακές διαδικασίες, παρέχοντας ταυτόχρονα την ευελιξία για δυναμικές ενημερώσεις, προκειμένου η επιχείρηση να μπορεί να ανταποκρίνεται στις μεταβαλλόμενες επιχειρησιακές απαιτήσεις. Αυτή η προσανατολισμένη στις υπηρεσίες προσέγγιση συμβάλει, επίσης, στη συμπίεση του κόστους και της πολυπλοκότητας που συνεπάγεται η διατήρηση πάρα πολλών ενοποιήσεων point-to-pont.
- **Διαχείριση (Manage):** Η φάση της διαχείρισης αφορά στην εδραίωση και στη διατήρηση της διαθεσιμότητας των υπηρεσιών και των χρόνων απόκρισης, καθώς και στη διαχείριση των υποκείμενων πόρων των υπηρεσιών. Η εταιρία μπορεί να παρακολουθεί τους κύριους δείκτες απόδοσης (KPI) σε πραγματικό χρόνο για να λαμβάνει τις πληροφορίες που χρειάζεται για την πρόληψη, την απομόνωση, τη διάγνωση και την επίλυση προβλημάτων. Η κατανόηση της απόδοσης των επιχειρησιακών διαδικασιών σε πραγματικό χρόνο δίνει τη δυνατότητα να παρέχονται σημαντικές πληροφορίες ώστε να είναι δυνατή η συνεχής βελτίωση στο μοντέλο της επιχειρησιακής διαδικασίας. Αυτή η φάση αφορά, επίσης, στην επανατροφοδότηση κρίσιμων πληροφοριών για τον επανασχεδιασμό του μοντέλου της διαδικασίας. Η φάση διαχείρισης, τελικά, επιτρέπει ταχύτερη λήψη των σωστών επιχειρηματικών αποφάσεων απ’ ό,τι προηγουμένως. Διαχείριση υπηρεσιών πληροφορικής. Οι υπηρεσίες αυτές παρέχουν δυνατότητες που σχετίζονται με το απαιτούμενο επίπεδο κλιμάκωσης και απόδοσης συμπεριλαμβανομένων υπηρεσιών edge (edge services), υπηρεσιών δημιουργίας συμπλεγμάτων (clustering) και

δυνατοτήτων εικονικοποίησης για να είναι δυνατή η αποδοτική χρήση των πόρων πληροφορικής βάσει μοντέλων φόρτου εργασίας.



Εικόνα 6 Ο κύκλος ζωής μια υπηρεσιοστρεφής αρχιτεκτονικής

Τα παραπάνω τέσσερα στάδια μπορούν να αναλυθούν περαιτέρω στα παρακάτω μικρότερα και πιο εύκολα διαχειρίσιμα επτά στάδια:

- **Στάδιο ανάπτυξης (Develop)**. Σε αυτό το στάδιο οι οργανισμοί σχεδιάζουν και κατασκευάζουν υπηρεσίες που ανταποκρίνονται στα ακριβή βήματα μιας επιχειρησιακής διαδικασίας. Ο συνδυασμός αυτών των υπηρεσιών δημιουργεί μια συνθέτη υπηρεσία, η οποία αντιπροσωπεύει μια συγκεκριμένη επιχειρηματική λειτουργία. Ο τρόπος με τον οποίο μια υπηρεσία θα γίνει διαθέσιμη προς κατανάλωση καθορίζεται σε αυτό το στάδιο, επίσης μετά την ανάπτυξη της υπηρεσίας γίνεται προετοιμασία της υπηρεσίας για τυχόν μελλοντικές αλλαγές και περαιτέρω ανάπτυξη καθώς σε ένα επιχειρηματικό περιβάλλον τα δεδομένα και οι απαιτήσεις συνήθως αλλάζουν κατά καιρούς. Ο κατάλληλος χειρισμός των μεταδεδομένων (metadata) και η τήρηση ιστορικού εκδόσεων (versioning), επιτρέπουν στους οργανισμούς να

ενισχύσουν τις υπηρεσίες τους, καθώς και να αναπτύξουν πολλές εκδόσεις μιας υπηρεσίας και όλα αυτά με οικονομικά αποδοτικό και παραγωγικό τρόπο.

- **Στάδιο Ολοκλήρωσης (Integrate)**. Αφού η υπηρεσία έχει σχεδιαστεί και η διεπαφή έχει προγραμματιστεί, η υπηρεσία περνάει στο επόμενο στάδιο, που είναι το στάδιο της ενσωμάτωσης. Σε αυτό το στάδιο η υπηρεσία ενσωματώνεται με άλλα στοιχεία του οργανισμού όπως είναι άλλες υπηρεσίες, οι βάσεις δεδομένων, τα συστήματα διαχείρισης των συναλλαγών και άλλες εφαρμογές. Σε τέτοιου είδους ενσωματώσεις, τα δεδομένα συνήθως χρίζουν αλλαγών, για να είναι αξιοποιήσιμα, προσβάσιμα και κατανοητά, αυτό συνήθως επιτυγχάνεται: (μέσο μετασχηματισμών των δεδομένων έτσι ώστε τα δεδομένα να είναι σωστά χαρτογραφημένα μεταξύ διαφορετικών σχημάτων βάσεων, (ii) καθώς και μέσω της δυναμικής δρομολόγησης κατά την ώρα εκτέλεσης για την σύνδεση των κατάλληλων υπηρεσιών.
- **Στάδιο ενορχήστρωσης και χορογραφίας (Orchestrate)**: Μόλις έχει σχεδιαστεί και ενσωματωθεί ένας ικανοποιητικός αριθμός υπηρεσιών, τότε αυτές ή κάποιες από αυτές μπορούν να συνδυαστούν με ενορχηστρωμένα βήματα η με την μορφή χορογραφίας για την δημιουργία συνεχών και αξιόπιστων ροών διαδικασιών. Οι συνδυασμοί αυτοί οι υπηρεσιών αναλύονται σε επόμενη ενότητα.
- **Στάδιο διασφάλισης (Secure)**. Σε αυτό το στάδιο πραγματοποιείται η ενσωμάτωση των παραμέτρων ασφάλειας στις υπηρεσίες. Είναι σημαντικό πριν την κατανάλωση των υπηρεσιών από τους εν δυνάμει πελάτες, αυτές να έχουν διασφαλιστεί. Επίσης κρίσιμης σημασίας για μια υπηρεσία είναι η διασφάλιση της ταυτοποίησης και τις αδειοδότησης των χρηστών, καθώς και η παροχή και διαχείριση της ταυτότητας τους.
- **Στάδιο διαχείρισης (Manage)**. Στο στάδιο της διαχείρισης γίνεται ο καθορισμός, η επιβολή των συμφωνιών στο επίπεδο των υπηρεσιών, καθώς και των λειτουργικών πολιτικών, για τον έλεγχο και χρέωση της χρήσης της υπηρεσίας. Η καλή διαχείριση της SOA μπορεί να εγγυηθεί σε έναν οργανισμό ότι οι υπηρεσίες του θα είναι πιθανότατα αξιόπιστες, διαθέσιμες και ότι θα παρακολουθούνται συνεχώς για λάθη ή αποτυχίες.

- **Στάδιο πρόσβασης (Access)**. Οι υπηρεσίες μπορούν να προσπελαστούν με διαφορετικούς τρόπους, ακόμα και μέσω ασύρματων συσκευών όπως είναι τα κινητά τηλέφωνα και οι συσκευές χειρός και συνήθως εκτίθενται στους χρήστες μέσω μιας πύλης ή μιας σύνθετης Web εφαρμογής. Ένα περιβάλλον SOA υποστηρίζει πολλαπλά κανάλια πρόσβασης στις υπηρεσίες δίνοντας έτσι τη δυνατότητα στους οργανισμούς να προσαρμόζουν τα περιβάλλοντα εργασίας του χρήστη χωρίς να τροποποιήσουν τις υποκείμενες υπηρεσίες, παρέχοντας έτσι αυξημένη ευελιξία στην εμπειρία του χρήστη κατά την πρόσβαση σε αυτές τις υπηρεσίες.
- **Στάδιο ανάλυσης (Analyze)**. Για να μπορέσουν οι διαχειριστές και οι εργαζόμενοι ενός οργανισμού να ελέγχουν αποτελεσματικά, να αναλύουν και να ανταποκρίνονται σε κρίσιμης σημασίας από άποψη χρόνου, ζητήματα, χρειάζεται η ανάλυση των υπηρεσιών, των γεγονότων, καθώς και των επιχειρηματικών διαδικασιών οι οποίες εμπλέκονται στις επιχειρηματικές λειτουργίες. Αυτό επιτρέπει στις επιχειρήσεις να εντοπίσουν τις οποιεσδήποτε δυσκολίες ή σημεία συμφόρησης παρουσιαστούν στις διαδικασίες τους και να ενημερώνουν το κατάλληλο προσωπικό όταν ένα συγκεκριμένο συμβάν είναι άξιο προσοχής.

Ακολουθώντας ένα κύκλο ζωής, όπως αυτόν που παρουσιάζεται σε αυτή τη προσέγγιση, εξασφαλίζει στον οργανισμό ότι οι υπηρεσίες του κατά την ανάπτυξη τους θα έχουν την κατάλληλη ασφάλεια, αξιοπιστία και διαθεσιμότητα.

2.4 Web Services

2.4.1 Ορισμός

“Ένα Web Service είναι ένα σύστημα λογισμικού σχεδιασμένο να υποστηρίζει την διαλειτουργικότητα μηχανής-προς-μηχανή μέσω του δικτύου. Διαθέτει μία διεπαφή που περιγράφεται σε τέτοια μορφή που την καθιστά ικανή προς επεξεργασία από μηχανή (συγκεκριμένα με τη Web Service Definition Language - WSDL). Τα άλλα συστήματα

αλληλεπιδρούν με το Web Service με το τρόπο που προδιαγράφεται στην περιγραφή της WSDL χρησιμοποιώντας μηνύματα SOAP, που συνήθως μεταφέρονται πάνω από HTTP, με την χρήση της XML για την μετατροπή των δεδομένων(XML Serialization) και σε συνεργασία με άλλα πρότυπα σχετικά με το Web."

W3C, 2004

Συνεπώς, όταν είναι επιθυμητό μια λειτουργικότητα να είναι προσβάσιμη από όλους τους κόμβους ενός δικτύου, ανεξαρτήτως του λειτουργικού το οποίο διαθέτουν και την γλώσσα προγραμματισμού που χρησιμοποιούν, μπορούμε να το επιτύχουμε με την χρήση μιας υπηρεσίας (service). Η λειτουργικότητα αυτή θα γίνεται διαθέσιμη μέσω ανταλλαγής μηνυμάτων τα οποία θα χρησιμοποιούν τα βασικά πρωτόκολλα επικοινωνίας και τις βασικές μορφές μηνυμάτων (message formats), εκπληρώνοντας της απαιτήσεις δια-λειτουργικότητας μεταξύ των κατανεμημένων υπηρεσιών και εφόσον η αρχιτεκτονική εστιάζει στο μήνυμα το οποίο ανταλλάσσεται επιτυγχάνεται μια χαλαρή σύνδεση των εφαρμογών (loosely coupled architecture), σχήμα 5

Η Microsoft μέσα από το MSDN της καταλήγει ότι όλα τα web services έχουν τρία (3) κοινά χαρακτηριστικά:

- Τα web services εκθέτουν χρήσιμη λειτουργικότητα σε χρήστες του διαδικτύου μέσα από ένα πρότυπο δικτυακό πρωτόκολο. Στις περισσότερες περιπτώσεις αυτό το πρωτόκολλο είναι το SOAP (Simple Object Access Protocol).
- Τα web services παρέχουν ένα τρόπο να περιγράψουν τις διεπαφές τους με αρκετή λεπτομέρεια ώστε να επιτρέψουν στο χρήστη τους να χτίσει μια εφαρμογή πελάτη η οποία να επικοινωνήσει μαζί τους. Η περιγραφή συνήθως παρέχεται σε ένα έγγραφο XML το οποίο ονομάζεται έγγραφο WSDL (Web Services Description Language).
- Τα web services καταχωρούνται ώστε οι δυνητικοί χρήστες να μπορούν να τα βρουν εύκολα. Αυτό γίνεται με το UDDI (Universal Discovery Description and Integration).

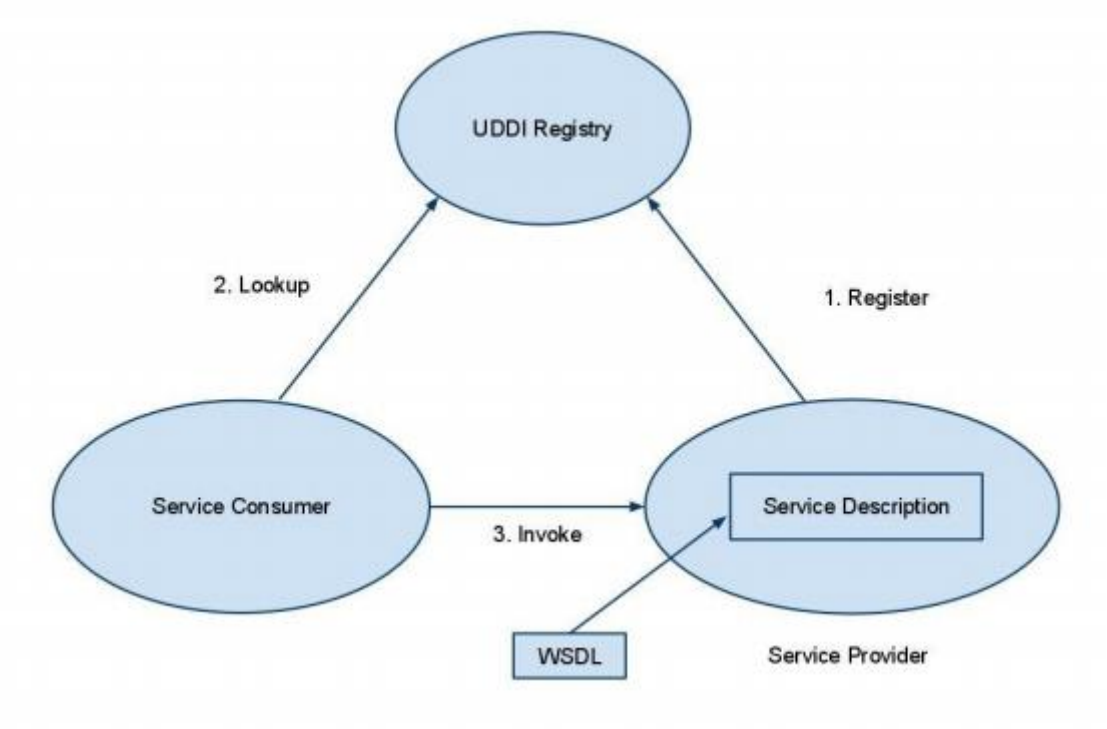
Τα web services λοιπόν αποτελούν μία αρχιτεκτονική κατανεμημένων συστημάτων κατασκευασμένη από πολλά διαφορετικά υπολογιστικά συστήματα τα οποία επικοινωνούν μέσω του δικτύου ώστε να δημιουργήσουν ένα σύστημα. Αποτελούνται από ένα σύνολο από πρότυπα τα οποία επιστρέπουν στους υπεύθυνους για την ανάπτυξη (προγραμματιστές - developers) να υλοποιήσουν κατανεμημένες εφαρμογές (χρησιμοποιώντας διαφορετικά εργαλεία από διαφορετικούς προμηθευτές) ώστε να κατασκευάσουν εφαρμογές που χρησιμοποιούν ένα συνδυασμό από ενότητες λογισμικού (software modules) οι οποίες καλούνται από συστήματα που ανήκουν σε διαφορετικά τμήματα ενός οργανισμού ή σε διαφορετικούς οργανισμούς.

2.4.2 Ρόλοι

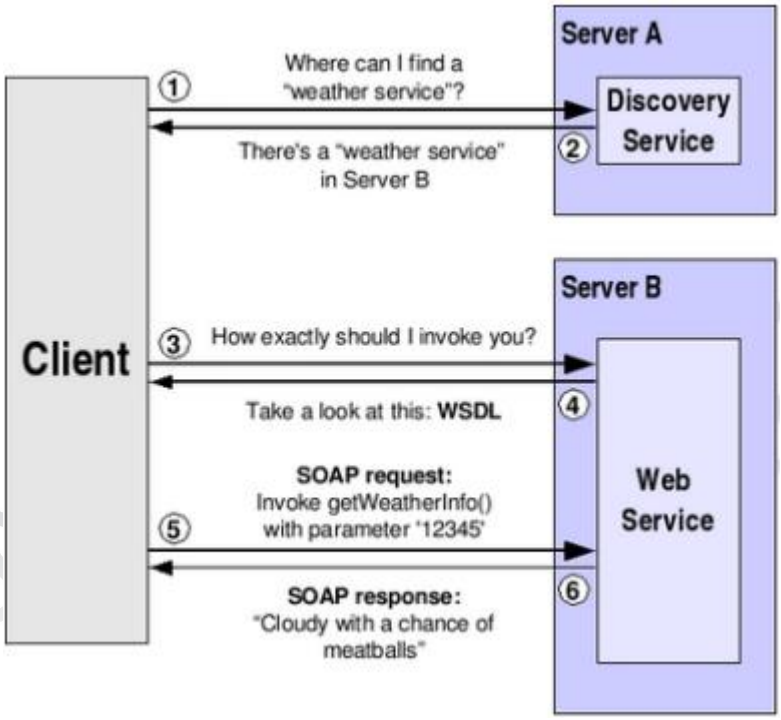
Στην αρχιτεκτονική των WS όπως και στην SOA υπάρχουν τρεις κύριοι ρόλοι και είναι παρόμοιοι:

1. **Πάροχος Υπηρεσίας** (Service Provider): Αυτός είναι ο πάροχος του WS. Συνήθως είναι αυτός που το έχει δημιουργήσει και να το έχει δημοσιοποιήσει (publish) σε ένα μητρώο υπηρεσιών στο Διαδίκτυο.
2. **Αιτών Υπηρεσίας** (Service Consumer): Αυτός είναι ο κάθε καταναλωτής της υπηρεσίας. Μπορεί να έχει βρει (find) το WS προς χρήση μέσω ενός μητρώου υπηρεσιών και να το χρησιμοποιήσει με το ανοίξει μια δικτυακή σύνδεση (bind) και στέλνοντας είναι XML αίτημα.
3. **Μητρώο Υπηρεσιών** (Service Registry): Αυτό αποτελεί είναι ένα κεντρικό κατάλογο των υπηρεσιών. Το μητρώο παρέχει ένα κεντρικό μέρος όπου οι προγραμματιστές μπορούν να δημοσιεύουν τις νέες υπηρεσίες ή να βρουν τις ήδη υπάρχουσες. Παράδειγμα ενός τέτοιου μητρώου αποτελεί το UDDI που αναλύεται σε επόμενη ενότητα.

Το σχήμα 4 αποτυπώνει τους κύριους ρόλους ενός WS και πως αλληλεπιδρούν μεταξύ τους και το σχήμα 10 δείχνει μια τυπική κλήση ενός WS



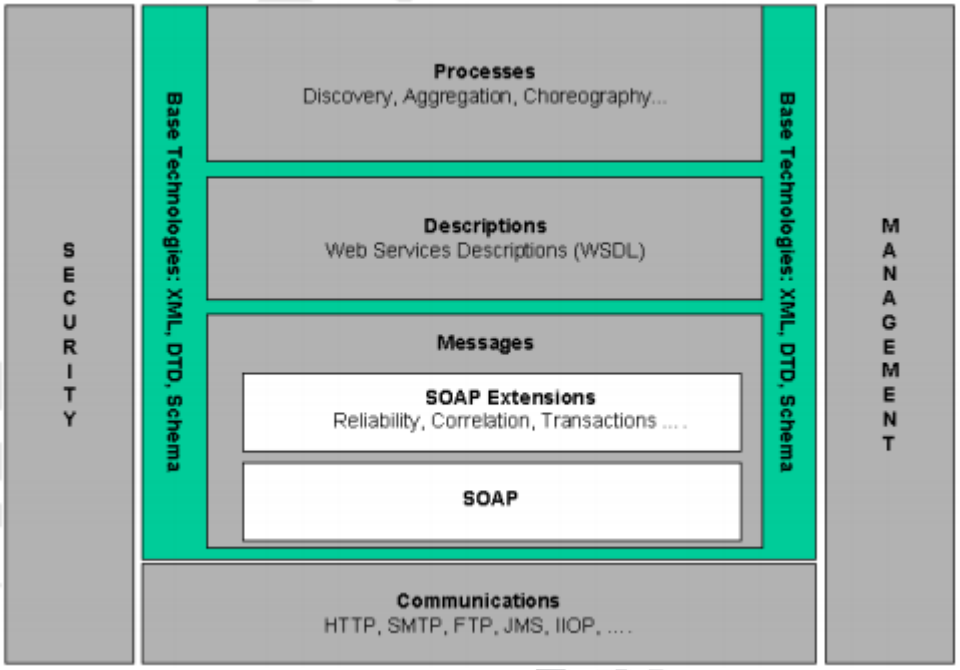
Σχήμα 4 Οι ρόλοι ενός Web Service



Σχήμα 5 Ένα τυπικό παράδειγμα κλήσης ενός Web Service

2.4.3 Αρχιτεκτονική Στοίβα

Τα βασικά στοιχεία της αρχιτεκτονικής των web services (Sotomayor, 2005):



Εικόνα 7 Η αρχιτεκτονική Στοίβα των Web Service

- **Διαδικασίες (Processes):** Αυτό το μέρος της αρχιτεκτονικής περιλαμβάνει συνήθως περισσότερα από ένα Web service. Για παράδειγμα, η ανακάλυψη ανήκει σε αυτό το μέρος της αρχιτεκτονικής, δεδομένου ότι επιτρέπει τον εντοπισμό μιας συγκεκριμένης υπηρεσίας, μεταξύ μια συλλογής από Web services. Κάποιοι παραλείπουν τις διαδικασίες στο βασικό κορμό της αρχιτεκτονικής στοίβας και παρουσιάζουν τις υπόλοιπες τρεις στην επονομαζόμενη wire stack.
- **Περιγραφές (Descriptions):** Ένα από τα πιο ενδιαφέροντα χαρακτηριστικά των Υπηρεσιών του Παγκοσμίου Ιστού είναι ότι είναι αυτοπεριγραφικά. Αυτό σημαίνει ότι, μετά την εύρεση ενός Web Service, δίνεται η δυνατότητα να του γίνει ερώτηση με σκοπό να περιγράψει το ίδιο το Web service ποιες λειτουργίες υποστηρίζει και το πώς μπορούν να κληθούν. Αυτό μπορεί να επιτευχθεί μέσω της Web Services Περιγραφή Language (WSDL).
- **Μηνύματα (Messages):** Η κλήση σε ένα Web service περιλαμβάνει ανταλλαγή μηνυμάτων μεταξύ του πελάτη και του εξυπηρετητή. Το SOAP καθορίζει το πώς πρέπει να μορφοποιηθούν τα αιτήματα προς στον

εξυπηρετητή και το πώς ο εξυπηρετητής πρέπει να μορφοποιήσει τις απαντήσεις του προς τον πελάτη.

- **Επικοινωνίες (Communications):** Τέλος, όλα αυτά τα μηνύματα πρέπει να μεταδίδονται με κάποιο τρόπο μεταξύ του εξυπηρετητή και του πελάτη. Το σύνηθες πρωτόκολλο που επιλέγεται για αυτό το τμήμα της αρχιτεκτονικής είναι το HTTP, το ίδιο πρωτόκολλο που χρησιμοποιείται για πρόσβαση στις συμβατικές ιστοσελίδες στο Διαδίκτυο. Θεωρητικά θα μπορεί κάποιος να είναι σε θέση να χρησιμοποιήσει και άλλα πρωτόκολλα, αλλά όπως προαναφέρθηκε το HTTP σήμερα είναι το πιο δημοφιλές.

2.4.4 Τεχνολογίες

Σε αυτή την ενότητα εξετάζονται οι κρίσιμης σημασίας τεχνολογίες που συνθέτουν ένα Web service και αυτές είναι η XML, το SOAP και η WSDL, υπάρχουν αρκετές ακόμα τεχνολογίες, άλλα οι τρεις αυτές είναι αυτές που δίνουν “ζωή” σε ένα Web service.

2.4.4.1 Πρωτόκολλα Επικοινωνίας

Ως Πρωτόκολλο επικοινωνίας ορίζεται ένα σύνολο κανόνων συμφωνημένων και από τα δυο επικοινωνούντα μέρη και που εξυπηρετούν την μεταξύ τους ανταλλαγή πληροφοριών. Το πρωτόκολλο επικοινωνίας είναι δηλαδή μια δέσμη κανόνων στους οποίους στηρίζεται η επικοινωνία των συσκευών (συνήθως, αλλά όχι πάντα, υπολογιστών) σε ένα δίκτυο. Οι κανόνες αυτοί καθορίζουν τη μορφή, το χρόνο και τη σειρά μετάδοσης των πληροφοριών στο δίκτυο.

Τα πιο διαδεδομένα πρωτόκολλα επικοινωνίας είναι τα:

- HTTP (HyperText Transfer Protocol)
- TCP (Transmission Control Protocol)
- JMS (Java Message Service)
- SMTP (Simple Mail Transfer Protocol)
- MQ (Message Queues)

- BEEP (Blocks Extensible Exchange Protocol) is a framework for creating network application protocols
- CORBA (Common Object Request Broker Architecture)

2.4.4.2 ExtensibleMarkupLanguage

Σκοπός της XML είναι ο διαμοιρασμός δομημένων δεδομένων (structured data) μεταξύ πληροφοριακών συστημάτων (information systems), κυρίως μέσω του διαδικτύου. Η XML ορίζεται ως:

> *"Η γλώσσα σήμανσης XML περιγράφει μια κατηγορία πληροφοριών (data objects) που καλούνται XML έγγραφα καθώς επίσης περιγράφει μερικώς τη συμπεριφορά των προγραμμάτων που τα επεξεργάζονται. Η XML ουσιαστικά προέρχεται από την SGML και αποτελεί μια προσαρμοσμένη έκδοσή της. Τα XML έγγραφα αποτελούνται από μονάδες αποθήκευσης που καλούνται οντότητες, οι οποίες περιέχουν αναλυμένες ή μη πληροφορίες. Οι αναλυμένες πληροφορίες αποτελούνται από χαρακτήρες οι οποίοι συνθέτουν τα δεδομένα χαρακτήρων και άλλοι οι οποίοι συνθέτουν την σήμανση. Η σήμανση κωδικοποιεί την περιγραφή της τελικής μορφής του εγγράφου προς αποθήκευση καθώς και τη λογική δομή του. Επίσης η XML παρέχει ένα μηχανισμό για την επιβολή περιορισμών όσον αφορά τη μορφή αποθήκευσης και λογική δομή."*
>
> *W3C, 2004*

Τα έγγραφα που είναι γραμμένα σε XML αποτελούνται από περιεχόμενο και σήμανση όπως αναφέρεται στον ορισμό. Η σήμανση μπορεί να είναι:

- Στοιχεία
- Αναφορές οντοτήτων
- Σχόλια, τα οποία αρχίζουν με <!-- και τελειώνουν με -->
- Εντολές επεξεργασίας οι οποίες αρχίζουν με <? και τελειώνουν με ?>
- Τμήματα CDATA (CDATA sections)

- Ορισμός τύπου εγγράφου (document type declaration)

Επιπλέον, η XML έχει ενσωματωμένο ένα μηχανισμό επικύρωσης δεδομένων, ο οποίος εγγυάται ότι η δομή των δεδομένων σε ένα έγγραφο που λαμβάνεται είναι έγκυρη. Ένα έγγραφο XML είναι έγκυρο όταν το περιεχόμενο του είναι σύμφωνο με έναν ορισμό τύπου εγγράφου ή ένα XML σχήμα (schema) το οποίο ορίζει τα επιτρεπόμενα στοιχεία, τις ιδιότητες και τα περιεχόμενά τους.

Η XML αποτελεί βασική τεχνολογία των Web service. Τα μηνύματα (SOAP) οι περιγραφές (WSDL) και οι διαδικασίες (Ανακάλυψης, συγκέντρωσης, χορογραφίας, ενορχήστρωσης, κλπ) βασίζονται σε αυτή.

2.4.5 SOAP

Το ακρώνυμο της SOAP μέχρι την έκδοση 1.1 μεταφραζόταν σε Simple Object Access Protocol αλλά η τεχνική επιτροπή αποφάσισε πλέον να μην αποτελεί ακρώνυμο (W3C, 2007) γιατί δεν ήταν πλέον απλή (Simple) ούτε έχει να κάνει με αντικείμενα (Objects).

> *“Το SOAP στην έκδοση 1.2 είναι ένα ελαφρύ πρωτόκολλο προορισμένο για την ανταλλαγή δομημένων πληροφοριών σε ένα αποκεντρωμένο, κατανεμημένο περιβάλλον. Χρησιμοποιεί τεχνολογίες XML για να καθορίσει ένα επεκτάσιμο πλαίσιο παρέχοντας μια δομή μηνυμάτων η οποία μπορεί να ανταλλαχθεί πάνω από ποικίλα δικτυακά πρωτόκολλα. Το πλαίσιο έχει σχεδιαστεί να είναι ανεξάρτητο από οποιοδήποτε προγραμματιστικό μοντέλο και άλλες ειδικές σημασιολογίες υλοποίησης..”*
>
> *W3C, 2007*

Τα πρωτόκολλα επιπέδου εφαρμογής (application layer) στα οποία στηρίζεται η παρούσα προδιαγραφή για την διαπραγμάτευση και μετάδοση μηνυμάτων είναι κυρίως τα Hypertext Transfer Protocol (HTTP) ή το απλό πρωτόκολλο μεταφοράς ταχυδρομείου (SMTP) και σε αυτά θα εστιάσουμε στην συνέχεια.

Τα μηνύματα SOAP αποτελούνται από τρία μέρη:

- **Έναν φάκελο (envelope),** ο οποίος προσδιορίζει την δομή του μηνύματος και τον τρόπο επεξεργασίας του
- **Ένα σύνολο κανόνων κωδικοποίησης** για την έκφραση παραδειγμάτων των τύπων δεδομένων που θα χρησιμοποιεί η εφαρμογή.
- **Μια σύμβαση** για την αναπαράσταση διαδικαστικών κλήσεων και των απαντήσεων

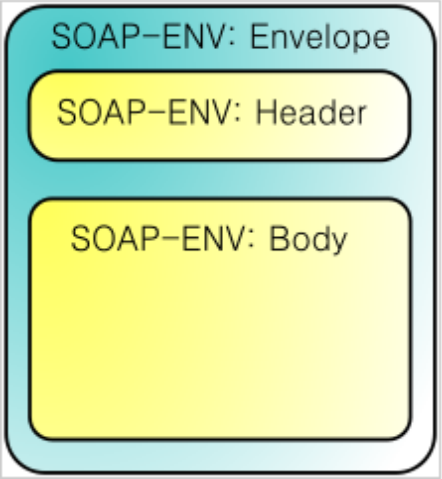
Τα τρία βασικά χαρακτηριστικά του SOAP είναι:

- **επεκτασιμότητα (extensibility):** ασφάλεια και WS - δρομολόγησης είναι μεταξύ των επεκτάσεων υπό ανάπτυξη
- **ουδετερότητα (neutrality)**: SOAP μπορεί να λειτουργήσει πάνω από οποιοδήποτε πρωτόκολλο μεταφοράς, όπως HTTP , SMTP , το πρωτόκολλο TCP , UDP , ή JMS
- **ανεξαρτησίας (independence)**: SOAP λειτουργεί σε κάθε μοντέλο προγραμματισμού

Τα μηνύματα SOAP αναπαριστώνται ως φάκελοι (envelopes), στους οποίους η εφαρμογή εσωκλείει τα δεδομένα προς αποστολή. Η δομή ενός μηνύματος SOAP μπορεί να φανεί στο σχήμα 13. Ένα μήνυμα έχει δύο βασικά μέρη:

Το header: το οποίο μπορεί να διαιρεθεί σε blocks, χρησιμοποιείται για δεδομένα επιπέδου infrastructure, και είναι προαιρετικό.

Το body: το οποίο μπορεί επίσης να διαιρεθεί σε blocks, χρησιμοποιείται για δεδομένα επιπέδου εφαρμογής, είναι υποχρεωτικό και σε αυτό γίνεται η διαχείριση τυχόν σφαλμάτων που θα προκύψουν.



Εικόνα 8 Δομή μηνύματος SOAP

Το SOAP υποστηρίζει δύο βασικές λειτουργίες αλληλεπίδρασης (Rosenberg & Remy, 2004):

- **RPC/Κωδικοποιημένη**: Στην λειτουργία αυτή η ανταλλαγή μηνυμάτων έχει πολλά κοινά με την ανταλλαγή μηνυμάτων μέσω του ενδιάμεσου λογισμικού (middleware) RPC, δηλαδή τα μηνύματα είτε αποτελούν αιτήσεις για την εκτέλεση μιας λειτουργίας και περιλαμβάνουν το όνομα της λειτουργίας και τυχόν παραμέτρους, είτε πιθανόν απαντήσεις που μεταφέρουν το αποτέλεσμα/τα των αιτήσεων. Οι επικοινωνίες μέσω του RPC τείνουν να είναι σύγχρονες και πιο “ομαλές” αυτό καθιστά αυτή την λειτουργία κατάλληλη για ενδοεπιχειρησιακή χρήση όπου η ταχύτητα είναι υψηλή και οι καθυστερήσεις μικρές.
- **Έγγραφα/Λεκτική**: Αυτή η λειτουργία υποστηρίζει μια πιο χαλαρής συνδεσιμότητα επικοινωνία. Η επικοινωνίες μέσω εγγράφων τείνουν να είναι ασύγχρονες και “μη-ομαλές”, καθιστώντας αυτή την λειτουργία ανταλλαγής μηνυμάτων κατάλληλη για επικοινωνία μεταξύ επιχειρήσεων. Η μορφή των εγγράφων που ανταλλάσσονται είναι προσυμφωνημένη από τους συμμετέχοντες (αποστολέα, παραλήπτη).

Υπάρχουν πολλοί και διαφορετικοί πρότυπα ανταλλαγής μηνυμάτων (MEP) μεταξύ δύο κόμβων SOAP. Αλλά στην προδιαγραφή του SOAP 1.2 περιγράφονται δύο:

- **Το Request-Response MEP**: καθορίζει ένα σχέδιο, όταν ένα μήνυμα SOAP αποστέλλεται ως αίτημα και ακολουθείται από ένα μήνυμα SOAP που αποστέλλεται ως απάντηση. Το αποτέλεσμα μιας ανταλλαγής που πραγματοποιήθηκε με επιτυχία είναι ακριβώς δύο μηνύματα SOAP. **Το Response MEP**: καθορίζει ένα σχέδιο, όταν ένα μήνυμα SOAP αποστέλλεται ως απάντηση σε μια μη SOAP αίτηση. Το αίτημα δεν περιέχεται σε φάκελο SOAP και δεν απαιτεί επεξεργασία, η αίτηση διαβιβάζεται με ένα ειδικό τρόπο binding. Το μήνυμα απάντησης περιέχει φάκελο SOAP. Αυτό το πρότυπο ανταλλαγής δεν μπορεί να χρησιμοποιηθεί με τα χαρακτηριστικά που εκφράζονται σε SOAP header μπλοκ γιατί το αίτημα δεν έχει SOAP φάκελο για να τα εισάγει.

```
POST /InStock HTTP/1.1

Host: www.example.org

Content-Type: application/soap+xml; charset=utf-8

Content-Length: nnn


<?xml version="1.0"?>

<soap:Envelope

xmlns:soap="http://www.w3.org/2001/12/soap-envelope"

soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">


<soap:Bodyxmlns:m="http://www.example.org/stock">
```

```
 <m:GetStockPrice>
        <m:StockName>IBM</m:StockName>
 </m:GetStockPrice>
</soap:Body>


</soap:Envelope>
```

Σχήμα 6 Παράδειγμα ενός SOAP request

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn


<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">


<soap:Bodyxmlns:m="http://www.example.org/stock">
 <m:GetStockPriceResponse>
        <m:Price>34.5</m:Price>
 </m:GetStockPriceResponse>
</soap:Body>
```

```
</soap:Envelope>
```

Σχήμα 7 Παράδειγμα ενός SOAP response

Και στις δύο MEPs τα σφάλματα που προκύπτουν στην μετάδοση του μηνύματος είτε στον αποστολέα είτε στον παραλήπτη μπορεί να αποσιωπηθούν, ή μπορεί να οδηγήσουν στην παραγωγή σφάλματος SOAP ή σφάλματος σχετιζόμενο με το binding.

2.4.6 Web Service Description Language (WSDL)

Η WSDL 1.0 όπως και το UDDI ήταν μια πρωτοβουλία των εταιριών IBM, Microsoft και Ariba για την περιγραφή των Web services για τις δικές τους SOAP εργαλειοθήκες. Η WSDL προέκυψε από τον συνδυασμό δύο γλωσσών της NASSL (Network Application Service Specification Language) που προερχόταν από την IBM και της SDL (Service Description Language) που προερχόταν από την Microsoft και δημοσιεύτηκε το Σεπτέμβριο του 2000. Η WSDL 1.1 δημοσιεύτηκε τον Μάρτιο του 2001, αποτελούσε την επισημοποίηση της WSDL 1.0 και δεν είχαν εισαχθεί σημαντικές αλλαγές από την έκδοση 1.0. Η επόμενη έκδοση WSDL 1.2 δημοσιεύθηκε τον Ιούνιο του 2003 και εξακολουθεί να είναι ένα λειτουργικό σχέδιο (draft) στον W3C. Σύμφωνα με το W3C: Η WSDL 1.2 είναι πιο εύκολη και πιο ευέλικτο για τους προγραμματιστές από την προηγούμενη έκδοση. Η WSDL 1.2 επιχειρεί την αφαίρεση των μη διαλειτουργικών χαρακτηριστικών και επίσης, σε αυτή ορίζεται το καλύτερο HTTP 1.1 binding. Η WSDL 1.2 όμως δεν υποστηρίχθηκε από την πλειοψηφία των διακομιστών/κατασκευαστών SOAP. Η WSDL 2.0 είναι η τελευταία έκδοση, βρίσκεται στην κατάσταση “σύσταση” από το W3C από τον Ιούνιο του 2007, και προέρχεται από την μετονομασία του WSDL 1.2. Η πιο διαδεδομένη έκδοση μεταξύ της 1.1 και της 2.0 όπως έμμεσα προαναφέρθηκε είναι η έκδοση WSDL 1.1 .

"Η WSDL είναι ένα σχήμα XML για την περιγραφή δικτυακών υπηρεσιών σαν ένα σύνολο από τελικά σημεία που λειτουργούν σε μηνύματα τα οποία περιέχουν πληροφορία είτε προσανατολισμένη στα έγγραφα είτε προσανατολισμένη στις διαδικασίες. Οι λειτουργίες και τα μηνύματα περιγράφονται περιληπτικά και τότε δίνονται σε ένα συγκεκριμένο πρωτόκολλο δικτύων με τη μορφή μηνυμάτων για να καθορίσουν ένα τελικό σημείο. Πολλά σχετικά τελικά σημεία συνδυάζονται σε υπηρεσίες (services)."

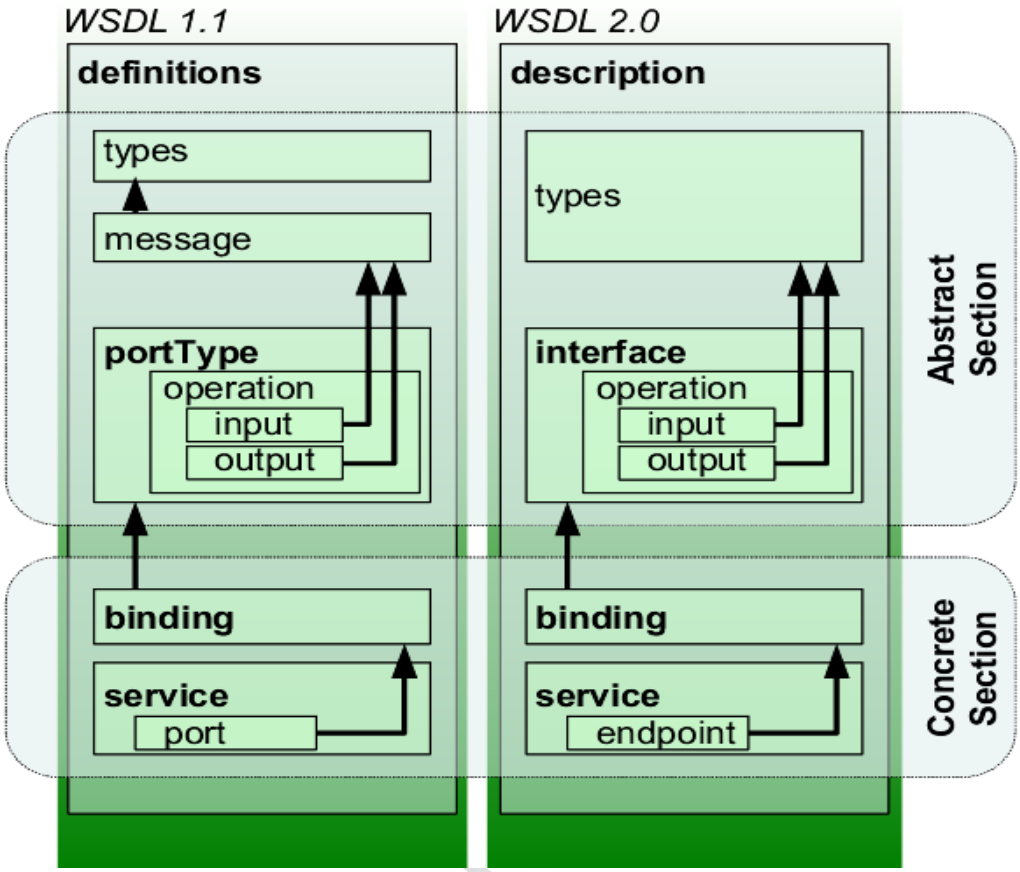
W3C, 2001

Ένα έγγραφο WSDL 1.1 χρησιμοποιεί τα παρακάτω αντικείμενα για τον ορισμό των web services :

- **Types** - ένα περίβλημα για τους ορισμούς των τύπων δεδομένων χρησιμοποιώντας ένα σύστημα τύπων (όπως για παράδειγμα το XML Schema).
- **Message** - ένας περιγραφικός ορισμός των δεδομένων που ανταλλάσσονται.
- **Operation** -περιγραφή μίας λειτουργίας που υποστηρίζεται από μία υπηρεσία
- **PortType** - ένα περιγραφικό σύνολο από λειτουργίες που υποστηρίζονται από ένα ή περισσότερα τελικά σημεία.
- **Binding** - ένα συγκεκριμένο πρωτόκολλο και μορφή δεδομένων για ένα συγκεκριμένο τύπο τελικών σημείων (port type).
- **Port** - ένα μοναδικό τελικό σημείο που ορίζεται σαν συνδυασμός μίας σύνδεσης (binding) και μιας διεύθυνσης δικτύου.
- **Service** - μία συλλογή από σχετικά τελικά σημεία.

Οι διαφορές στα αντικείμενα για τον ορισμό των web services μεταξύ WSDL 1.1 και WSDL 2.0, που φαίνονται και στο σχήμα 13 είναι οι ακόλουθες:

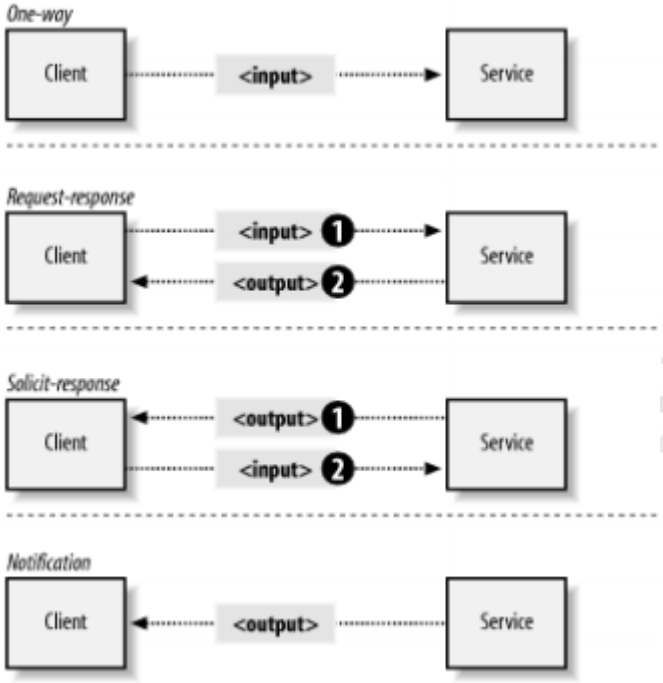
- Το PortType μετονομάστηκε σε Interface
- Το Port μετονομάστηκε σε Endpoint
- Το Message δεν χρησιμοποιείται στην προδιαγραφή WSDL 2.0 και αφαιρέθηκε, το XML σχήμα πλέον είναι αυτό που αποδίδει απλά και άμεσα για ορισμένες εισόδους τα απαραίτητα σφάλματα ή εξόδους.



Σχήμα 8 Αναπαράσταση των εννοιών που καθορίζονται στα έγγραφα WSDL 1.1 και WSDL 2.0

Στο WSDL 1.1 υποστηρίζονται τρία πρότυπα ανταλλαγής μηνυμάτων:

- **One-way**: Το endpoint λαμβάνει το μήνυμα.
- **Request-response**: Το endpoint λαμβάνει ένα μήνυμα αίτησης και στέλνει ένα μήνυμα απάντησης.
- **Solicit response**: Το endpoint στέλνει ένα μήνυμα και λαμβάνει έπειτα μια απάντηση.
- **Notification**: Το endpoint στέλνει ένα μήνυμα.



Σχήμα 9 Πρότυπα ανταλλαγής μηνυμάτων WSDL 1.1

Στο τελευταίο έγγραφο του WSDL 2.0 υποστηρίζονται τρία πρότυπα ανταλλαγής μηνυμάτων:

- **In-Only**: Αυτό το πρότυπο αποτελείται από ακριβώς ένα μήνυμα που λήφθηκε από την υπηρεσία κάποιου άλλου κόμβου. Μήνυμα σφάλματος δεν μπορεί να παραχθεί σε αυτό το πρότυπο.
- **Robust In-Only**: Αυτό το πρότυπο μπορεί να εκφραστεί ότι αποτελεί παραλλαγή του In-Only. Αποτελείται επίσης από ακριβώς ένα μήνυμα που λήφθηκε από την υπηρεσία κάποιου άλλου κόμβου, αλλά σε αυτή την περίπτωση μπορεί να ληφθεί ένα μήνυμα λάθους το οποίο ορίζεται στο “Message Triggers Fault propagation rule”.
- **In-Out**: Αυτό το πρότυπο αποτελείται από ακριβώς δύο μηνύματα: ένα που λήφθηκε από την υπηρεσία κάποιου άλλου κόμβου, ακολουθημένο από ένα μήνυμα που αποστέλλεται στον άλλο αυτό κόμβο. Μήνυμα σφάλματος δεν μπορεί να παραχθεί σε αυτό το πρότυπο. Το δεύτερο μήνυμα σε αυτό το πρότυπο μπορεί να αντικατασταθεί από ένα μήνυμα σφάλματος όπως αυτό ορίζεται στο “Fault Replaces Message propagation rule

2.4.7 Universal Description, Discovery and Integration (UDDI)

Το Universal Description, Discovery and Integration (UDDI) αποτελεί μια κεντρική υπηρεσία καταλάγου που δεν εξαρτάται από κάποια συγκεκριμένη πλατφόρμα (platform-independent) και έχει σαν βάση του την Extensible Markup Language (XML) μέσα από το οποίο επιχειρήσεις ανά τον κόσμο μπορούν να καταχωρηθούν στο Διαδίκτυο, και έναν μηχανισμό για την καταγραφή και εντοπισμό εφαρμογές υπηρεσιών (web service applications). Το UDDI είναι μια ανοιχτή πρωτοβουλία της βιομηχανίας της πληροφορικής υπό την αιγίδα του Οργανισμού για την Προώθηση των Δομημένων Συστημάτων Πληροφορικής (Organization for the Advancement of Structured Information Standards, OASIS), με στόχο την διευκόλυνση των επιχειρήσεων όσον αφορά την δημοσίευση καταχωρήσεων υπηρεσιών (service listings) την γρήγορη και δυναμική εύρεση καθώς και συναλλαγή με κάθε άλλη υπηρεσία, και να καθορίσει πώς οι υπηρεσίες ή οι εφαρμογές λογισμικού να αλληλοεπιδρούν μέσω του Διαδικτύου .

Το UDDI δεν υιοθετήθηκε με τον τρόπο με τον οποίο οι σχεδιαστές του ευελπιστούσαν και δεν έχει την επιτυχία που έχουν γνωρίσει το SOAP και η WSDL και για ορισμένους δεν συγκαταλέγεται στις βασικές προδιαγραφές ανάπτυξης Web Services. Ένας από τους λόγους που δεν έχει γνωρίσει αυτή την επιτυχία ίσως είναι ότι οι επιχειρήσεις φάνηκαν απρόθυμες να το χρησιμοποιήσουν για να εκκινήσουν συνεργασία με επιχειρήσεις τις οποίες δεν είχαν συνεργαστεί στο παρελθόν. Άλλος λόγος ήταν ότι το UDDI είχε κατασκευαστεί πριν το WSDL, για αυτό και αρχικά δεν υποστηριζόταν σωστά. Οι IBM, Microsoft και SAP ανακοίνωσαν ότι σταματούν την υποστήριξη των UDDI κόμβων τους στης 12 Ιανουαρίου του 2006 (Microsoft, n.d). Εξακολουθούν όμως να υπάρχουν δημόσια UDDI κόμβοι, από άλλους οργανισμούς και όπως λέγεται στον παραπάνω ορισμό “Τα web services έχουν νόημα μόνο όταν δυνητικοί χρήστες μπορούν να βρουν πληροφορίες ικανές ώστε να επιτρέψουν την εκτέλεσή τους”.

Ο παρακάτω πίνακας περιγράφει περιληπτικά τις προσφερόμενες υπηρεσίες του UDDI καθώς και κάποιες από τις βασικές δομές δεδομένων όπως αυτές περιγράφονται στο πρότυπο. Επίσης μπορεί να φανεί και η σχέση που έχουν μεταξύ τους οι βασικές αυτές δομές δεδομένων στο σχήμα 15.

Πληροφορία	**Λειτουργίες**	**Λεπτομέρειες**
White pages: Πληροφορίες όπως το όνομα, η διεύθυνση, το τηλέφωνο και άλλες πληροφορίες επικοινωνίας για μία επιχείρηση.	**Publish:** Πώς ο προμηθευτής ενός web service καταχωρεί των εαυτό του.	**Business Information:** Περιλαμβάνεται σε ένα αντικείμενο **BusinessEntity**, το οποίο με τη σειρά του περιλαμβάνει πληροφορίες για υπηρεσίες, κατηγορίες, επαφές, URLs και άλλα αναγκαία στοιχεία για να αλληλεπιδράσουμε με μία επιχείρηση.
Yellow Pages: Πληροφορίες που κατηγοριοποιούν επιχειρήσεις. Βασίζονται σε υπάρχοντα πρότυπα κατηγοριοποίησης (μη ηλεκτρονικά).	**Find:** Πώς μία εφαρμογή βρίσκει ένα συγκεκριμένο web service.	**Service Information:** Περιγράφει μία ομάδα από web services. Αυτές περιλαμβάνονται σε ένα αντικείμενο **BusinessService**.
Green Pages: Τεχνικές πληροφορίες για τα web services που παρέχονται από μία επιχείρηση.	**Bind:** Πώς μία εφαρμογή συνδέεται και αλληλεπιδρά με ένα web service αφού αυτό βρεθεί.	**Binding Information:** Οι απαραίτητες τεχνικές λεπτομέρειες για την κλήση ενός web service. Περιλαμβάνουν τα URLs, πληροφορίες για ονόματα μεθόδων, τύπους ορισμάτων και κτλ. Το αντικείμενο **BindingTemplate** αναπαριστά αυτά τα δεδομένα. **Service Specification Detail**: Πρόκειται για μεταδεδομένα για διάφορες προδιαγραφές που υλοποιούνται από ένα web service. Αυτά καλούνται **tModels**.

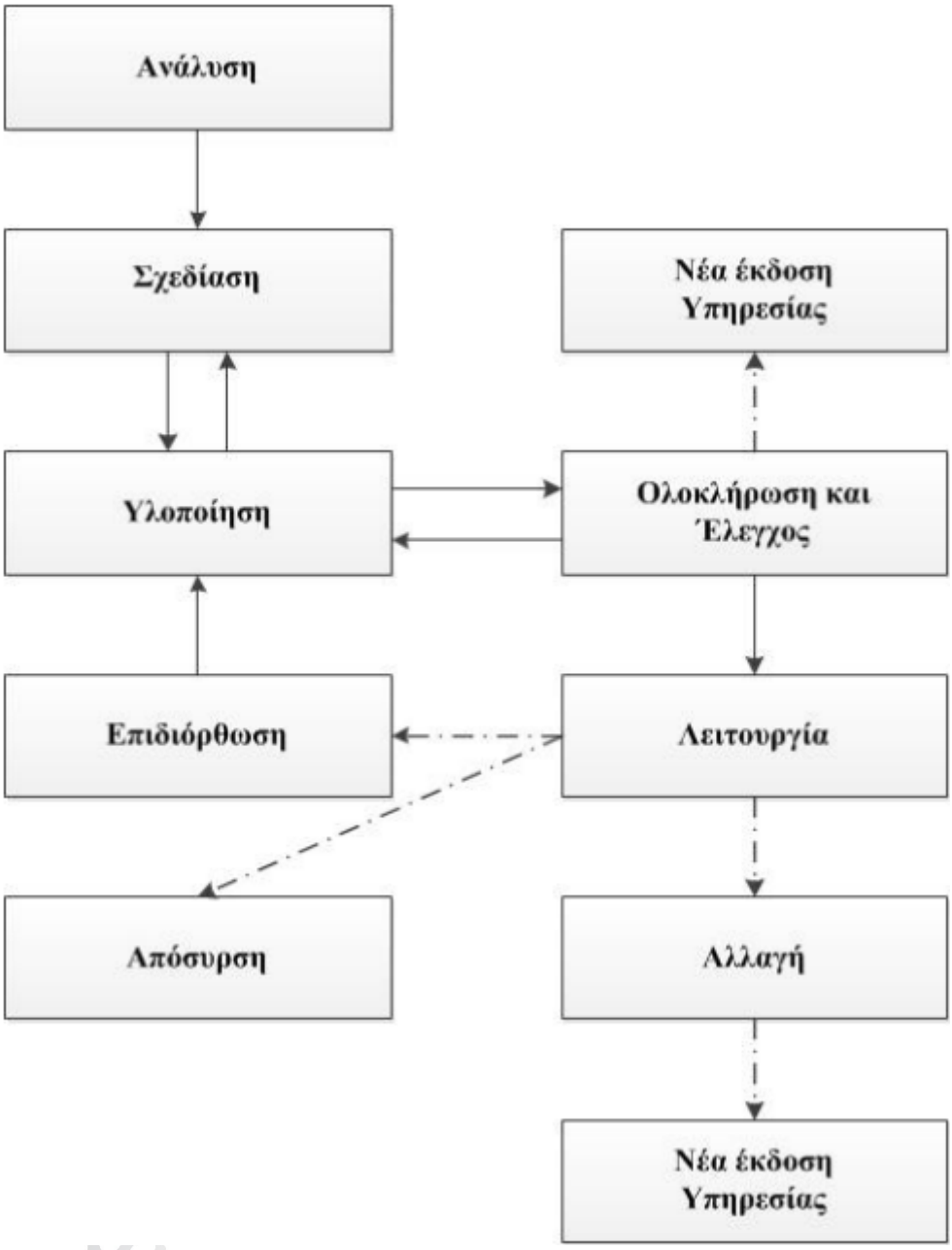
Πίνακας 2: Υπηρεσίες UDDI

2.4.8 Κύκλος Ζωής στην ανάπτυξη ενός Web Service

Παρόλη την προσοχή που έχουν τραβήξει πάνω τους τα web services την τελευταία δεκαετία και τις εκτενής βιβλιογραφικές αναφορές και έρευνες δεν έχει καθοριστεί από κα κάποιος κοινά αποδεκτός κύκλος ζωής κατά την φάση της ανάπτυξης. Ένας ολοκληρωμένος κύκλος ζωής παρουσιάζεται από τους Θεμιστοκλέους και Ματζάνα (2010) και αποτελείται από τα εξής εννιά στάδια:

- Στάδιο 1 – Ανάγκη Δημιουργίας Υπηρεσίας: Σε αυτό το στάδιο συλλαμβάνεται η ιδέα για την δημιουργία ενός Web service, που συνήθως προκύπτει από την ανάγκη του οργανισμού για ένα Web service αυτόνομου ή εντεταγμένου σε κάποιο κεντρικό του πλάνο.

- Στάδιο 2 – Σχεδίαση: Στο στάδιο αυτό γίνεται η σχεδίαση του Web service και πρέπει να ληφθούν υπόψη παράμετροι που επηρεάζουν και αφορούν την διαλειτουργικότητα, την επαναχρησιμοποίηση και την διακυβέρνηση SOA.
- Στάδιο 3 - Υλοποίηση: Το τρίτο στάδιο της υλοποίησης είναι το στάδιο κατά το οποίο αναπτύσσεται ένα Web service. Έμφαση σε αυτό το στάδιο πρέπει να δοθεί, στις παραμέτρους ασφαλείας, στις λειτουργικές και μη λειτουργικές απαιτήσεις, αν θα πρέπει να υπάρξει προτυποποίηση ή χρήση κοινών προτύπων.
- Στάδιο 4 – Ολοκλήρωση και έλεγχος: Σε αυτό το στάδιο πραγματοποιείται ο έλεγχος του Web service. Ελέγχονται η λειτουργικότητα του καθώς και η επικοινωνία του με τους πελάτες του.
- Στάδιο 5 – Έκδοση Υπηρεσίας: Εφόσον η ολοκλήρωση και ο έλεγχος του Web service ολοκληρώθηκαν με επιτυχία το επόμενο στάδιο είναι η έκδοση του, αυτό συμβαίνει μέσω της δημοσιοποίησης του WSDL και του UDDI.
- Στάδιο 6 – Λειτουργία ή Χρήση Υπηρεσίας: Το Web service τίθεται σε λειτουργία σε αυτό το στάδιο και είναι έτοιμο προς να κλήση.
- Στάδιο 7 – Επιδιόρθωση: Τυχόν αδυναμίες και σφάλματα που μπορεί να παρουσιαστούν από το Web service αντιμετωπίζονται σε αυτό το στάδιο. Η επιδιόρθωση οδηγεί σε μια νέα έκδοση του Web service ή οποία τρέχει παράλληλα με την προηγούμενη.
- Στάδιο 8 - Αλλαγή: Κατά την λειτουργία του Web service μπορεί να διαπιστωθεί η ανάγκη για αλλαγή της υπηρεσίας. Όμοια με τον προηγούμενο στάδιο, και σε αυτό στάδιο η τροποποιημένη υπηρεσία οδηγεί σε μια νέα έκδοση του Web service
- Στάδιο 9: Απόσυρση: Στο στάδιο αυτό τυχόν εκδόσεις ή όλες οι εκδόσεις ενός Web service που δεν χρησιμοποιούνται έπειτα από ένα συγκεκριμένο χρονικό διάστημα που ορίζουν κάποιοι κανόνες του οργανισμού, αποσύρονται.



Εικόνα 9: Κύκλος ζωής Web Service

2.4.9 Πλεονεκτήματα

1. Ευκολότερος χειρισμός δεδομένων

Παραδοσιακά το κυριότερο πρόβλημα στις κατανεμημένες τεχνολογίες ήταν το λεγόμενο tight-coupling ή στα ελληνικά η ισχυρή συνδεδιμότητα. Μια εφαρμογή που καλούσε μια άλλη απομακρυσμένη ήταν αυστηρά δεμένη με αυτή από την κλήση λειτουργίας (function call) που εκτελούσε και τις παραμέτρους που περνούσε. Στα περισσότερα συστήματα πριν από την έλευση των web services ο τρόπος επικοινωνίας ήταν μια σταθερή διεπαφή με λίγη έως καθόλου ευελεξία ή προσαρμοστικότητα στα περιβάλλοντα ή τις ανάγκες που μεταβάλλονται συνεχώς.

Τα web services χρησιμοποιούν τη γλώσσα XML η οποία μπορεί να περιγράψει οποιαδήποτε δεδομένα σε ένα πραγματικά ανεξάρτητο από πλατφόρμα τρόπο για ανταλλαγή αυτών των δεδομένων μεταξύ συστημάτων. Με αυτόν τον τρόπο οδηγούμαστε σε εφαρμογές με χαλαρή συνδεσιμότητα (loosely-coupled). Επιπλέον τα web services μπορούν να λειτουργήσουν σε πιο αφηρημένο επίπεδο στο οποίο μπορούν να επαναξιολογήσουν, να τροποποιήσουν ή να χειριστούν τύπους δεδομένων δυναμικά κατά περίπτωση. Έτσι σε τεχνικό επίπεδο τα web services μπορούν να χειριστούν δεδομένα πολύ ευκολότερα και να επιτρέψουν στο λογισμικό να επικοινωνεί πιο ελεύθερα.

2. Απλότητα πρωτοκόλλου επικοινωνίας

Τα web services χρησιμοποιούν ως πρωτόκολλο επικοινωνίας το SOAP. Το πρωτόκολλο αυτό είναι πολύ πιο απλό από πρωτόκολλα παλαιότερων τεχνολογιών όπως αυτά που χρησιμοποούνταν από τα κατανεμημένα περιβάλλοντα CORBA , DCOM, RPC. Έτσι το να δημιουργήσει κανείς μια υλοποίηση SOAP που υπόκειται στα πρότυπα (standards-compliant) είναι πολύ πιο εύκολο. Σήμερα μπορεί να βρει κανείς υλοποιήσεις του SOAP από τις μεγαλύτερες εταιρίες πληροφορικής αλλά ακόμη και από μεμονομένους προγραμματιστές, πράγμα αδιανόητο για παλαιότερες κατανεμημένες τεχνολογίες.

3. Απλότητα υποδομής

Τα web services λειτουργούν με πρότυπες γλώσσες και πρωτόκολλα όπως η XML , το HTTP και το TCP/IP. Η πλειονότητα των εταιριών έχουν ήδη την δικτυακή υποδομή και τους ανθρώπους με γνώσεις και εμπειρία που τη συντηρούν. Έτσι το

κόστος για την εφαρμογή των web services είναι σημαντικά μικρότερο από αυτό των προηγούμενων τεχνολογιών.

4. Ευκολία στην επικοινωνία

Με τις προηγούμενες τεχνολογίες η συνεργασία μεταξύ εταιριών ήταν ένα θέμα διότι κατανεμημένες τεχνολογίες όπως CORBA και DCOM χρησιμοποιούσαν μη πρότυπες πόρτες. Σαν αποτέλεσμα η συνεργασία σήμαινε άνοιγμα "οπών" στα τείχη προστασίας (firewalls) κάτι που πολλές φορές δεν ήταν αποδεκτό από τους ανθρώπους της πληροφορικής σε μια εταιρία αφού έθετε σε κίνδυνο στην ασφάλεια των συστημάτων. Το γεγονός αυτό δεν επέτρεπε δυναμική συνεργασία λόγω του ότι απαιτούσε μια χειροκίνητη διαδικασία για τη συνεργασία μιας εταιρίας με τους συνεργάτες της. Τα web services μπορούν να χρησιμοποιήσουν (μεταξύ άλλων) το HTTP ως πρωτόκολλο μεταφοράς και τα περισσότερα τείχη προστασίας επιτρέπουν την πρόσβαση μέσω της θύρας 80 (πρότυπη θύρα για το HTTP). Με αυτόν τον τρόπο οδηγούμαστε σε ευκολότερες και δυναμικές συνεργασίες μεταξύ των συστημάτων των εταιριών.

5 Διαλειτουργικότητα και ευκολία ανάπτυξης νέων εφαρμογών

Οι προηγούμενες κατανεμημένες τεχνολογίες υπέφεραν από ζητήματα διαλειτουργικότητας διότι κάθε προμηθευτής (vendor) υλοποιούσε το δικό του πρότυπο για distributed object messaging. Με την XML σαν το μόνο πρότυπο στα web services, συστήματα φτιαγμένα από διαφορετικές τεχνολογίες όπως η Java και το .Net μπορούν να επικοινωνήσουν μεταξύ τους. Επιπλέον λόγω της απλότητας της XML είναι πολύ πιο εύκολο να γραφτούν νέες εφαρμογές σε μικρό χρονικό διάστημα.

2.4.10 Μειονεκτήματα

Κάθε νέα τεχνολογία πέρα από τα πλεονεκτήματα έχει και τα μειονεκτήματα της η λίστα που ακολουθεί αναφέρει κάποια ζητήματα που πρέπει να ληφθούν υπόψη όταν κάποιος επιλέγει να χρησιμοποιήσει τα Web services:

- Η δυναμική σύνδεση: Ένα Web service απαιτεί τα περιεχόμενα ενός μητρώο υπηρεσιών να είναι έμπιστα. Προς το παρόν μόνο τα ιδιωτικά μητρώα υπηρεσιών μπορούν να έχουν τέτοιο έλεγχο στα περιεχόμενα τους.

- Η ανταπόκριση στις απαιτήσεις: Κάθε φορά που δημιουργείται μια γενική υπηρεσία για να διαχειριστεί μια ποικιλία πελατών και θέλει κάποιος οργανισμός να την χρησιμοποιήσει μπορεί να έρθει αντιμέτωπος με κάποιες εξειδικευμένες για αυτόν απαιτήσεις. Αν ο οργανισμός δεν μπορεί να συμβαδίσει-ευθυγραμμιστεί με αυτές θα πρέπει να αναζητήσει άλλες λύσεις.
- Οι αμετάβλητες διεπαφές: Αν το Web service χρησιμοποιείται από πολλούς πελάτες και δεν υπάρχει μεταξύ του πάροχου υπηρεσιών και αυτών κάποια σύμβαση-συμφωνία για: (i)τις παρεχόμενες μεθόδους, (ii) τις παραμέτρουςτων προαναφερθέντων μεθόδων, (iii) και τα επιστρεφόμενα αποτελέσματα, ή ακόμα αν δεν υπάρχει επικοινωνία μεταξύ πάροχου και πελάτη, η οποιαδήποτε αλλαγή στις παραπάνω τρεις περιπτώσεις θα προκαλούσε την κατάρρευση των προγραμμάτων από την μεριά των πελατών. Μόνο ηπροσθήκη νέων μεθόδων ή η αλλαγή της εσωτερικής λογικής των webservices είναι αυτά που επιδέχεται αυτή η περίπτωση και καμία άλλη αλλαγή.
- Οι επιδόσεις: Τα Web services έχουν χαμηλότερες επιδόσεις σε σχέση με άλλες προσεγγίσεις συστημάτων όπως το COBRA ή η RMI. Αυτό είναι αποτέλεσμα της μεταφοράς των δεδομένων μέσω κειμένου XML, το οποίο περιέχει επιπλέον πληροφορίες-δεδομένα που χρειάζονται στο SOAP πρωτόκολλο. Επίσης η κωδικοποίηση πολλών MIME σε XML προσθέτει πλεονάζοντα byte και χρειάζεται και περισσότερο χρόνο επεξεργασίας, σε σχέση με την αντίστοιχη δυαδική απεικόνιση.
- Οι προσωρινές διασυνδέσεις. Τα Web services χρησιμοποιούν στην πλειοψηφία σαν κύριο πρωτόκολλο μεταφοράς το HTTP το ίδιο ακριβώς πρωτόκολλο που χρησιμοποιείται για την επικοινωνία με μια ιστοσελίδα. Το HTTP είναι ένα πρωτόκολλο το οποίο έχει σχεδιαστεί να εξυπηρετεί χιλιάδες πελάτες ταυτόχρονα χωρίς να κρατάει την κατάσταση μιας σύνδεσης για μεγάλο χρονικό διάστημα. Με το που τελειώσει η μεταφορά των δεδομένων το HTTP κλείνει την σύνδεση. Όλη αυτή η διαδικασία σπαταλάει αρκετό χρόνο και κλείνει τις συνδέσεις των πελατών οι οποίοι επιθυμούν να κάνουν μεγάλο όγκο κλήσεων. Άλλες τεχνολογίες όπως το DCOM, η CORBA και το RMI δεν

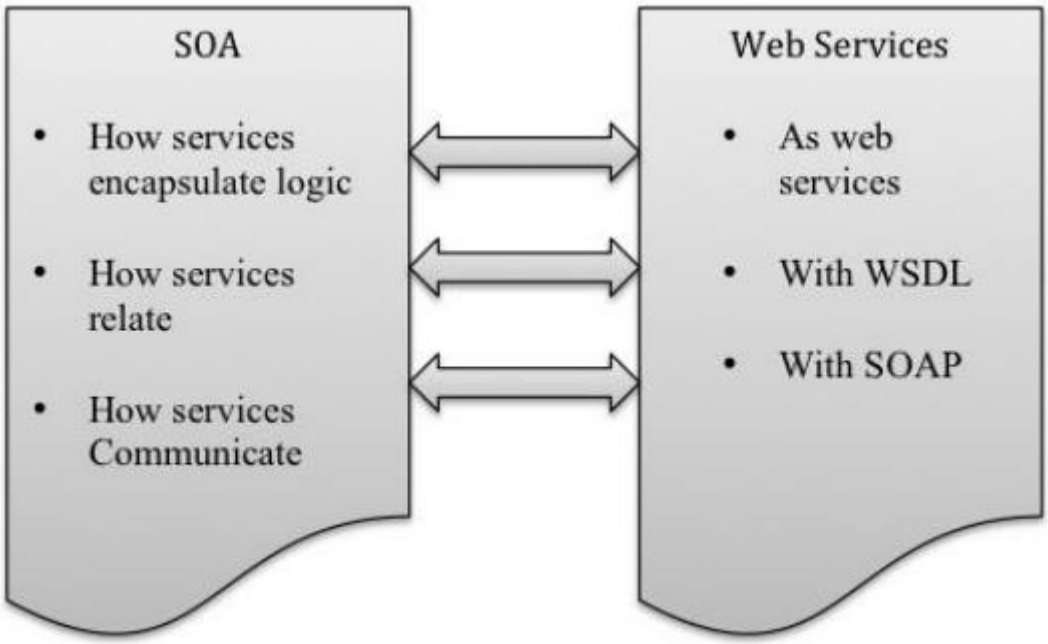
έχουν αυτό το πρόβλημα καθώς διατηρούν την σύνδεση σε όλο τον κύκλο ζωής της εφαρμογής.

- Η μη προσφορά κάποιων χαρακτηριστικών. Τα απλά web service δεν προσφέρουν κάποια χαρακτηριστικά υποδομών και QoS, όπως η ασφάλεια, οι και αρκετά χρόνια. Τα web service προσπαθούν να καλύψουν το σημαντικό κενό αυτό με την εισαγωγή πρόσθετων προδιαγραφών (WS-*).

2.4.11 Web Service και SOA

Όπως έχει προαναφερθεί σε προηγούμενη ενότητα τα web services και η SOA αρχιτεκτονική είναι δύο διαφορετικά πράγματα. Τα Web services αποτελούν μόνο ένα κομμάτι της SOA αρχιτεκτονικής και περιορίζονται στο στρώμα των υπηρεσιών, σχήμα 3 και σχήμα 10. Υπάρχουν όμως και άλλα σημαντικά στρώματα στην αρχιτεκτονική της SOA στα οποία πρέπει να δοθεί εξίσου μεγάλο βάρος, όπως είναι το στρώματα σύνθεσης επιχειρησιακών διαδικασιών και το στρώμα ενσωμάτωσης.

Οι περισσότεροι αναλυτές, κατασκευαστές και συγγραφείς στις μέρες μας συνιστούν μόνο έναν ενδεδειγμένο τρόπο για να καταλάβει κάποιος το τοπίο της SOA αρχιτεκτονικής και αυτός είναι μέσω των Web services. Είναι σημαντικό, όμως, να τονιστεί ότι τα Web service αν και είναι ο ενδεδειγμένος τρόπος για να καταλάβει κάποιος την SOA αρχιτεκτονική στην πράξη δεν είναι ο μοναδικός. Υπάρχουν πολλά παραδείγματα οργανισμών που έχουν εφαρμόσει επιτυχώς την SOA με την χρήση άλλων τεχνολογιών που υπάρχουν. Όπως υπάρχουν και πολλά παραδείγματα χρήση των Web services για την εφαρμογή αρχιτεκτονικών που δεν είναι service-oriented.



Εικόνα 10: Web Service και SOA

2.5 Χορογραφία και ενορχήστρωση υπηρεσιών

Ο συνδυασμός Web services για την δημιουργία υψηλού επιπέδου δια-οργανικών επιχειρησιακών διαδικασιών χρειάζεται πρότυπα τα οποία θα μοντελοποιούν τις αλληλεπιδράσεις. Οι προσεγγίσεις της χορογραφίας (choreography) και της ενορχήστρωσης (orchestration) υπηρεσιών ανήκουν στο στρώμα σύνθεσης επιχειρησιακών διαδικασιών της αρχιτεκτονικής SOA (βλ. σχήμα 3) και είναι τα πρότυπα που μοντελοποιούν τις παραπάνω αλληλεπιδράσεις. Οι επόμενες ενότητες θα αναφερθούμε εν συντομία στην χορογραφία και την ενορχήστρωση των υπηρεσιών.

2.5.1 Ενορχήστρωση (Orchestration)

Ενορχήστρωση περιγράφει την αυτόματη ρύθμιση (automated arrangement), το συντονισμό και τη διαχείριση πολύπλοκων ηλεκτρονικών συστημάτων υπολογιστών, middleware και υπηρεσιών .

Η ενορχήστρωση των υπηρεσιών ιστού επιτρέπει στις υπηρεσίες να είναι συνδεδεμένες μεταξύ τους σε προκαθορισμένα σχέδια και να εκτελούνται ακολουθώντας τα βήματα ενός προκαθορισμένου σεναρίου. Αυτά τα σενάρια

συνήθως αντιπροσωπεύουν επιχειρησιακές διαδικασίες. Τα σενάρια περιγράφουν την αλληλεπίδραση μεταξύ των εφαρμογών αναγνωρίζοντας τα μηνύματα, καθορίζοντας τους ρόλους μεταξύ των εφαρμογών και περιγράφοντας την λογική σειρά της ανταλλαγής των μηνυμάτων.

Τα προγράμματα που εκτελούν τέτοιου είδους σενάρια ονομάζονται **μηχανές ενορχήστρωσης**. Οι μηχανές αυτές λειτουργούν ως κεντρικές authorities για να συγχρονίσουν την επικοινωνία μεταξύ των υπηρεσιών.

2.5.2 Χορογραφία (Choreography)

Η χορογραφία υπηρεσιών (service choreography) είναι μια μορφή σύνθεσης υπηρεσιών στην οποία το πρωτόκολλο αλληλεπίδραση μεταξύ των διαφόρων υπηρεσιών ορίζεται από μια παγκόσμια προοπτική.

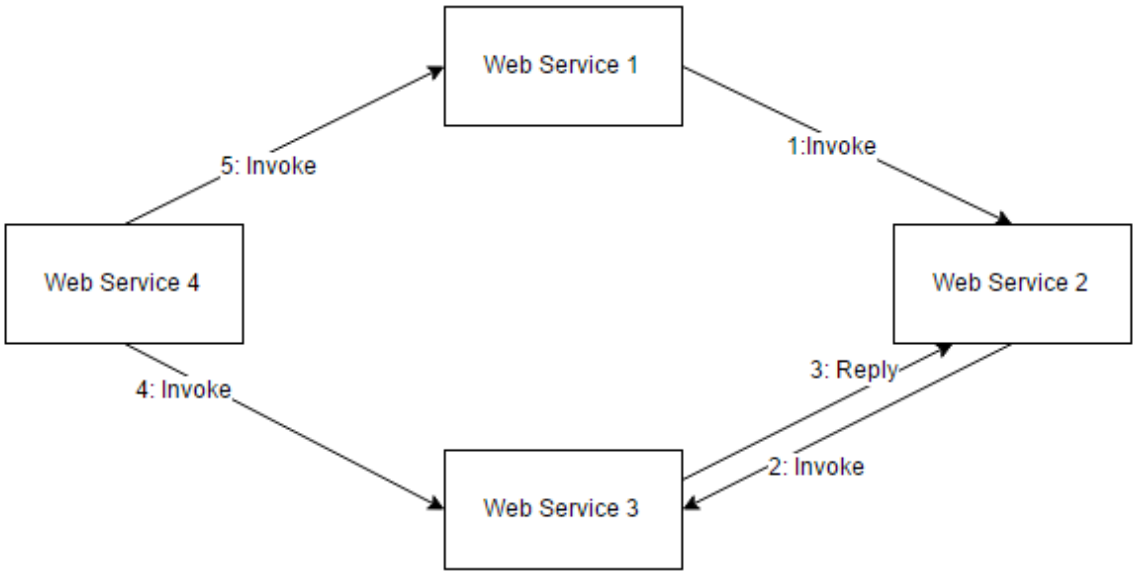
Η χορογραφία ασχολείται με την επικοινωνία μεταξύ των υπηρεσιών ιστού. Μια χορογραφία απεικονίζει συνολικά από μια εξωτερική άποψη την παρατηρήσιμη συμπεριφορά των υπηρεσιών που θέλουν να προσέλθουν σε μια συμφωνία. Το χαρακτηριστικό της χορογραφίας είναι ότι δεν υπάρχει ένα κεντρικό σημείο ελέγχου. Μια abstract χορογραφία περιλαμβάνει την περιγραφή των τύπων δεδομένων που χρησιμοποιούνται και τις συνθήκες κάτω από τις οποίες ένα μήνυμα θα σταλεί. Το πλεονέκτημα της abstract χορογραφίας είναι ότι, πιθανότατα, μπορεί να επαναχρησιμοποιηθεί. Συνήθως βέβαια απαιτείται και μια πιο λεπτομερής περιγραφή των πληροφοριών που θα ανταλλάσονται. Για το λόγο αυτό, η W3C προσφέρει δύο τύπους χορογραφίας, την portable και την concrete, που επεκτείνουν τον ορισμό της abstract χορογραφίας. Η portable χορογραφία περιλαμβάνει ορισμούς της φυσικής δομής των πληροφοριών που ανταλλάσσονται. Η concrete χορογραφία επεκτείνει μια portable χορογραφία περιλαμβάνοντας επίσης URLs, συγκεκριμένους κανόνες ασφαλείας κ.λ.π. Υπάρχουν ήδη αρκετές γλώσσες περιγραφής μιας χορογραφίας. Κάποιες από αυτές είναι η BPMN, BPEL4CHOR, LET´S DANCE, WS-CDL. Οι πρώτες τρεις είναι σε ένα πιο abstract επίπεδο ενώ η WS-CDL μας επιτρέπει να δημιουργήσουμε περισσότερο λεπτομερή μοντέλα.

2.5.3 Σύγκριση ενορχήστρωσης και χορογραφίας

Η ενορχήστρωση, συνήθως χρησιμοποιείται συνήθως σε ιδιωτικές επιχειρηματικές διαδικασίες. Μια κεντρική διαδικασία (coordinator) λαμβάνει τον ελέγχο των εμπλεκομένων Web services και διευθύνει την εκτέλεση των διαφόρων λειτουργιών με τα Web services που ενέχονται σε αυτές λειτουργίες. Τα εμπλεκόμενα Web Services δεν "γνωρίζουν" ότι συμμετέχουν σε μια σύνθετη διαδικασία και ότι λαμβάνουν μέρος σε μια ανώτερου επιπέδου επιχειρηματική διαδικασία. Μόνο ο κεντρικός συντονιστής της ενορχήστρωσης είναι ενήμερος για τον στόχος της, έτσι ώστε η ενορχήστρωση να είναι συγκεντρωτική και με ρητούς ορισμούς των λειτουργιών και της σειράς κλήσης των Web services. (Βλέπε σχήμα 11).

Εικόνα 11: Σύνθεση υπηρεσιών με ενορχήστρωση

Η Χορογραφία, αντίθετα, δεν στηρίζεται σε ένα κεντρικό συντονιστή. Αντίθετα, κάθε Web service που εμπλέκεται στη χορογραφία ξέρει ακριβώς πότε να εκτελέσει τις εργασίες του και με ποίους θα αλληλεπιδράσει. Η χορογραφία είναι μια συλλογική προσπάθεια και εστιάζει στην ανταλλαγή μηνυμάτων σε δημόσιες επιχειρηματικές διαδικασίες. Όλοι οι συμμετέχοντες στη χορογραφία πρέπει να γνωρίζουν την επιχειρηματική διαδικασία, τις εργασίες που εκτελούν, τα μηνύματα προς ανταλλαγή καθώς και την χρονική στιγμή της ανταλλαγής των μηνυμάτων. (Βλέπε σχήμα 12).



Εικόνα 12: Σύνθεση υπηρεσιών με την ενορχήστρωση

Ένα απλό παράδειγμα της διαφοράς μεταξύ της χορογραφίας και της ενορχήστρωσης είναι το μοντέλο ελέγχου της ροής της πληροφορίας σε μια διασταύρωση. Στην περίπτωση αυτή, η ενορχήστρωση θα ενέπλεκε ένα φανάρι κυκλοφορίας (ελέγχοντας πότε κάθε όχημα περνάει την διασταύρωση), ενώ η χορογραφία θα συγκρινόταν με μια κυκλική πλατεία, εφόσον δεν υπάρχει κάποιος κεντρικός έλεγχος υπάρχουν μόνο κάποιοι γενικοί κανόνες ορισμένοι που προδιαγράφουν ότι τα οχήματα που πλησιάζουν τη διασταύρωση πρέπει να περιμένουν μέχρι να υπάρχει διαθέσιμος χώρος για να εισέλθουν στην πλατεία, στρίβουν δεξιά και στη συνέχεια εξέρχονται βάση της κατάλληλης για αυτά εξόδου.

3 Τεχνολογίες/Πρωτόκολλα ανάπτυξης εφαρμογής

Σε αυτό το κεφάλαιο, γίνεται αναφορά στις τεχνολογίες και στα πρωτόκολλα ανάπτυξης που χρησιμοποιηθήκαν κατά την ανάπτυξη της εφαρμογής.

3.1 PHP 5

Εικόνα 13 PHP5 logo

Η ανάπτυξη της γλώσσας PHP ξεκίνησε το 1994 όταν ο Rasmus Lerdorf έγραψε με την βοήθεια της γλώσσας C μια σειρά από εκτελέσιμα αρχεία Common Gateway Interface (CGI), τα οποία και χρησιμοποίησε για να συντηρήσει την προσωπική του ιστοσελίδα. Το εγχείρημα αυτό επεκτάθηκε με την προσθήκη της ικανότητας να συνεργάζεται με φόρμες ιστού (web forms) και να επικοινωνεί με βάσεις δεδομένων (databases), και ονομάστηκε "Personal Home Page/Forms Interpreter" ή PHP/FI.

Η PHP/FI μπορούσε να χρησιμοποιηθούν για την κατασκευή απλών, δυναμικών διαδικτυακών εφαρμογών. Ο Lerdorf για νσ επιταχύνει τον εντοπισμό προβλημάτων (bug) και την βελτίωση του κώδικα δημοσιοποίηση την έκδοση 1.0 στην ομάδα συζήτησης Usenet (comp.infosystems.www.authoring.cgi) στης 8 Ιουνίου του 1995. Η έκδοση εκείνη περιείχε μεταβλητές τύπου Perl, χειρισμό φορμών (form), καθώς και την ικανότητα ενσωμάτωσης HTML (embed HTML). Η σύνταξη έμοιαζε με εκείνη της Perl , αλλά ήταν απλούστερη, πιο περιορισμένη και λιγότερο συνεπής (consistent).

Η PHP, αρχικά, δεν γράφτηκε με σκοπό να εξελιχθεί σε μια νέα γλώσσα προγραμματισμού, και εξελίχθηκε οργανικά. Μια ομάδα ανάπτυξης άρχισε να σχηματίζεται και, μετά από μήνες εργασίας και beta testing , κυκλοφόρησε επίσημα PHP / FI 2 τον Νοέμβριο του 1997.

Μια από τις επικρίσεις της PHP είναι ότι δεν είχε σχεδιαστεί με αποτέλεσμα να παρουσιάζει ασυνεπείς ονομασία των συναρτήσεων (functions) και αντιφατική διάταξη των παραμέτρων (parameters) τους.

Οι Zeev Suraski και Andi Gutmans ξανάγραψαν το πρόγραμμα ανάλυσης (parser) το 1997 και δημιούργησαν τη βάση της PHP 3, αλλάζοντας το όνομά της γλώσσας στον αναδρομικό ακρωνύμιο του PHP: Hypertext Preprocessor. Μετά από αυτό οι Suraski και Gutmans ξεκίνησαν μια νέα επανεγγραφή του πυρήνα της PHP, παράγοντας το Zend Engine το 1999. Επίσης, ίδρυσαν το Zend Technologies στο Ramat Gan του Ισραήλ .

Στις 22 Μαΐου του 2000, κυκλοφόρησε η PHP 4 , άπό την Zend Engine και μέχρι τον Αύγουστο του 2008 έφτασε στην έκδοση 4.4.9 . Η PHP 4 δεν είναι πλέον υπό ανάπτυξη, ούτε θα κυκλοφορήσουν ενημερώσεις ασφαλείας (security updates).

Στης 13 Ιουλίου του 2004 κυκλοφόρησε η έκδοση PHP 5, η οποία περιλαμβάνει νέα χαρακτηριστικά, όπως η βελτιωμένη υποστήριξη για αντικειμενοστραφή προγραμματισμό, μια επέκταση (extension) των αντικειμένων PHP Data (PDO), η οποία εισάγει μια ελαφριά (lightweight) και συνεπή (consistent) διεπαφή για πρόσβαση σε βάσεις δεδομένων, και πολλές βελτιώσεις απόδοσης .

Από το 2008 η PHP 5 αποτελεί την μόνη σταθερή έκδοση υπό ανάπτυξη. Η εκ των υστέρων στατική δέσμευση (late static binding) έλλειπε από την PHP και προστέθηκε με την έκδοση 5.3.

Πολλά μεγάλης σημασίας (high-profile) έργα ανοικτού κώδικα έπαψαν να υποστηρίζουν την PHP 4 από της 5 Φεβρουαρίου του 2008, λόγο της πρωτοβουλίας GoPHP5, που παρέχεται από μια κοινοπραξίας προγραμματιστών PHP οι οποίοι προωθούν την μετάβαση από την PHP4 στην PHP5.

Με την πάροδο του χρόνου, οι διερμηνείς (interpreters) της PHP έγιναν διαθέσιμοι για τα περισσότερα λειτουργικά συστήματα 32-bit και 64-bit, είτε δημιουργούντας τους από των πηγαίο κώδικα της PHP, είτε χρησιμοποιώντας προ - μεταγλωττισμένα εκτελέσιμα πακέτα

3.2 Yii 2

Εικόνα 14 Yii logo

Yii είναι ένα υψηλής απόδοσης, component-based πλαίσιο (framework) για PHP με στόχο την ταχεία ανάπτυξη σύγχρονων εφαρμογών Web

Yii is a high performance, component-based PHP framework for rapidly developing modern Web applications. Το όνομα Yii μπορεί να θεωρηθεί ως το ακρωνύμιο για το Yes It Is!. Το Yii 2 παραμένοντας πιστό στο πνεύμα του Yii παραμένει ένα απλό , γρήγορο και άκρως επεκτάσιμο πλαίσιο (framework) PHP.

Τα βασικότερα πλεονεκτήματα του Yii

1. Το Yii είναι ένα γενικό πλαίσιο (generic framework) προγραμματισμού στο Web
2. Το Yii μπορεί να χρησιμοποιηθεί για την ανάπτυξη όλων των ειδών των εφαρμογές Web που βασίζονται στην PHP
3. Η αρχιτεκτονική του Yii είναι component-based και διαθέτει μια εξελιγμένη υποστήριξη για caching
4. Το Yii είναι ιδιαίτερα κατάλληλο για την ανάπτυξη εφαρμογών μεγάλης κλίμακας, όπως πύλες (portals), forums, συστήματα διαχείρισης περιεχομένου (CMS), έργα του ηλεκτρονικού εμπορίου (e-commerce), RESTful υπηρεσίες (services) Web κ.τ.λ.
5. Το Yii υλοποιεί το MVC (Model-View-Controller) πρότυπο σχεδιασμού
6. Ισχυρή ομάδα ανάπτυξης του πυρήνα του έργου
7. Υποστήριξη μέσω forum στην ηλεκτρονική σελίδα του yii .

3.3 Apache 2

Εικόνα 15 Apache logo

Το έργο Apache HTTP Server αποτελεί μια συλλογική προσπαθεί με στόχο την ανάπτυξη μιας εύρωστης, εμπορικής εκτέλεσης (implementation) ενός HTTP διακομιστή (server) με πληθώρα λειτουργιών (featureful), ο πηγαίος κώδικας του οποίο θα είναι ελεύθερα διαθέσιμος. Το έργο διευθύνεται από κοινού από μια ομάδα εθελοντών που βρίσκονται σε όλο τον κόσμο και χρησιμοποιούν το Διαδίκτυο (Internet) και τον Παγκόσμιο Ιστό (Web) για να επικοινωνούν, να σχεδιάζουν, και να αναπτύσσουν το διακομιστή (server) και τη σχετική τεκμηρίωση του (documentation). Το έργο αυτό αποτελεί μέρος του Apache Foundation Λογισμικού. Επιπλέον, εκατοντάδες χρήστες έχουν συνεισφέρει ιδέες, κώδικα, και στην τεκμηρίωση του έργου. Η ενότητα αυτή έχει σκοπό να περιγράψει εν συντομία την ιστορία του διακομιστή Apache HTTP και να αναγνωρίσει τις πολλές συνεισφορές του.

Η ιστορία του Apache ξεκινάει το Φεβρουάριο του 1995 με τον HTTP δαίμων δημοσίου τομέα (public domain HTTP daemon) τον οποίο είχε αναπτύξει ο Rob McCool για το Εθνικό Κέντρο για Εφαρμογές Υπερυπολογιστών (National Center for Supercomputing Applications). Μετά την αποχώρηση του Rob McCool από το NCSA οι webmasters ανέπτυσαν ο καθένας τις επεκτάσεις και διορθώσεις τους. Μια μικρή ομάδα αυτών συγκεντρώθηκαν με σκοπό να συντονίσουν της αλλαγές τους. Οι Brian Behlendorf και Cliff Skolnick επέλεξαν την ομάδα βρήκαν κοινόχρηστο χώρο για διαμοιρασμό της πληροφορίας και έδωσαν κωδικούς σύνδεσης στα βασικά μηχανήματα πού γίνονταν ο προγραμματισμός και βρίσκονταν στην Bay Area της Καλιφόρνια, με εύρος ζώνης που δώρισε η HotWired. Μέχρι το τέλος του Φεβρουαρίου , οκτώ βασικοί συνεργάτες αποτέλεσε τη βάση του αρχικού Apache Group.

Το αποτέλεσμα της ομάδας αυτής ήταν η πραγματοποίηση του πρώτου διακομιστή Apache, τον Απρίλιο του 1995. Αν και ο αρχικός διακομιστής Apache αποτέλεσε μεγάλη επιτυχία η ομάδα γνώριζε ότι η βάση τους κώδικα χρειαζόταν μια γενική επανεξέταση και επανασχεδιασμό. Κατά την περίοδο Μάιος-Ιούνιος 1995, και ενώ ο Rob Hartill και η υπόλοιπη ομάδα επικεντρώνονταν στην υλοποίηση νέων χαρακτηριστικών για την 0.7.x (όπως οι pre-forked child processes) και την υποστήριξη της ταχέως αναπτυσσόμενης κοινότητας των χρηστών Apache, ο Robert Thau σχεδίασε την νέα αρχιτεκτονική του διακομιστή (η οποία ονομαζόταν Shambhala) η οποία περιελάβανε μια σπονδυλωτή δομή και API για την καλύτερη επεκτασιμότητα , pool-based κατανομή της μνήμης, και ένα προσαρμοστικό μοντέλο για pre-forking process. Η ομάδα πέρασε σε αυτή τη νέα βάση εξυπηρετητή (server base) τον Ιούλιο και πρόσθεσε τα χαρακτηριστικά από την 0.7.x, με αποτέλεσμα τον Apache 0.8.8 (και των συναφών του) τον Αύγουστο . Μετά από εκτεταμένες δοκιμές beta, πολλές πόρτες (ports) σε αφανής πλατφόρμες, και μια νέα σειρά τεκμηριώσεων (από τον David Robinson), και την προσθήκη πολλών δυνατοτήτων στα βασιμά modules, ο Apache 1.0 κυκλοφόρησε την 1 Δεκεμβρίου του 1995.

Σε λιγότερο από ένα χρόνο μετά την συγκρότηση της ομάδες, ο διακομιστής Apache ξεπέρασε των NCSA's httpd ως ο No1 διακομιστή στο Διαδίκτυο και σύμφωνα με έρευνα του Netcraft διατηρεί αυτή την θέση μέχρι και σήμερα.

Το 1999 τα μέλη του Apache Group ίδρυσαν το Apache Software Foundation με σκοπό να παρέχουν οργανωτική, νομική και οικονομική υποστήριξη για τον διακομιστή Apache HTTP. Το ίδρυμα έχει θέσει το λογισμικό σε στέρεες βάσεις για τη μελλοντική ανάπτυξη, και έχει επεκτείνει σε μεγάλο βαθμό τον αριθμό των έργων λογισμικού Ανοικτού Κώδικα (Open Source software projects), τα οποία εμπίπτουν κάτω από την ομπρέλα αυτού του Ιδρύματος .

3.4 Nginx

Εικόνα 16 Nginx logo

Το Nginx (προφέρεται engine-x) αποτελεί ένα δωρεάν, ανοικτού-κώδικα, διακομηστή HTTP υψηλών επιδόσεων και αντίστροφο proxy, καθώς και έναν IMAP/POP3 διακομιστή proxy (proxy server). Ο Igor Sysoev ξεκίνησε την ανάπτυξη του Nginx το 2002, και η πρώτη δημόσια έκδοση του έγινε το 2004. Σήμερα το Nginx φιλοξενεί περίπου 12,18% (22.M) των ενεργών ιστοσελίδων σε όλες της διευθύνσεις διαδικτύου (domains). Το Nginx είναι γνωστό για τις υψηλές επιδόσεις, την σταθερότητα, τα πλούσια χαρακτηριστικά (feature), απλή ρύθμιση (simple configuration) και χαμηλή κατανάλωση πόρων

Nginx είναι ένας από τους ελάχιστους διακομιστές που αντιμετωπίζουν το πρόβλημα C10K. Σε αντίθεση με τις παραδοσιακές διακομιστές, ο Nginx δεν βασίζεται σε threads για τη διεκπεραίωση των αιτήσεων (requests). Αντ' αυτού χρησιμοποιεί μια αρχιτεκτονική η οποία εξαρτάται από τα event (ασύγχρονη) και κατά συνέπεια έχει πολύ καλύτερα κλιμάκωση. Αυτή η αρχιτεκτονική χρησιμοποιεί μικρές, αλλά το σημαντικότερα προβλέψιμες ποσότητες μνήμης υπό φορτίο

Ακόμα και αν μια εφαρμογή δεν χρειάζεται να χειρίζεται χιλιάδες ταυτόχρονες αιτήσεις, μπορεί να επωφεληθείς από της υψηλές αποδόσεις του Nginx και το μικρό ίχνος μνήμης. Το Nginx κλιμακώνει (scales) προς όλες τις κατευθύνσεις: από το μικρότερο VPS έως συστάδες εξυπηρετητών (clusters of servers).

Το Nginx χρησιμοποιείται από πολλούς μεγάλου οργανισμούς όπως Netflix, Hulu, Pinterest, CloudFlare, Airbnb, WordPress.com, GitHub, SoundCloud, Zynga, Eventbrite, Zappos, Media Temple, Heroku, RightScale, Engine Yard και MaxCDN

3.5 Linux

Εικόνα 17 Linux logo

Τα Linux είναι σαν τα Unix και επί το πλείστων συμβατό με το λειτουργικό σύστημα POSIX το οποίο φτιάχτηκε σύμφωνα με το μοντέλο ανάπτυξης και διανομής ελεύθερου και ανοικτού κώδικα. Το πιο καθοριστικό στοιχείο των Linux είναι ο πυρήνας (kernel) των Linux, ένας λειτουργικός πυρήνας κυκλοφόρησε για πρώτη φορά στις 5 Οκτωβρίου του 1991 από τον Linus Torvalds. Το Ίδρυμα Ελεύθερου Λογισμικού χρησιμοποιεί το όνομα GNU/Linux για να περιγράψει το λειτουργικό σύστημα , το οποίο έχει οδηγήσει σε κάποια διαμάχη.

Το Linux αρχικά αναπτύχθηκε ως ένα ελεύθερο λειτουργικό σύστημα για προσωπικούς υπολογιστές που βασιζόταν στην αρχιτεκτονική x86 της Intel, αλλά έκτοτε έχει μεταφερθεί σε περισσότερες πλατφόρμες υλικού από οποιοδήποτε άλλο λειτουργικό σύστημα. Αποτελεί το κυρίαρχο λειτουργικό σύστημα σε διακομιστές (server) και άλλα μεγάλα συστήματα όπως υπολογιστές mainframe και υπερυπολογιστές (supercomputers), όμως χρησιμοποιείται μόλις από το 1.5% τον επιτραπέζιων υπολογιστών (desktop computers). Το Linux τρέχει επίσης σε ενσωματωμένα (embedded) συστήματα, τα οποία αποτελούν συσκευές των οποίων το λειτουργικό σύστημα είναι συνήθως ενσωματωμένο στο firmware και είναι κομμένα και ραμμένα πάνω στο σύστημα, αυτό περιλαμβάνει τα κινητά τηλέφωνα, τα tablet υπολογιστών, δρομολογητές δικτύου, αυτοματισμούς εγκατάστασης, τηλεοράσεις και κονσόλες βιντεοπαιχνιδιών. Το Android, το πιο ευρέως χρησιμοποιούμενο λειτουργικό σύστημα για tablet και smartphones , είναι χτισμένο πάνω σε έναν πυρήνα Linux.

Η ανάπτυξη του Linux είναι ένα από τα πιο γνωστά παραδείγματα του συνεργασίας για την ανάπτυξη ελεύθερου και ανοικτού κώδικα λογισμικού. Ο υποκείμενος πηγαίος κώδικας μπορεί να χρησιμοποιηθεί, να τροποποιηθεί , και να διανεμηθεί - εμπορικά ή μη - από οποιονδήποτε κάτοχο άδειας, όπως η GNU General Public License . Τυπικά , το Linux είναι συσκευασμένο σε μια μορφή που είναι γνωστή ως μια διανομή Linux, και χρησιμοποιείται τόσο για desktop , όσο και για διακομιστές (server). Μερικοί δημοφιλής διανομής Linux είναι οι include Debian, Ubuntu, Linux Mint, Fedora, openSUSE, Arch Linux, και το εμπορική Red Hat Enterprise Linux και SUSE Linux Enterprise Server. Οι Διανομές Linux περιλαμβάνουν συνήθως τον πυρήνα του Linux , υποστηρικτικές υπηρεσίες και βιβλιοθήκες και συνήθως κάποια εφαρμογή η οποία εκπληρώνει ην προβλεπόμενη χρήση της διανομής .

Οι διανομές που αφορούν τους επιτραπέζιους υπολογιστές περιλαμβάνουν συνήθως X11, μια εφαρμογή του Wayland ή του Mir ως σύστημα παραθύρων , και ένα συνοδευτικό περιβάλλον εργασίας όπως το GNOME ή το KDE Software Compilation. Κάποιες από τις διανομές αυτές μπορεί να περιλαμβάνουν κάποιες επιφάνειες εργασίας με λιγότερες απαιτήσεις ως προς τους πόρους όπως το LXDE or Xfce, για χρήση σε παλαιότερα ή λιγότερα ισχυρούς υπολογιστές. Διανομές που προορίζονται για διακομιστές μπορούν να παραλείψουν όλα τα γραφικά περιβάλλοντα από την κανονική εγκατάσταση , και αντ' αυτού να περιλαμβάνουν άλλο λογισμικό για να δημιουργηθεί και να λειτουργήσει μια solution stack, όπως LAMP. Επειδή το Linux διανέμεται ελεύθερα , ο καθένας μπορεί να δημιουργήσει μια διανομή για κάθε προβλεπόμενη χρήση .

3.6 MySQL

Εικόνα 18 MySQL logo

Η MySQL είναι η πιο δημοφιλής βάση δεδομένων ανοιχτού κώδικα στον κόσμο. Μ αποδεδειγμένη απόδοση, αξιοπιστία και ευκολία χρήσης η MySQL έχει αναδειχτεί σε πρώτη επιλογή όσον αφορά την βάση δεδομένων για web - based εφαρμογές και χρησιμοποιείται από σημαντικούς ιδιότητες ιστού (web properties) συμπεριλαμβανομένων του Facebook, Twitter, YouTube, Yahoo! Και πολλών άλλων.

Η Oracle ηγείται της καινοτομία της MySQL παρέχοντας νέες δυνατότητες οι οποίες θα ενδυναμώσουν την νέα γενιά web, cloud, mobile και ενσωματωμένων εφαρμογών.

3.7 Bootstrap

Εικόνα 19 Bootstrap logo

Δημιουργήθηκε, αρχικά , από έναν σχεδιαστή και έναν προγραμματιστή στο Twitter, και πλέον το Bootstrap έχει γίνει ένα από τα πιο δημοφιλή front-end πλαίσια (frameworks) και προγράμματα ανοιχτού κώδικα στον κόσμο .

Το Bootstrap δημιουργήθηκε στον οργανισμό Twitter στα μέσα το 2010 από τους @mdo και @fat. Πριν γίνει ένα πλαίσιο ανοιχτού κώδικα, το Bootstrap ήταν γνωστό ως Twitter Blueprint.

Λίγους μήνες μετά την έναρξη της ανάπτυξης, το Twitter πραγματοποίησε την πρώτη του Hack Week και το έργο εκτοξεύθηκε καθώς προγραμματιστές όλων των επιπέδων και δεξιοτήτων ανέλαβαν δράση χωρίς οποιαδήποτε εξωτερική καθοδήγηση. Χρησίμευε ως οδηγός στυλ (style guide) για τα εσωτερικά εργαλεία ανάπτυξης της εταιρεία για πάνω από ένα χρόνο πριν από τη δημόσια κυκλοφορία του , και συνεχίζει να το κάνει μέχρι και σήμερα .

Αρχικέ κυκλοφόρησε την Παρασκευή 19 Αυγούστου του 2011, και από τότε κυκλοφόρησαν περισσότερες από είκοσι εκδόσεις, περιλαμβανομένων δύο μεγάλων βελτιώσεων με τις v2 και v3. Με το Bootstrap 2 προστέθηκε σε ολόκληρο το πλαίσιο (framework) λειτουργικότητα ανταπόκρισης (responsive functionality) ως προαιρετική λειτουργικότητα. Βασιζόμενοι πάνω στην Bootstrap 2, στην Bootstrap 3 ξαναγράφτηκε, με γνώμονα τις κινητές συσκευές, η βιβλιοθήκη ώστε η λειτουργικότητα ανταπόκρισης (responsive) να υπάρχει εξ ορισμού (default)

3.8 XML

Εικόνα 20 XML logo

Η XML είναι μια γλώσσα σήμανσης για έγγραφα που περιέχουν δομημένες πληροφορίες.

Οι Δομημένες πληροφορίες μπορούν να περιέχουν (λέξεις, εικόνες, κλπ.) και κάποια ένδειξη του ρόλου που παίζει το περιεχόμενο (για παράδειγμα, το

περιεχόμενο σε ένα τίτλο έχει διαφορετική έννοια από το περιεχόμενο σε μια υποσημείωση, που σημαίνει κάτι διαφορετικό από το περιεχόμενο σε ένα σχήμα λεζάντα ή το περιεχόμενο σε έναν πίνακα βάσης δεδομένων, κ.λπ.). Σχεδόν όλα τα έγγραφα έχουν κάποια δομή.

Μια γλώσσα σήμανσης είναι ένας μηχανισμός για τον εντοπισμό δομών σε ένα έγγραφο. Η προδιαγραφή XML καθορίζει ένα πρότυπο τρόπο για να προσθέσετε σήμανσης στα έγγραφα

3.9 JSON

Εικόνα 211 JSON logo

Ο όρος JSON ή αλλιώς Javascript Object Notation, είναι μια μορφή κειμένου για την απεικόνιση αντικειμένων/δομών η οποία είναι αναγνώσιμη από τον άνθρωπο, είναι εύκολη και συνάμα ελαφρύς τρόπος μεταφοράς δεδομένων, είναι εύκολο και για τις μηχανές να το διαβάσουν και να το δημιουργήσουν. Βασίζεται σε ένα υποσύνολό της γλώσσας Javascript, και στο πρότυπο ECMA-262 3η έκδοση.

Το JSON εμπεριέχει 2 δομές:

- Μια συλλογή από ζεύγη όνομα – τιμή
- Μια ταξινομημένη λίστα τιμών

4 Παρουσίαση

Σε αυτό το κεφάλαιο, πραγματοποιείται ανάλυση της εφαρμογής που αναπτύχθηκε, και παρουσίαση της χρήσης της.

Η βασική λειτουργία της εφαρμογής είναι επικοινωνία με διάφορους παρόχους δεδομένων ποδοσφαίρου,, με σκοπό την πληρέστερη, ποιοτικότερη και καλύτερη πληροφόρησή, των παιχτών στοιχημάτων ποδοσφαίρου, είτε αυτό είναι ηλεκτρονικό είτε σε κάποιο πρακτορείο.

Η εφαρμογή που υλοποιήθηκε και παρουσιάζεται, στα πλαίσια αυτής της διπλωματικής εργασίας, βρίσκεται στα πρώτα στάδια λειτουργίας της. Δηλαδή, περιλαμβάνει δοκιμαστικά 2 παρόχους δεδομένων, και κάνει χρήση των δεδομένων αυτών που παρέχουν ελεύθερα, (χωρίς πληρωμή). Τα Δεδομένα αυτά τα οποία λαμβάνει, τα επεξεργάζεται βάση προκαθορισμένων ερωτημάτων και επιστρέφει το αποτέλεσμα αυτών σε μορφή που θα επιλέξει ο χρήστης, (JSON ή XML).

Σε μελλοντική ανάπτυξη της εφαρμογή μπορεί να δημιουργηθεί γραφικό περιβάλλον χρήστη με περισσότερες δυνατότητες όπως η δημιουργία σύνθετων ερωτημάτων, με συνδυασμό τον υπάρχον των ή κάποιων καινούργιων. Καθώς επίσης μπορεί να αυξηθεί ο αριθμός των παροχών που αντλεί δεδομένα η εφαρμογή. Με αποτέλεσμα την ακόμα πιο ποιοτική και πληρέστερη πληροφόρηση του παίκτη (χρήστη) της εφαρμογής.

Η SOA αρχιτεκτονική είναι μια πολύ-επίπεδη (multi-tier) αρχιτεκτονική και έχοντας αυτό ως πρότυπο η εφαρμογή που αναπτύχθηκε βασίστηκε σε τέσσερα επίπεδα. Τα επίπεδα αυτά είναι τα εξής:

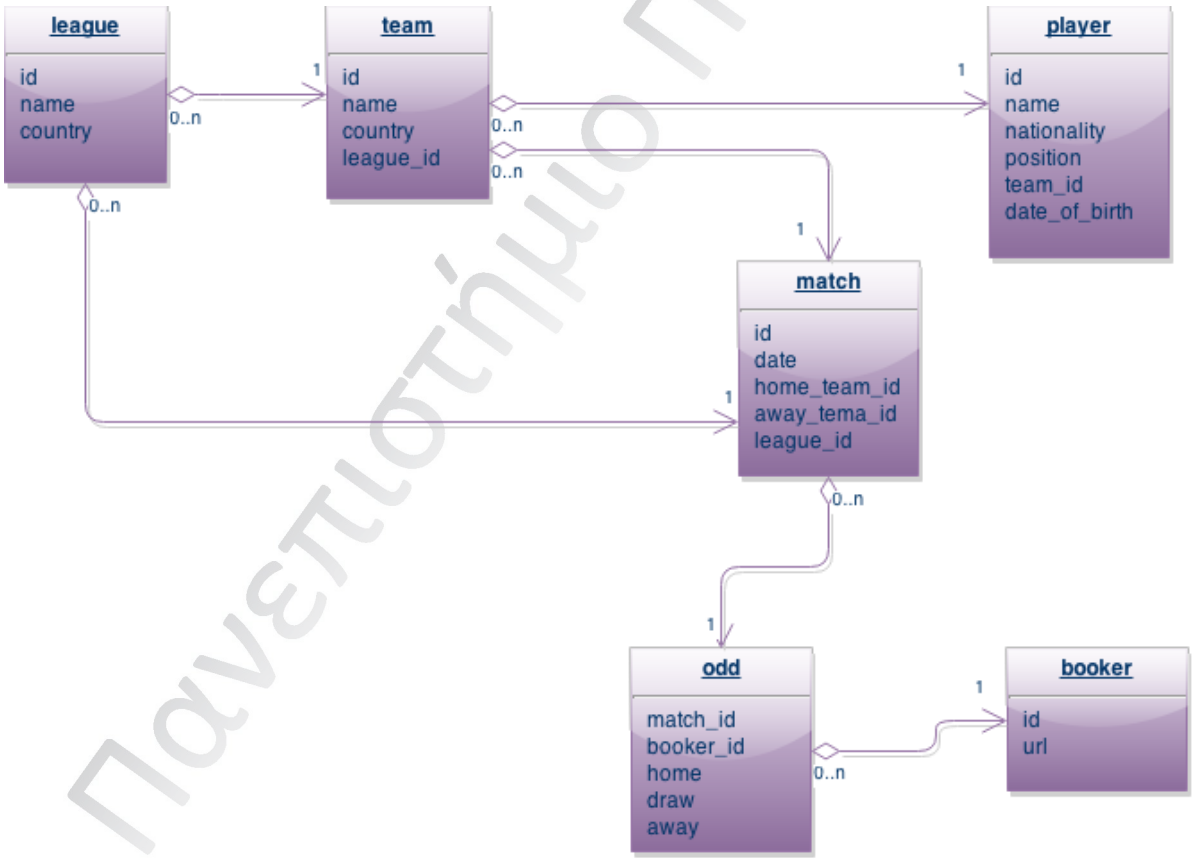
- Τα δεδομένα (Database)
- Τα απλά Web-services
- Τα σύνθετα Web Services
- Και το γραφικό περιβάλλον

Τα επίπεδα αναλύονται στις επόμενες ενότητες.

4.1 Βάση δεδομένων

Σημαντικό στοιχείο της εφαρμογής που αναπτύχθηκε αποτελεί το πρώτο επίπεδο της, το επίπεδο των δεδομένων. Για την διαχείριση επαναλαμβανόμενων κλήσεων/αιτήσεων, πού έχουν να κάνουν κυρίως με παρελθοντικά δεδομένα, αναπτύχθηκε μια βάση δεδομένων.

Στην οποία και αποθηκεύονται τα δεδομένα όπως θα τα επιστρέφαμε στον χρήστη που έστειλε το ερώτημα, με σκοπό την ταχύτερη απάντηση στο όμοιο ερώτημα, και την αποφυγή τεχνικών περιορισμών που θέτουν οι πάροχοι όπως τον αριθμό τον ερωτημάτων που μπορούν να δεχτούν για τα ελεύθερα διαμοιραζόμενα δεδομένα που προσφέρουν.



Διάγραμμα 1 Απεικόνιση βάσης δεδομένων

Στο παραπάνω διάγραμμα (Διάγραμμα 1), μπορούμε να δούμε τις ιδιότητες αλλά και τις σχέσεις που έχουν οι βασικές οντότητες που εμπεριέχονται στην εφαρμογή μας.

- **Οι διοργανώσεις** (league)
  - id, name, country
- **Οι ομάδες** (team)
  - id, name, country, league_id
- **Οι παίκτες** (player)
  - id, name, nationallity, position, team_id, date_of_birth
- **Οι αγώνες** (match)
  - id, date, home_team_id, away_team_id, league_id
- **Οι αποδώσεις** (odd)
  - match_id, booker_id, home, draw, away
- **Οι στοιχηματικες πηγές** (booker)
  - id, url

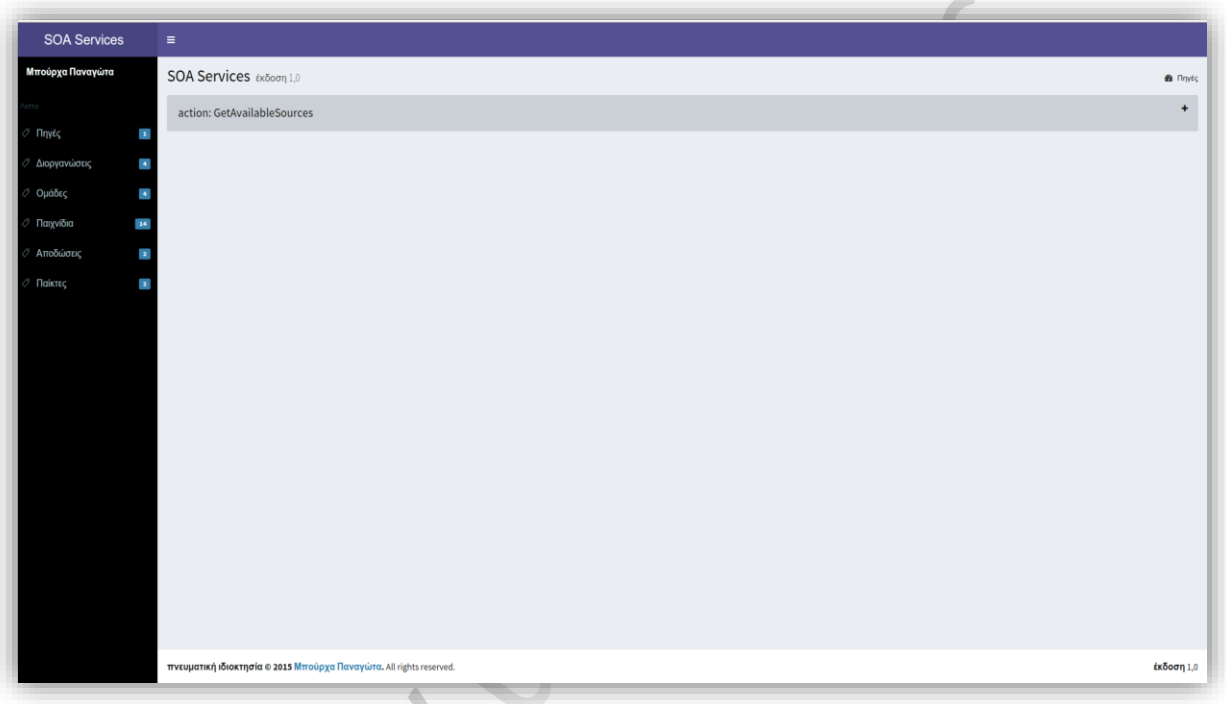
4.2 Web Services

Το επίπεδο των απλών Web Services αποτελείται από web services τα οποία εκτελούν απλές επιχειρηματικές διαδικασίες. Το επίπεδο αυτό είναι αυτό που έρχεται σε αλληλεπίδραση με την βάση δεδομένων.

Στο επίπεδο αυτό έχουν υλοποιηθεί web services που εκτελούν μικρές και απλές μεθόδους, αρκετές από αυτές τις μεθόδους αλληλεπιδρούν με την βάση μας. Στο επίπεδο αυτό οι συγκεκριμένες αυτές μέθοδοι επιτρέπουν σε εξωτερικά συστήματα να τις καλέσουν μέσω τον endpoint και να τις χρησιμοποιήσουν.

Η παρουσίαση όλων των υλοποιημένων web services βρίσκεται στην ηλεκτρονική σελίδα: http://pbourcha.techmind.gr, στην οποία υπάρχουν επίσης πληροφορίες όσο αφορά τις δυνατότητες, τις απαιτήσεις κάθε service, καθώς και οι πληροφορίες που ανταλλάσσονται με την κάθε μια κλήση, παρουσιάζονται παραδείγματα χρήσης αυτών.

Το κάθε service έχει το δικό του endpoint, και όλα μπορούν να καλεστούν μέσω **POST** ή **GET**. Επίσης όλα τα services επιτρέπουν στον χρήστη που τα καλεί να προσδιορίσει την επιστρεφόμενη μορφή δεδομένων που επιθυμεί (**XML** ή **JSON**), μέσω της ιδιότητας - όρισμα «output».



Εικόνα 22 SOA Services Presentation

Τα Web Services που αναπτύχθηκαν είναι τα παρακάτω:

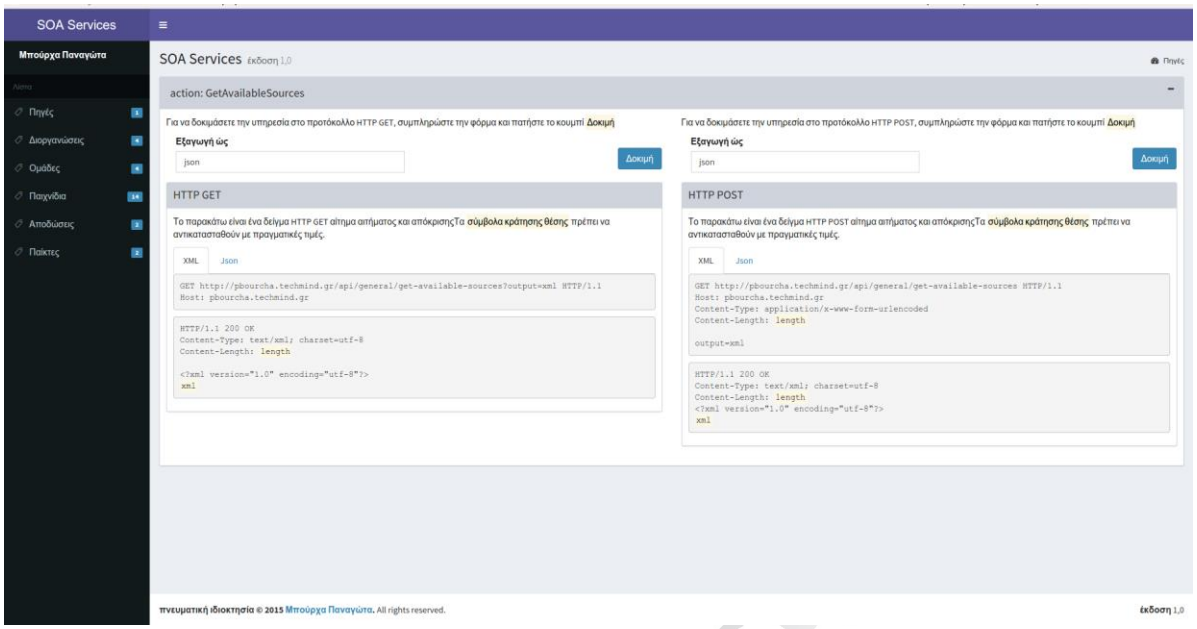
4.2.1 Πηγές

4.2.1.1 GetAvailableSources

Στο web service αυτό ο χρήστης μπορεί να ενημερωθεί για τους παρόχους από τους οποίους λαμβάνει δεδομένα η εφαρμογή μας. Οι παράμετροι αλλά και οι πληροφορίες της φαίνονται στην (Εικόνα 23)

Endpoint: http://pbourcha.techmind.gr/api/general/get-available-sources

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/index



Εικόνα 23 Service GetAvailableSources

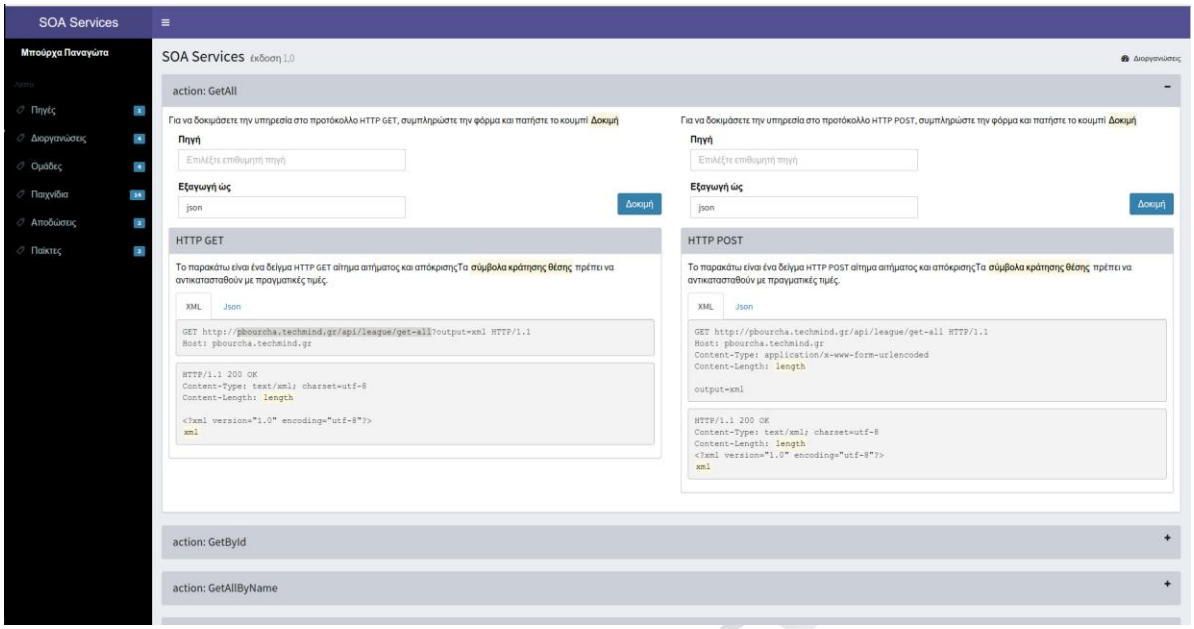
4.2.2 Διοργανώσεις

4.2.2.1 GetAll

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με το ποια πρωταθλήματα υποστηρίζονται από την εφαρμογή. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 24)

Endpoint: http://pbourcha.techmind.gr/api/league/get-all

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/leagues



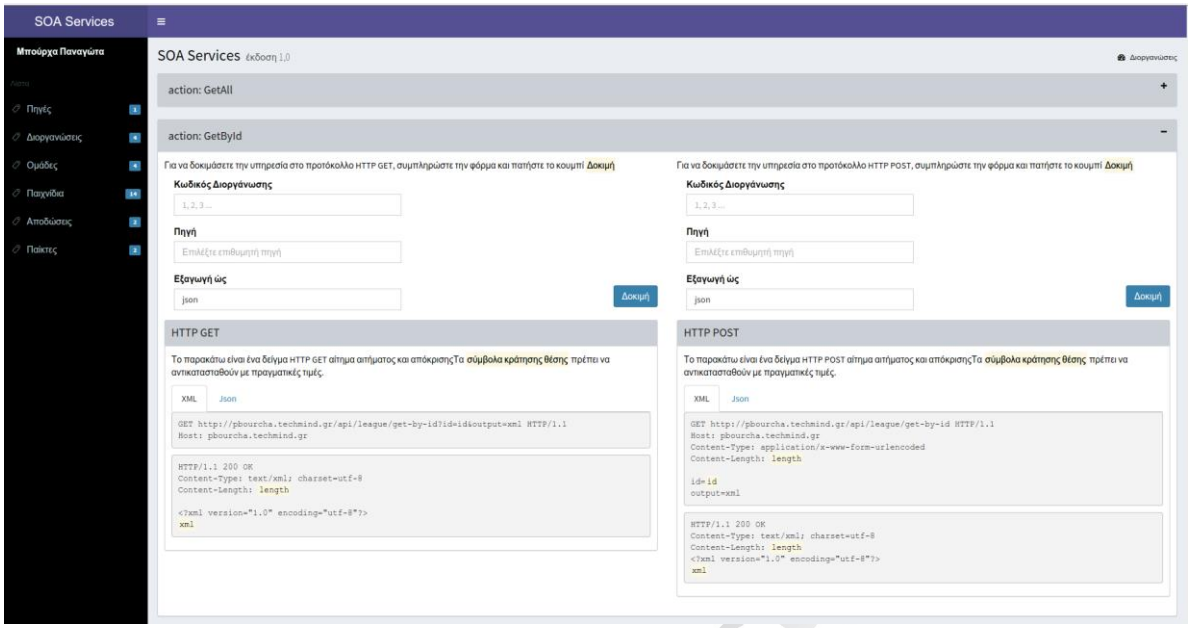
Εικόνα 24 Service Διοργανώσεις GetAll

4.2.2.2 GetById

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με ένα συγκεκριμένο πρωτάθλημα, για να το εκτελέσει αυτό χρειάζεται γνώση του id του πρωταθλήματος, το οποίο είναι εφικτό από το προηγούμενο service GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 25)

Endpoint: http://pbourcha.techmind.gr/api/league/get-by-id

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/leagues



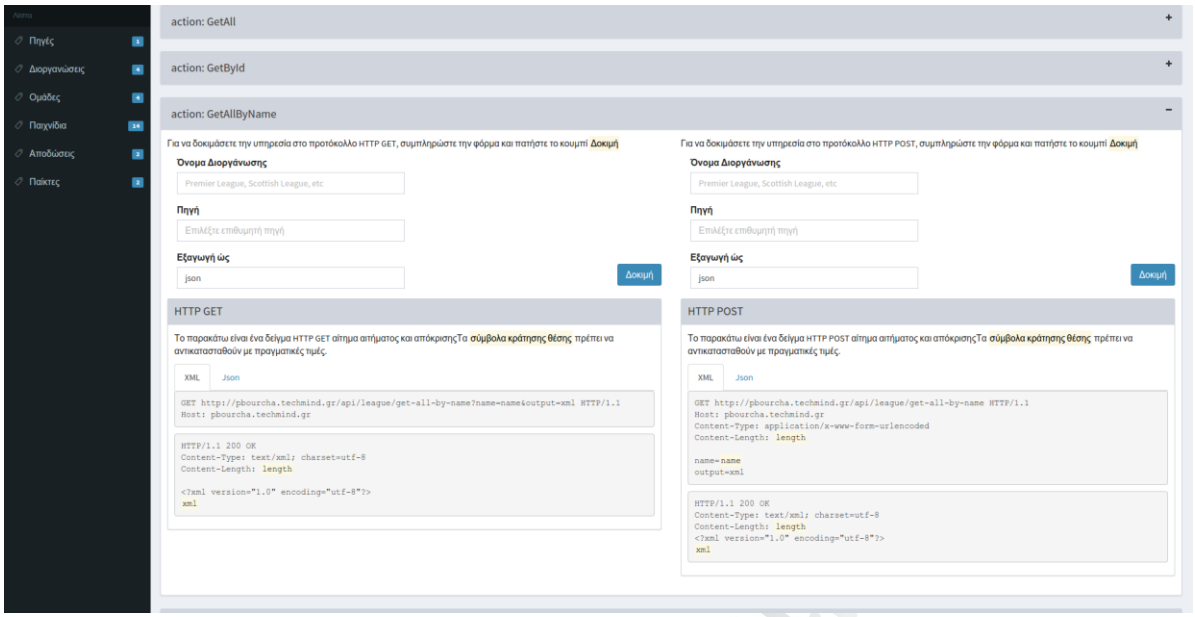
Εικόνα 25 Service Διοργανώσεις GetById

4.2.2.3 GetAllByName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με ένα συγκεκριμένο πρωτάθλημα, για να το εκτελέσει αυτό χρειάζεται γνώση του ονόματος (ολόκληρου ή μέρους) του πρωταθλήματος, το οποίο είναι εφικτό από το προηγούμενο service GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 26)

Endpoint: http://pbourcha.techmind.gr/api/league/get-all-by-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/leagues



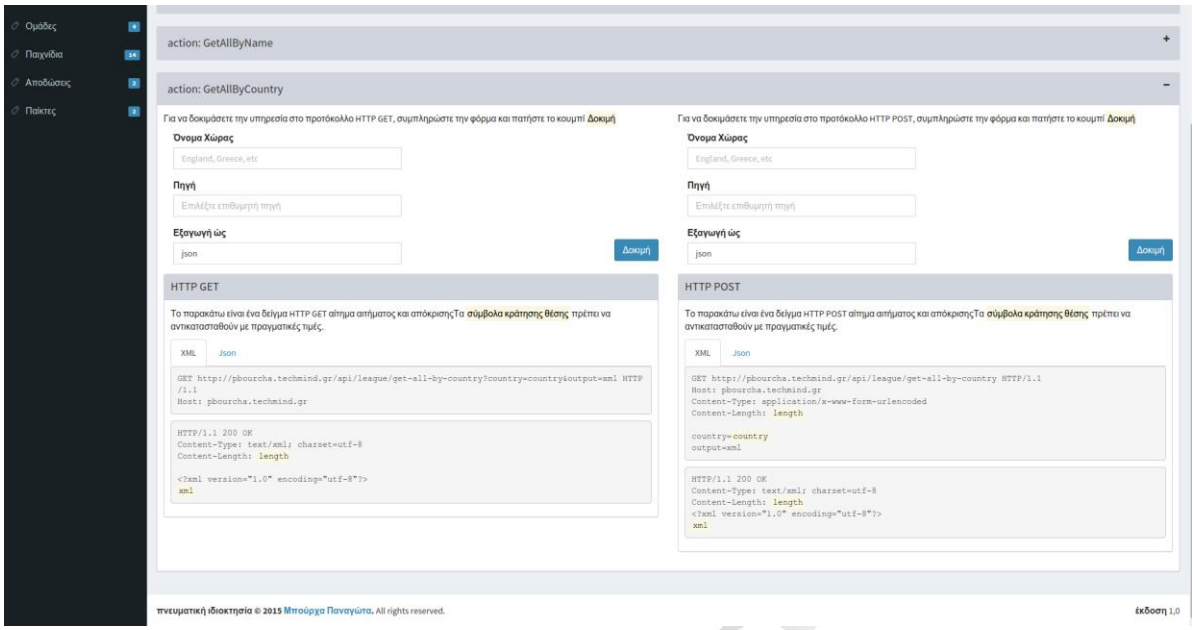
Εικόνα 26 Service Διοργανώσεις GetAllByName

4.2.2.4 GetAllByCountry

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με όλα τα πρωταθλήματα που υποστηρίζονται ανά χώρα, για να το εκτελέσει αυτό χρειάζεται να δώσει το όνομα της χώρας που ενδιαφέρεται, το οποίο μπορεί να το βρει από το προηγούμενο service GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 27)

Endpoint: http://pbourcha.techmind.gr/api/league/get-all-by-country

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/leagues



Εικόνα 27 Service Διοργανώσεις GetAllByCountry

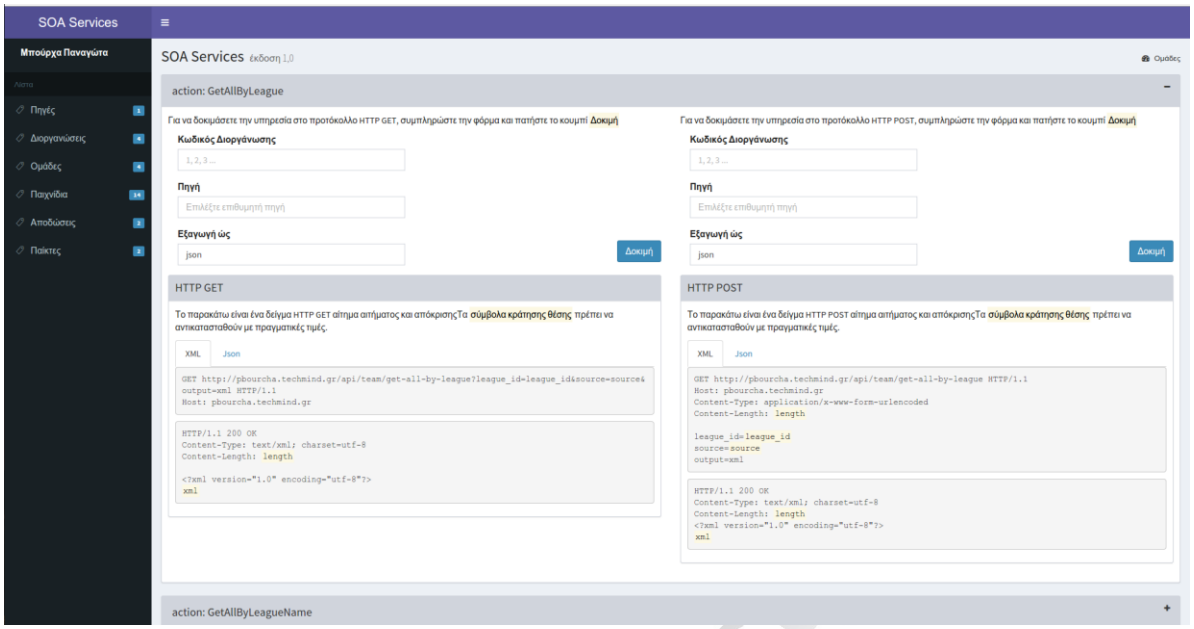
4.2.3 Ομάδες

4.2.3.1 GetAllByLeague

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τις ομάδες ενός πρωταθλήματος, βάση το id του πρωταθλήματος που επιθυμεί, για να μπορέσει να έχει το id, θα πρέπει να έχει καλέσει κάποιο από τα services των διοργανώσεων/πρωταθλημάτων που αναφέραμε πιο πάνω, πχ GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 28)

Endpoint: http://pbourcha.techmind.gr/api/team/get-all-by-league

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/teams



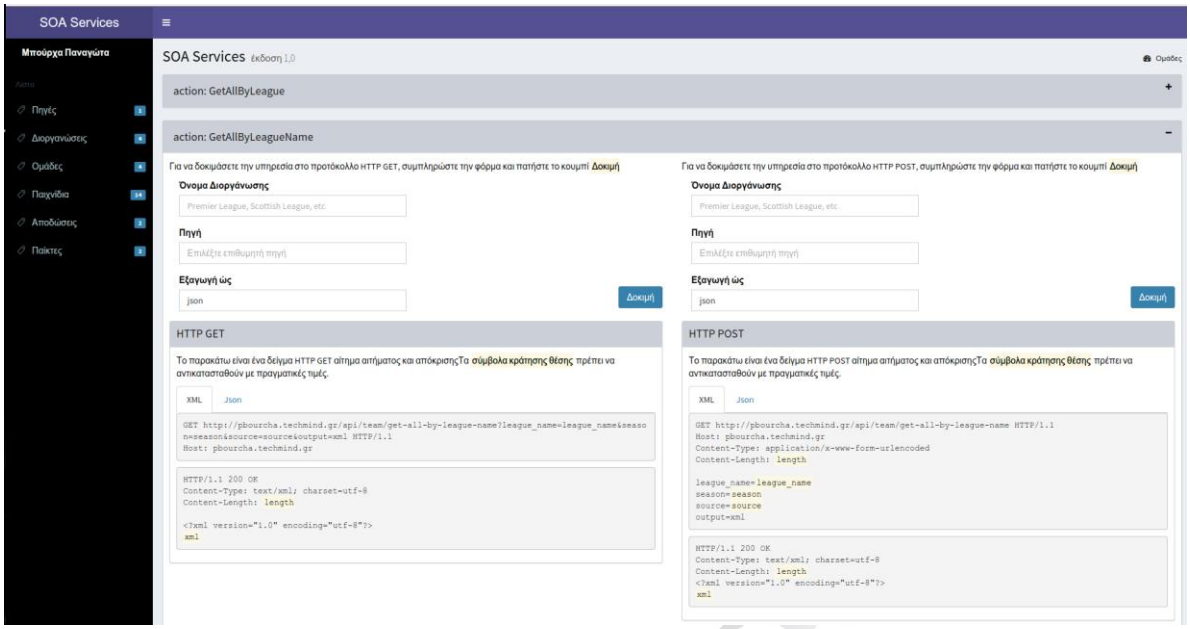
Εικόνα 28 Service Ομάδες GetAllByLeague

4.2.3.2 GetAllByLeagueName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τις ομάδες ενός πρωταθλήματος, βάση ονόματος του πρωταθλήματος που επιθυμεί, για να μπορέσει να έχει το όνομα, θα πρέπει να έχει καλέσει κάποιο από τα services των διοργανώσεων/πρωταθλημάτων που αναφέραμε πιο πάνω, πχ GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 29)

Endpoint: http://pbourcha.techmind.gr/api/team/get-all-by-league-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/teams



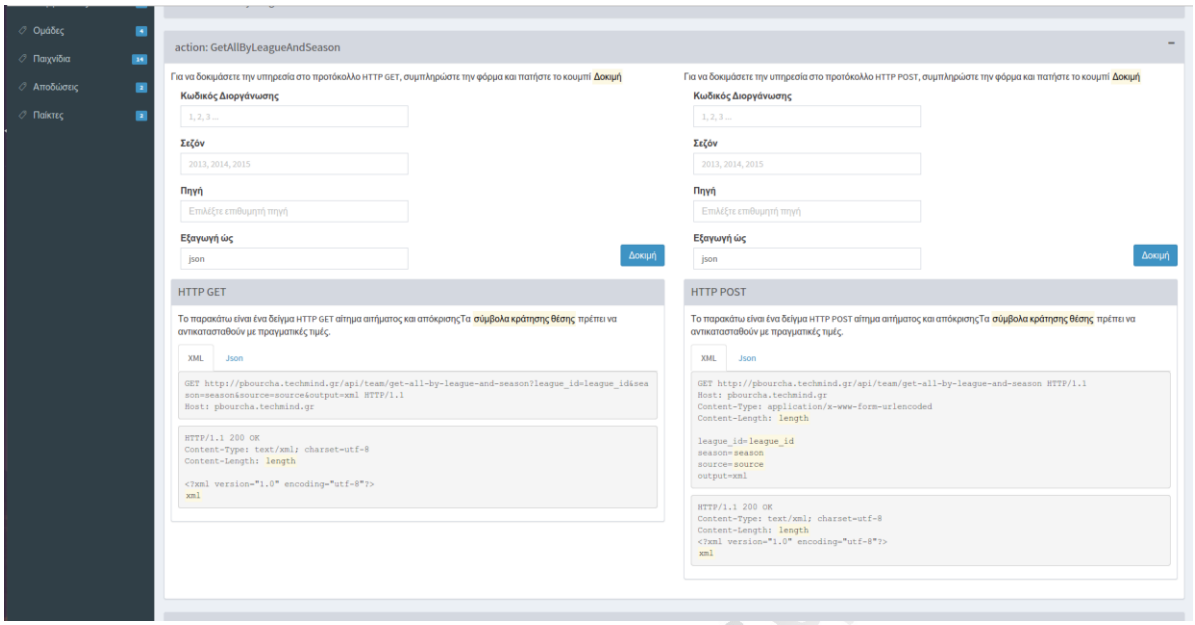
Εικόνα 29 Service Ομάδες GetAllByLeagueName

4.2.3.3 GetAllByLeagueAndSeason

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τις ομάδες ενός πρωταθλήματος, βάση id του πρωταθλήματος που επιθυμεί, καθώς επίσης και την σεζον που ενδιαφέρεται, για να μπορέσει να έχει το id, θα πρέπει να έχει καλέσει κάποιο από τα services των διοργανώσεων/πρωταθλημάτων που αναφέραμε πιο πάνω, πχ GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 30)

Endpoint: http://pbourcha.techmind.gr/api/team/get-all-by-league-and-season

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/teams



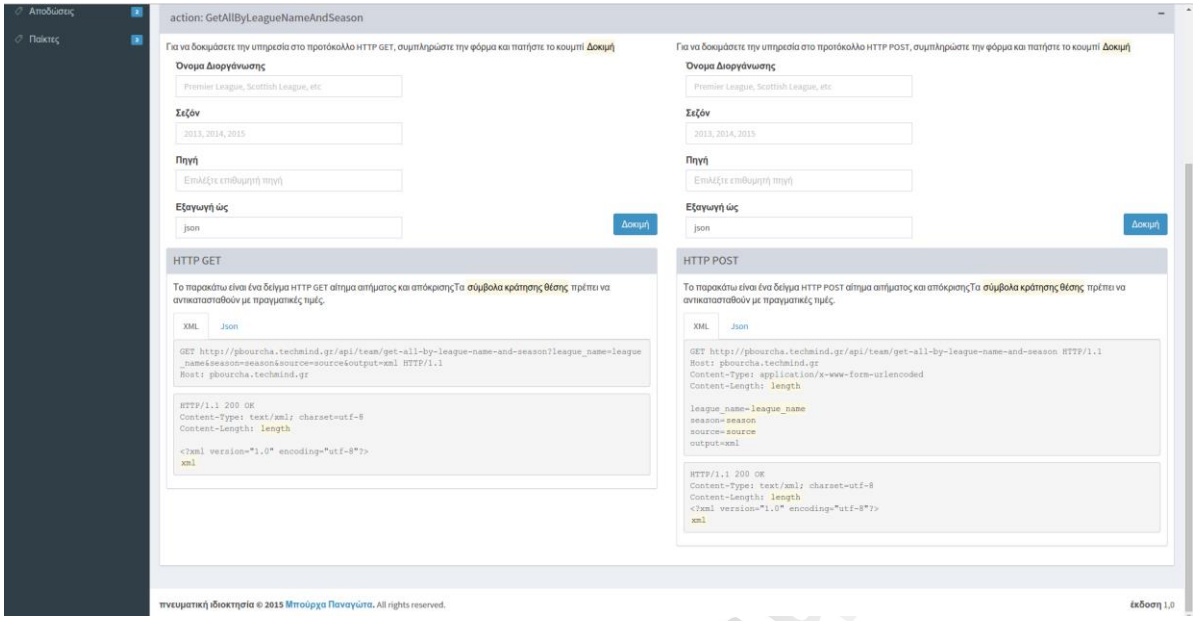
Εικόνα 30 Service Ομάδες GetAllByLeagueAndSeason

4.2.3.4 GetAllByLeagueNameAndSeason

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τις ομάδες ενός πρωταθλήματος, βάση το όνομα του πρωταθλήματος που επιθυμεί, καθώς επίσης και την σεζον που ενδιαφέρεται, για να μπορέσει να έχει το όνομα, θα πρέπει να έχει καλέσει κάποιο από τα services των διοργανώσεων/πρωταθλημάτων που αναφέραμε πιο πάνω, πχ GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 31).

Endpoint: http://pbourcha.techmind.gr/api/team/get-all-by-league-name-and-season

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/teams



Εικόνα 31 Service Ομάδες GetAllByLeagueNameAndSeason

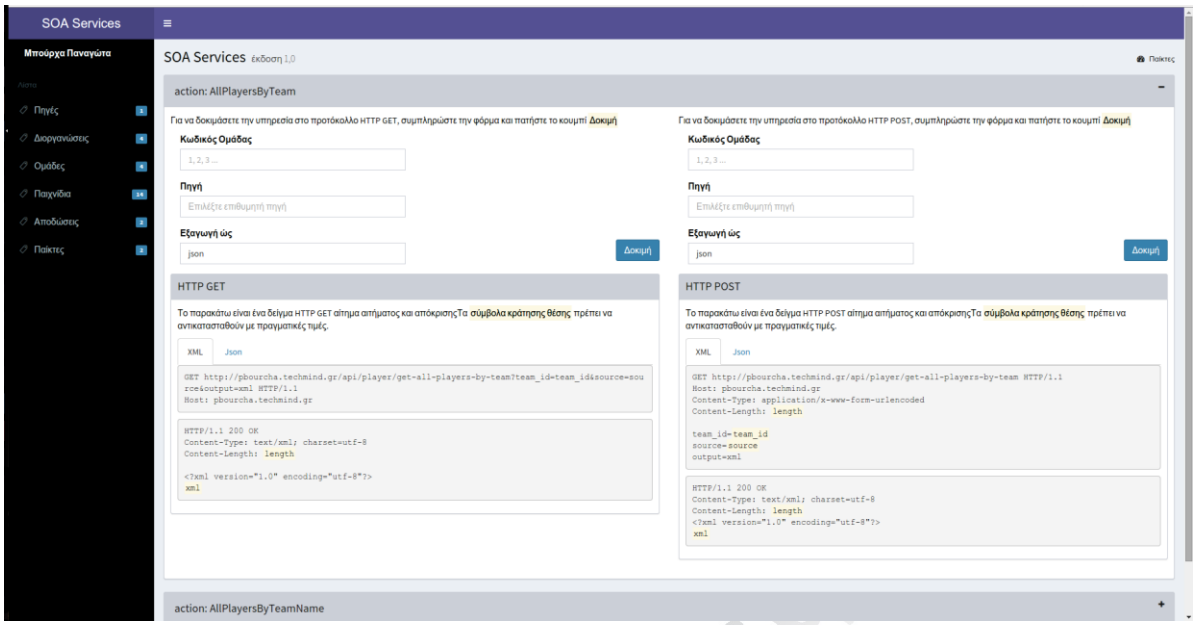
4.2.4 Παίκτες

4.2.4.1 AllPlayersByTeam

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους παίκτες κάποιας ομάδας, βάση του id της ομάδας που επιθυμεί, για να μπορέσει να έχει το id, θα πρέπει να έχει καλέσει κάποιο από τα services των ομάδων που αναφέραμε πιο πάνω, πχ GetAllByLeague. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 32)

Endpoint: http://pbourcha.techmind.gr/api/player/all-players-by-team

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/players



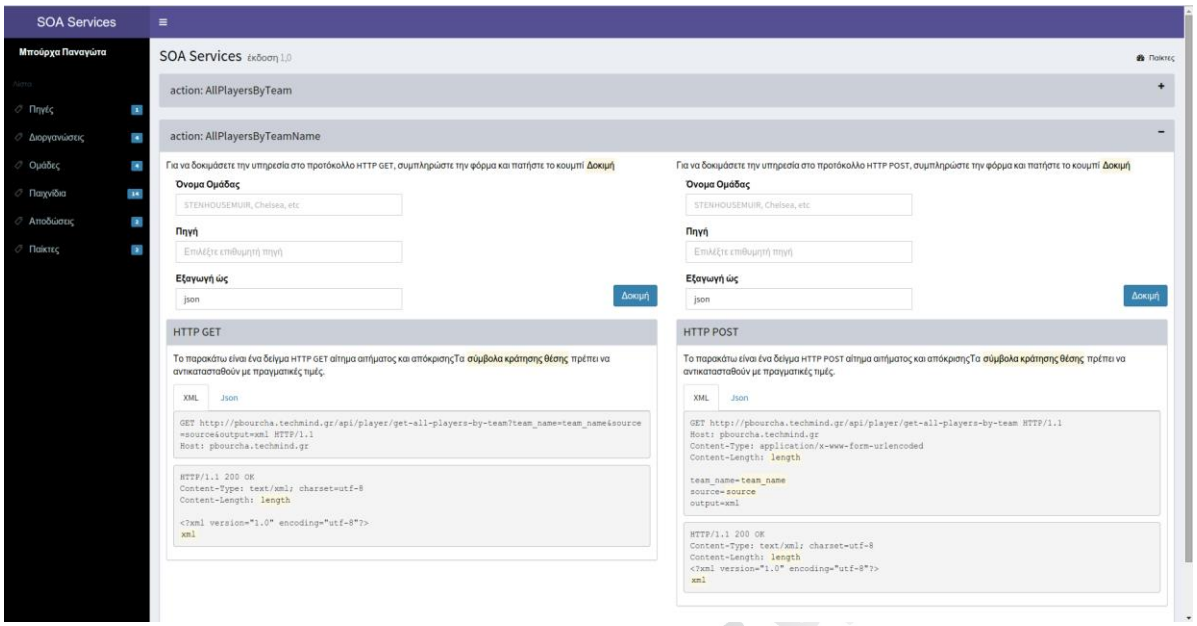
Εικόνα 32 Service Παίκτες AllPlayersByTeam

4.2.4.2 AllPlayersByTeamName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους παίκτες κάποιας ομάδας, βάση του ονόματος της ομάδας που επιθυμεί, για να μπορέσει να έχει το όνομα, θα πρέπει να έχει καλέσει κάποιο από τα services των ομάδων που αναφέραμε πιο πάνω, πχ GetAllByLeague. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 33)

Endpoint: http://pbourcha.techmind.gr/api/player/all-players-by-team-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/players



Εικόνα 33 Service Παίκτες AllPlayersByTeamName

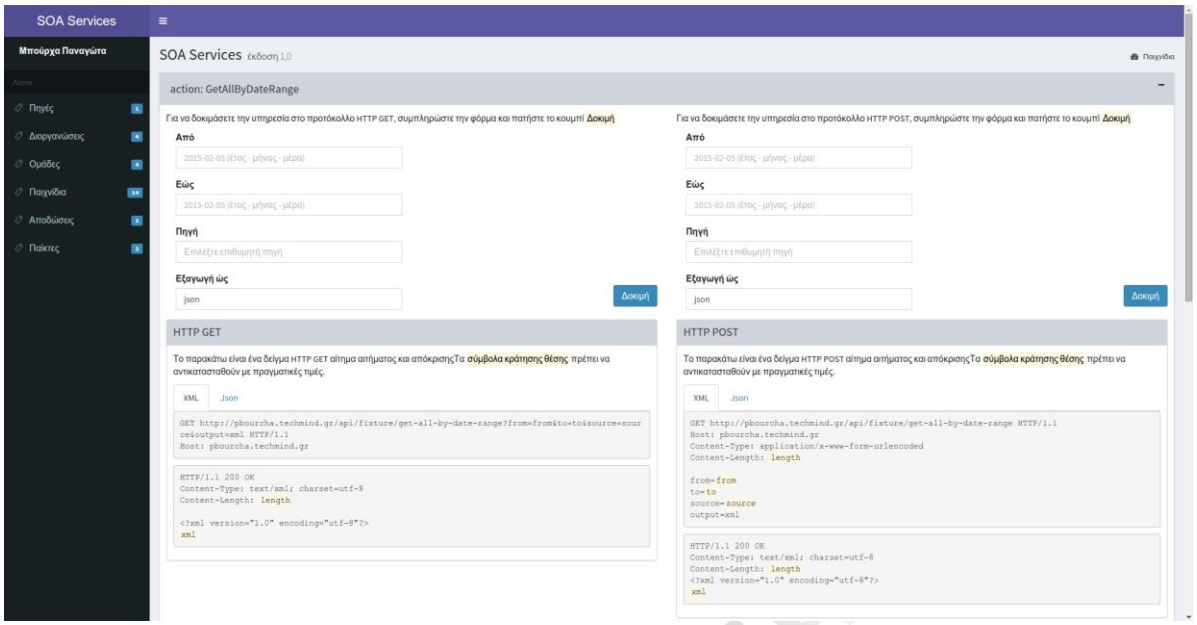
4.2.5 Αγώνες

4.2.5.1 GetAllByDateRange

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με αγώνες σε συγκεκριμένη χρονική περίοδο. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 34).

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-all-by-date-range

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



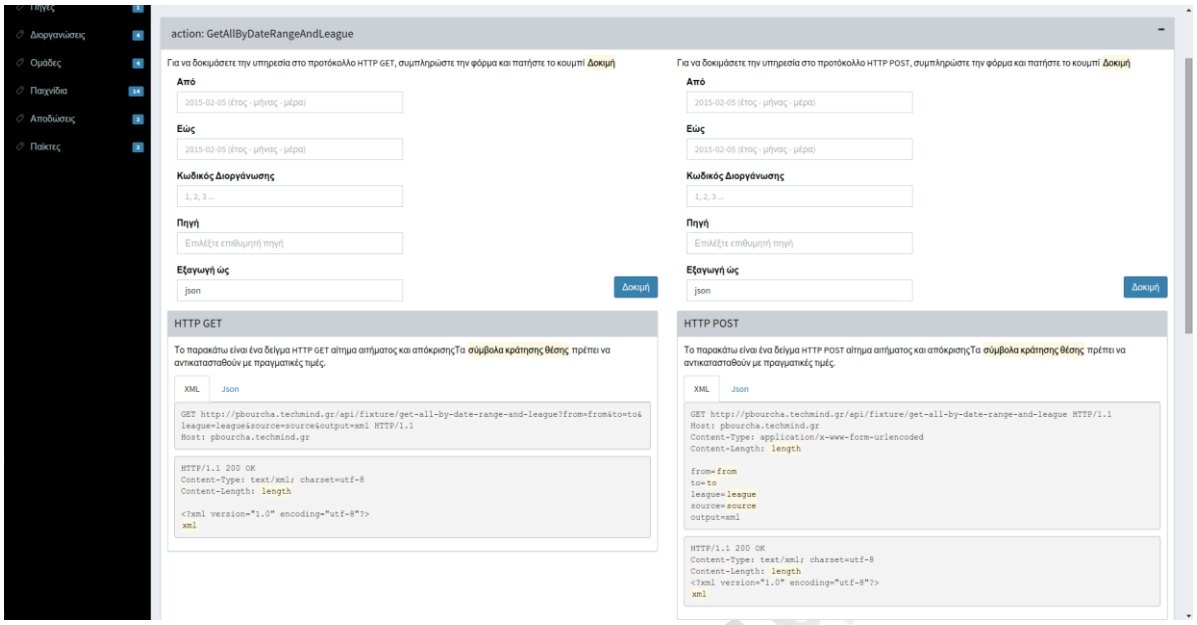
Εικόνα 34 Service Αγώνες GetAllByDateRange

4.2.5.2 GetAllByDateRangeAndLeague

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με αγώνες σε συγκεκριμένη χρονική περίοδο, και από συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του id του πρωταθλήματος, για να μπορεί να γνωρίζει το id θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 35)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-all-by-date-range-and-league

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



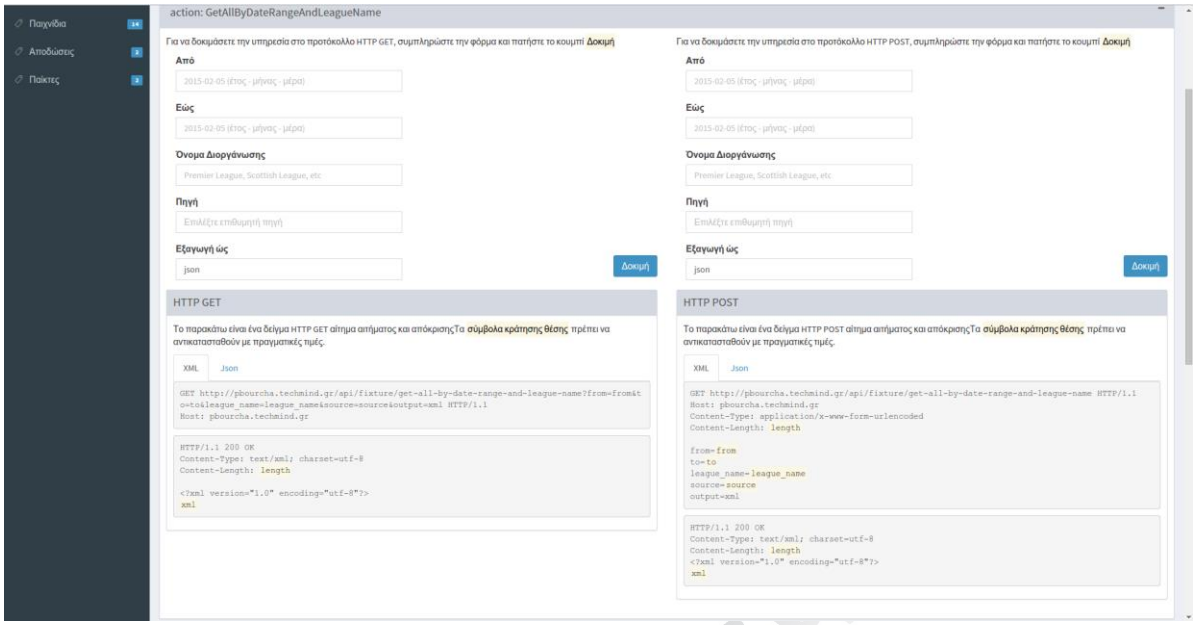
Εικόνα 35 Service Αγώνες GetAllByDateRangeAndLeague

4.2.5.3 GetAllByDateRangeAndLeagueName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με αγώνες σε συγκεκριμένη χρονική περίοδο, και από συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του ονόματος του πρωταθλήματος, για να μπορεί να γνωρίζει το όνομα θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 36)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-all-by-date-range-and-league-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



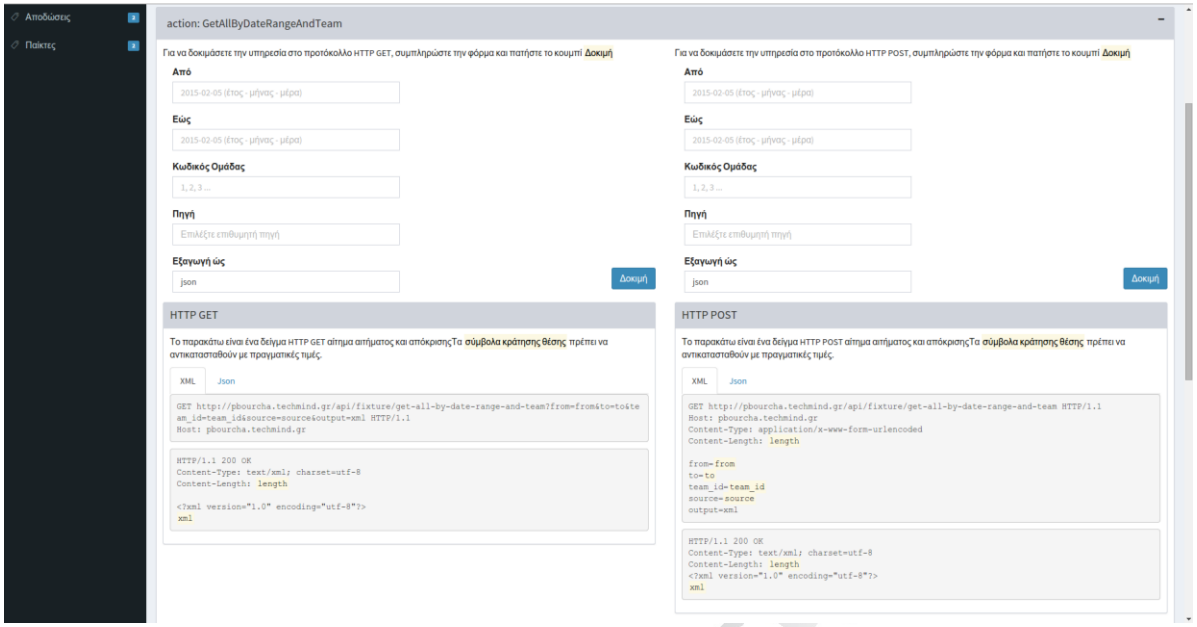
Εικόνα 36 Service Αγώνες GetAllByDateRangeAndLeagueName

4.2.5.4 GetAllByDateRangeAndTeam

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με αγώνες σε συγκεκριμένη χρονική περίοδο, και για συγκεκριμένη ομάδα, κάνοντας χρήση του id της ομάδας, για να μπορεί να γνωρίζει το id θα πρέπει να έχει καλέσει κάποιο από τα services των ομάδων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAllByLeague. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 37)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-all-by-date-range-and-team

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



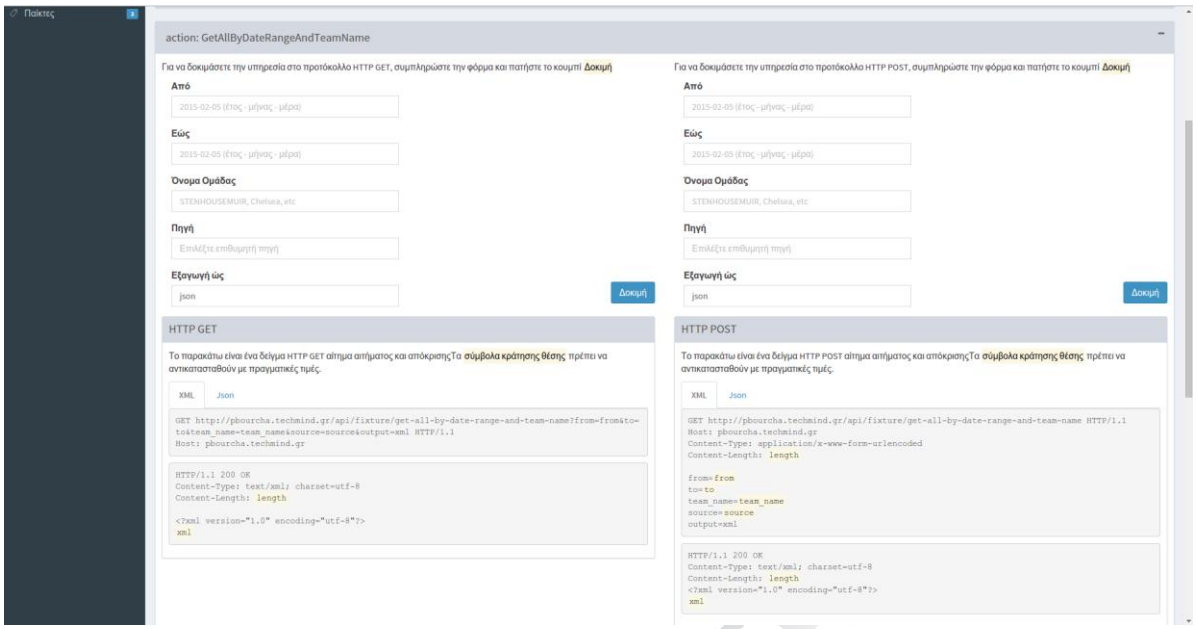
Εικόνα 37 Service Αγώνες GetAllByDateRangeAndTeam

4.2.5.5 GetAllByDateRangeAndTeamName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με αγώνες σε συγκεκριμένη χρονική περίοδο, και για συγκεκριμένη ομάδα, κάνοντας χρήση του ονόματος της ομάδας, για να μπορεί να γνωρίζει το όνομα θα πρέπει να έχει καλέσει κάποιο από τα services των ομάδων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAllByLeague. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 38)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-all-by-date-range-and-team-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



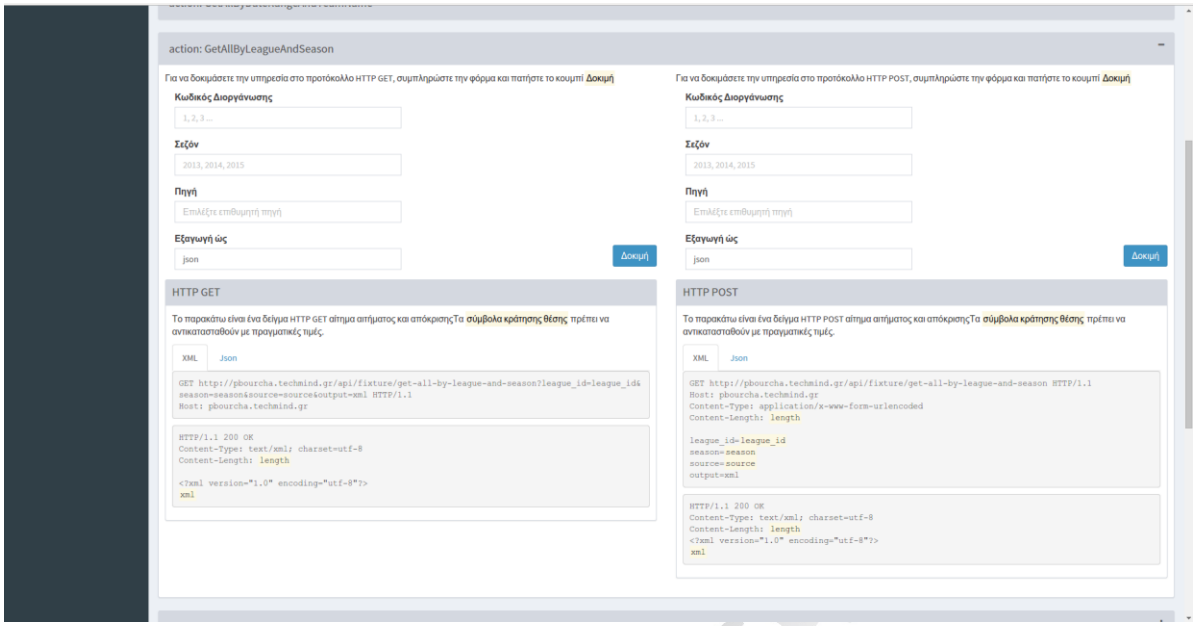
Εικόνα 38 Service Αγώνες GetAllByDateRangeAndTeamName

4.2.5.6 GetAllByLeagueAndSeason

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με αγώνες σε συγκεκριμένη αγωνιστική περίοδο, και από συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του id του πρωταθλήματος, για να μπορεί να γνωρίζει το id θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 39)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-all-by-league-and-season

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



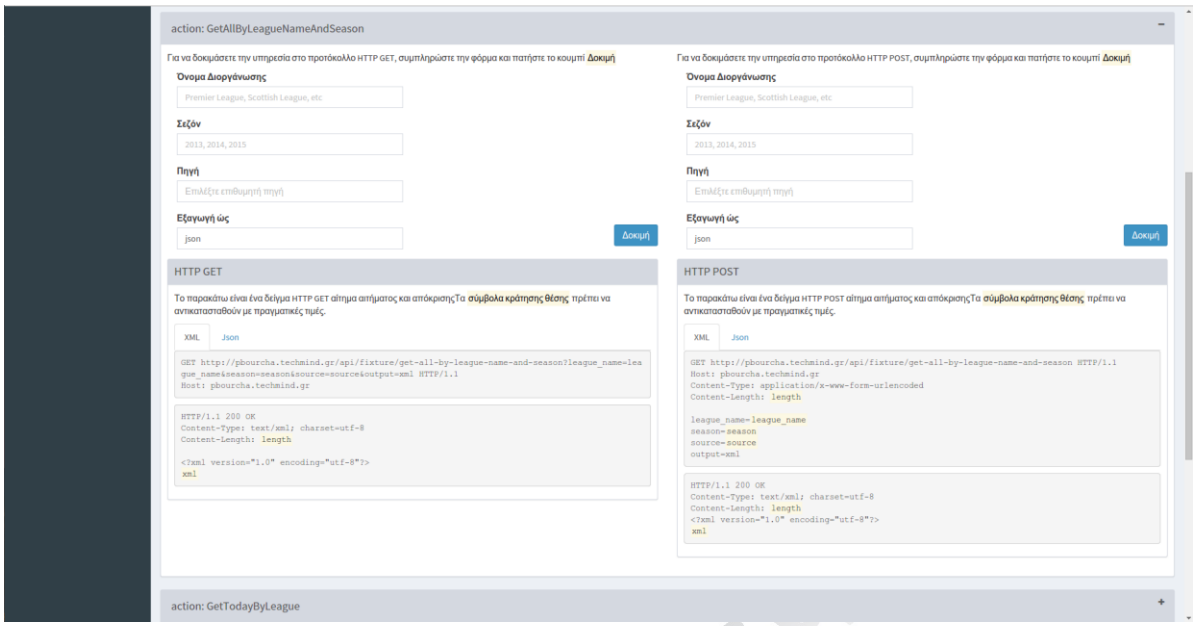
Εικόνα 39 Service Αγώνες GetAllByLeagueAndSeason

4.2.5.7 GetAllByLeagueNameAndSeason

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με αγώνες σε συγκεκριμένη αγωνιστική περίοδο, και από συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του ονόματος του πρωταθλήματος, για να μπορεί να γνωρίζει το όνομα θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 40)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-all-by-league-name-and-season

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



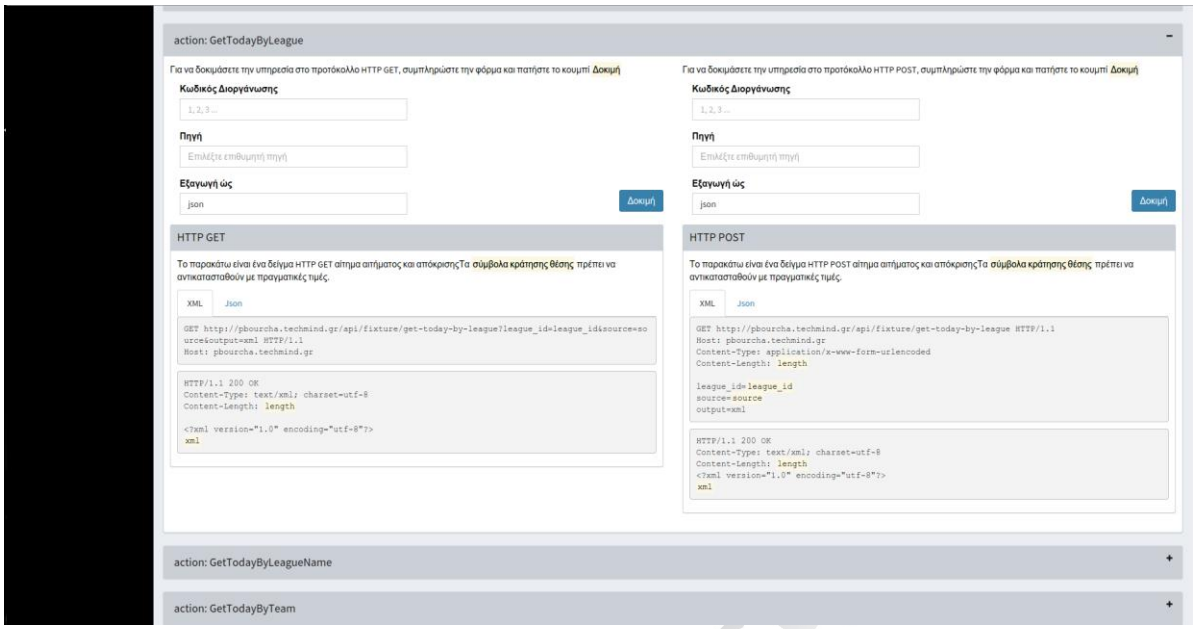
Εικόνα 40 Service Αγώνες GetAllByLeagueNameAndSeason

4.2.5.8 GetTodayByLeague

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους σημερινούς αγώνες από συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του id του πρωταθλήματος, για να μπορεί να γνωρίζει το id θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 41)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-today-by-league

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



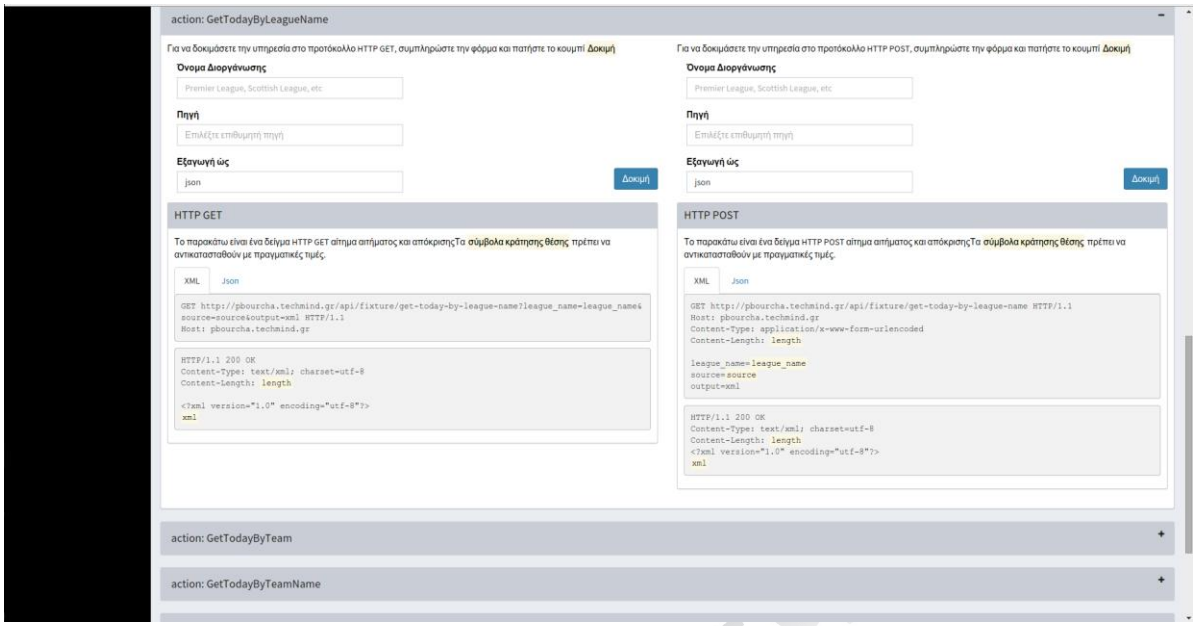
Εικόνα 41 Service Αγώνες GetTodayByLeague

4.2.5.9 GetTodayByLeagueName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους σημερινούς αγώνες από συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του ονόματος του πρωταθλήματος, για να μπορεί να γνωρίζει το όνομα θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 42)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-today-by-league-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



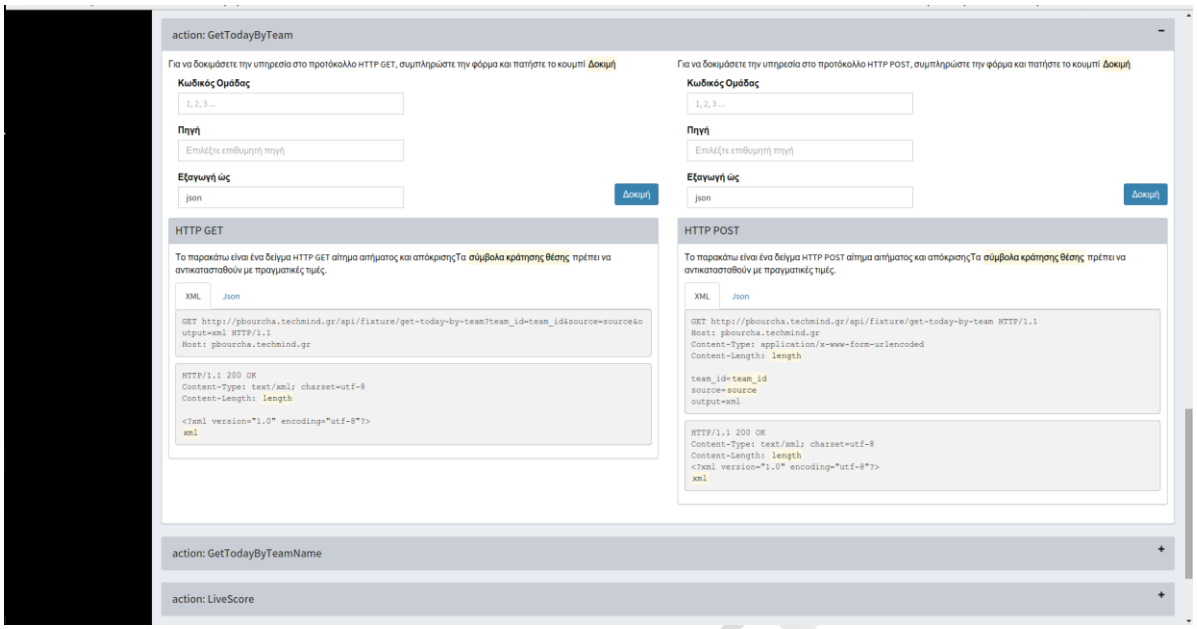
Εικόνα 42 Service Αγώνες GetTodayByLeagueName

4.2.5.10 GetTodayByTeam

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους σημερινούς αγώνες για συγκεκριμένη ομάδα, κάνοντας χρήση του id της ομάδας, για να μπορεί να γνωρίζει το id θα πρέπει να έχει καλέσει κάποιο από τα services των ομάδων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAllByLeague. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 43)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-today-by-team

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



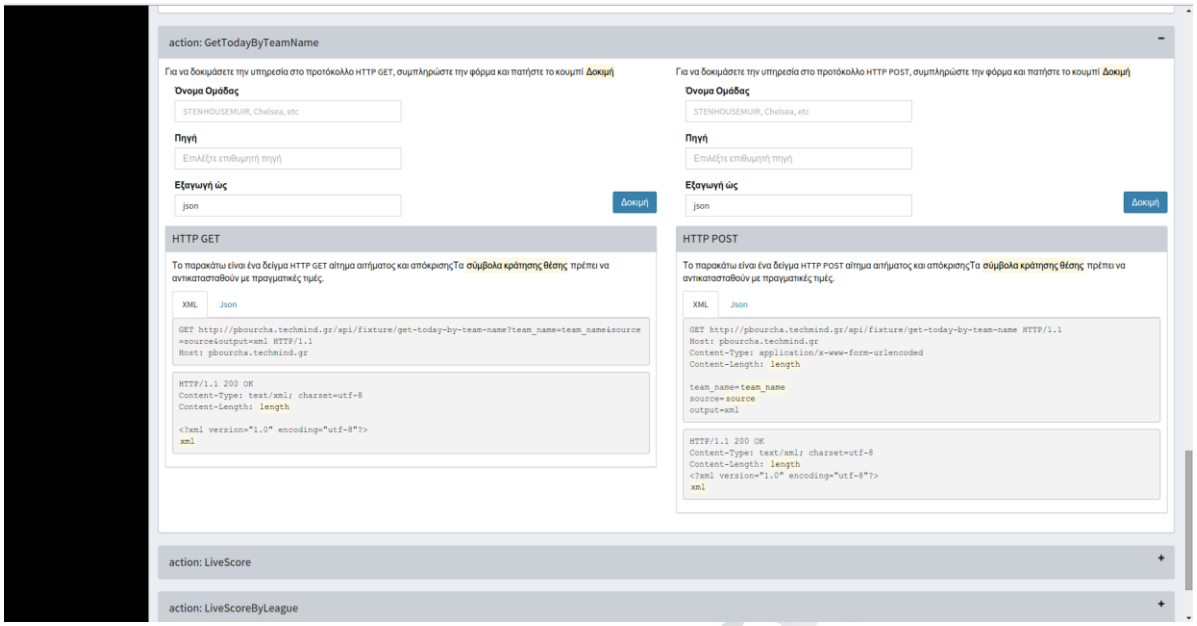
Εικόνα 43 Service Αγώνες GetTodayByTeam

4.2.5.11 GetTodayByTeamName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους σημερινούς αγώνες για συγκεκριμένη ομάδα, κάνοντας χρήση του ονόματος της ομάδας, για να μπορεί να γνωρίζει το όνομα θα πρέπει να έχει καλέσει κάποιο από τα services των ομάδων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAllByLeague. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 44)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-today-by-team-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



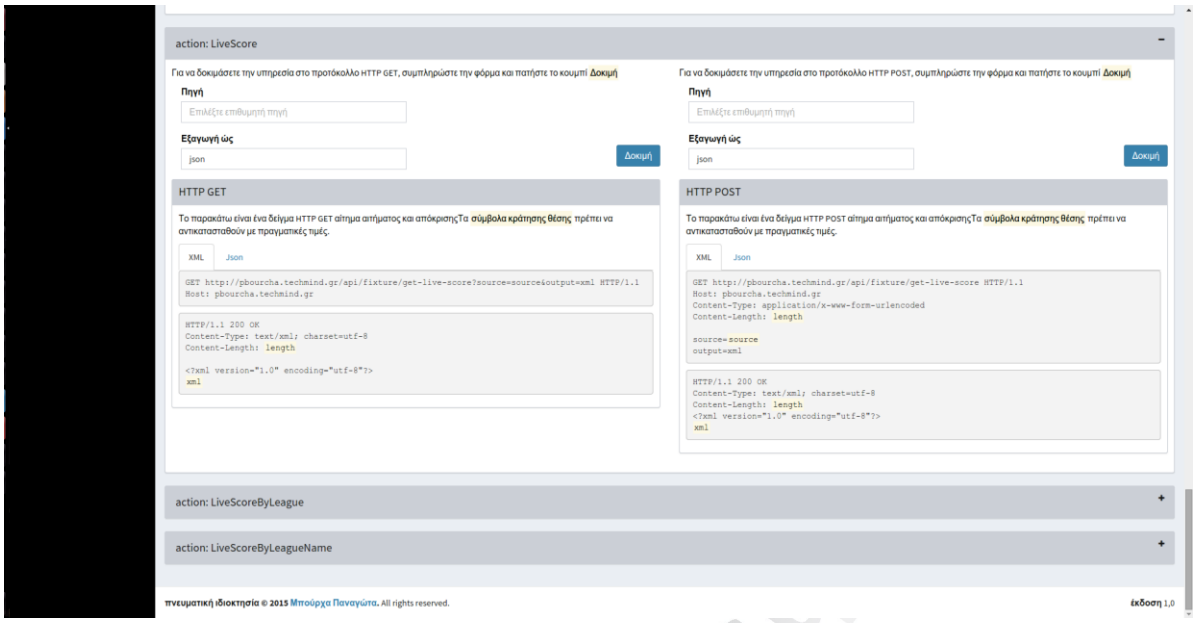
Εικόνα 44 Service Αγώνες GetTodayByTeamName

4.2.5.12 LiveScore

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους αγώνες σε εξέλιξη που έχει κάθε πάροχος. Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 45)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-live-score

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



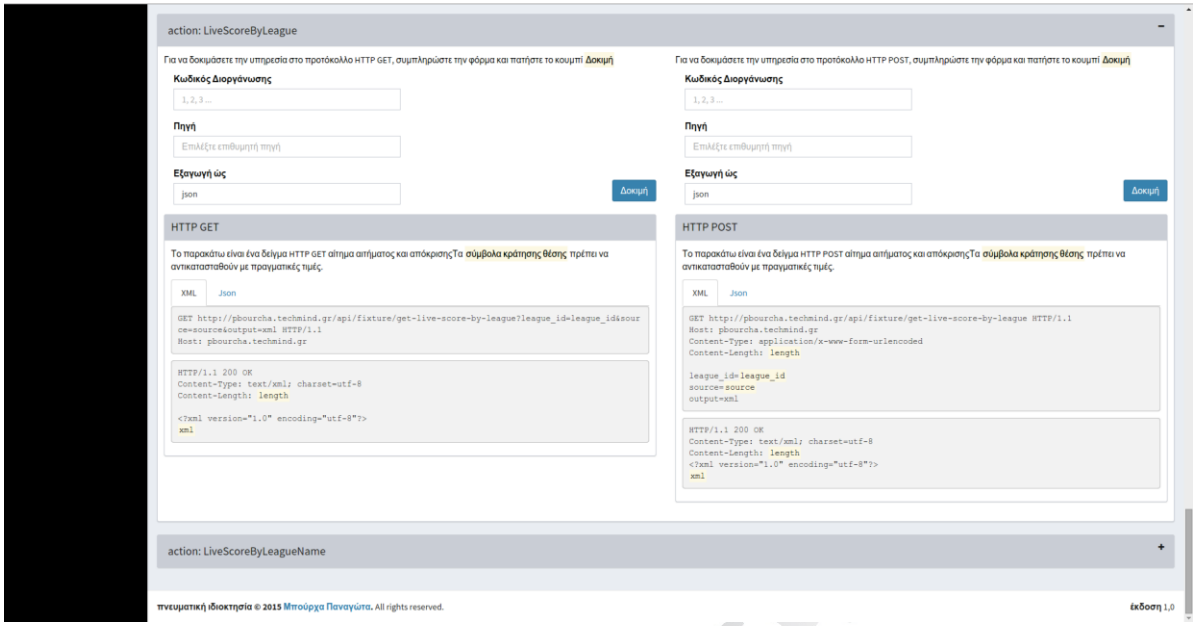
Εικόνα 45 Service Αγώνες LiveScore

4.2.5.13 LiveScoreByLeague

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους αγώνες σε εξέλιξη που έχει κάθε πάροχος και για συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του id του πρωταθλήματος, για να μπορεί να γνωρίζει το id θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 46)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-live-score-by-league

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



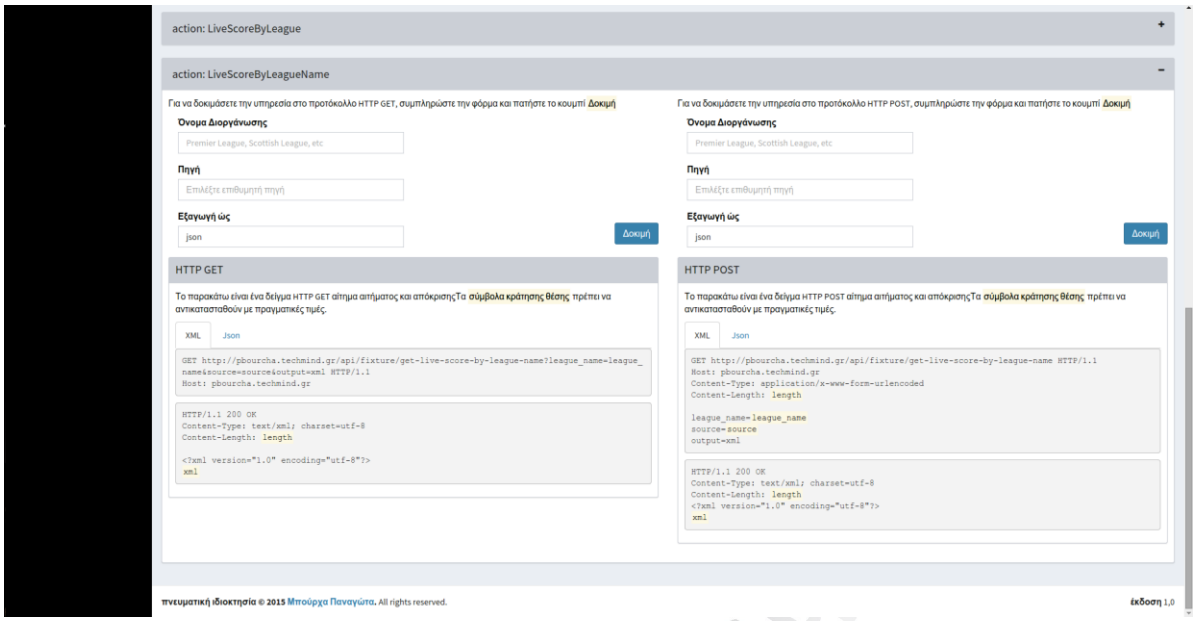
Εικόνα 46 Service Αγώνες LiveScoreByLeague

4.2.5.14 LiveScoreByLeagueName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τους αγώνες σε εξέλιξη που έχει κάθε πάροχος και για συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του ονόματος του πρωταθλήματος, για να μπορεί να γνωρίζει το όνομα θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 47)

Endpoint: http://pbourcha.techmind.gr/api/fixture/get-live-score-by-league-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/fixtures



Εικόνα 47 Service Αγώνες LiveScoreByLeagueName

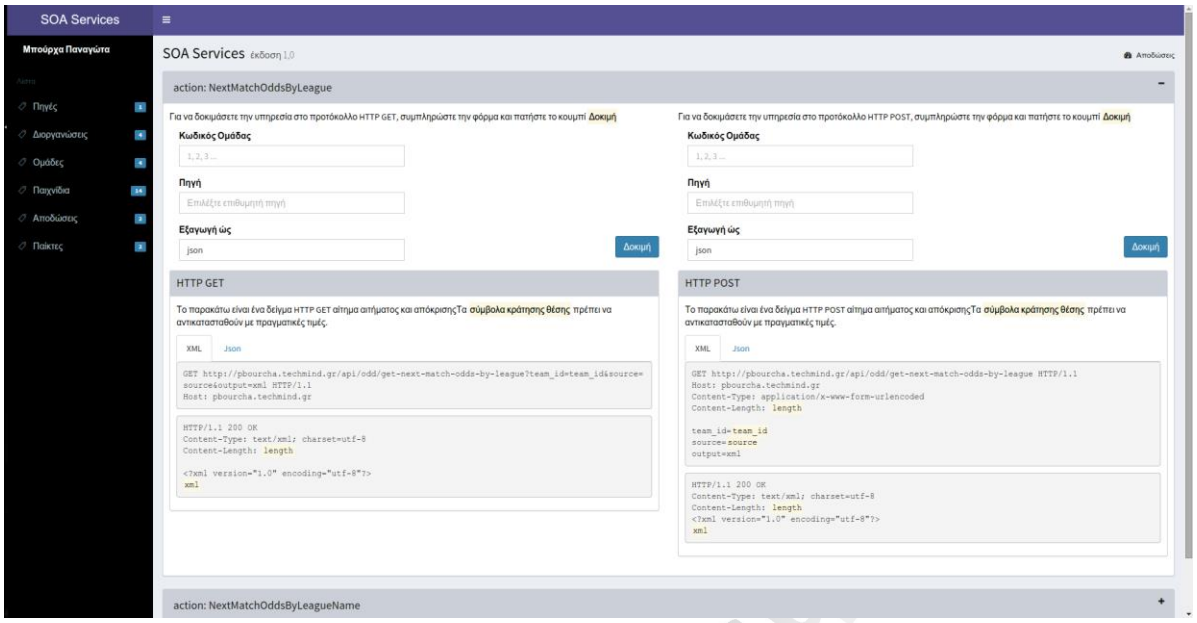
4.2.6 Αποδώσεις

4.2.6.1 NextMatchOddsByLeague

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τις αποδώσεις που υπάρχουν για συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του id του πρωταθλήματος, για να μπορεί να γνωρίζει το id του πρωταθλήματος, θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 48)

Endpoint: http://pbourcha.techmind.gr/api/odd/get-next-match-odds-by-league

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/odds



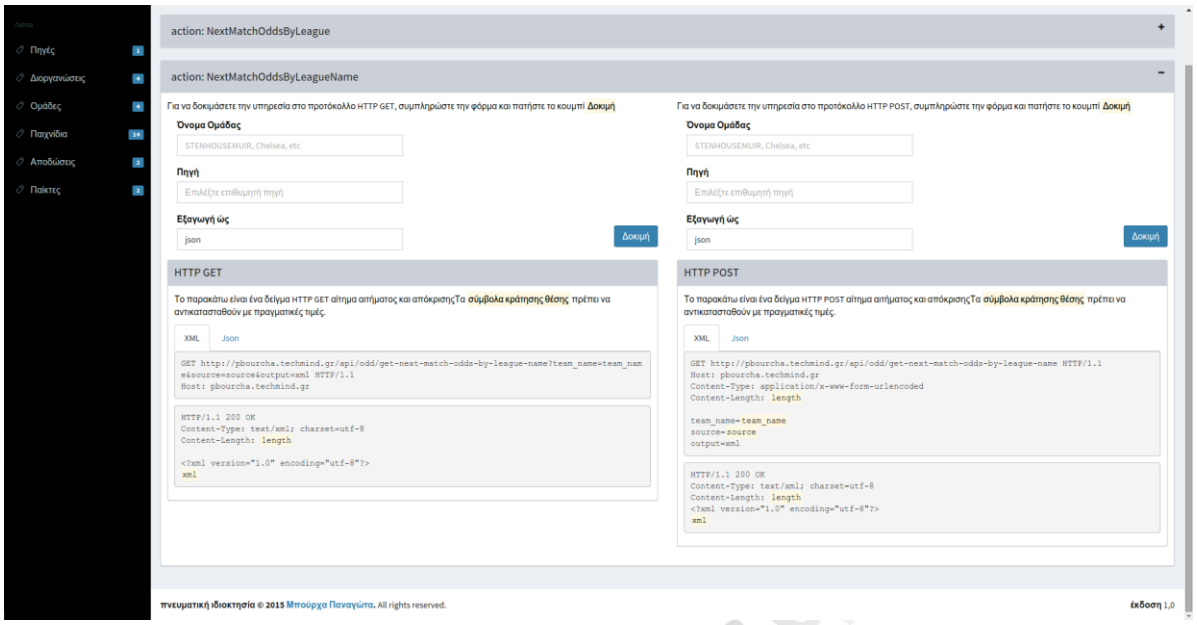
Εικόνα 48 Service Αποδώσεις NextMatchOddsByLeague

4.2.6.2 NextMatchOddsByLeagueName

Το web service αυτό το χρησιμοποιεί ο χρήστης όταν θέλει να πάρει πληροφορίες σχετικές με τις αποδώσεις που υπάρχουν για συγκεκριμένο πρωτάθλημα, κάνοντας χρήση του ονόματος του πρωταθλήματος, για να μπορεί να γνωρίζει το όνομα του πρωταθλήματος, θα πρέπει να έχει καλέσει κάποιο από τα services των πρωταθλημάτων που αναφέραμε πιο πάνω, όπως για παράδειγμα GetAll Οι παράμετροι αλλά και οι πληροφορίες φαίνονται στην (Εικόνα 49)

Endpoint: http://pbourcha.techmind.gr/api/odd/get-next-match-odds-by-league-name

Παρουσίαση: http://pbourcha.techmind.gr/api-preview/odds



Εικόνα 49 Service Αποδώσεις NextMatchOddsByLeagueName

5 Συμπεράσματα και μελλοντική ερευνά

Σε αυτό το τελευταίο κεφάλαιο θα αναφερθούμε στα συμπεράσματα που εξήχθησαν κατόπιν της βιβλιογραφικής ανασκόπησης του θέματος και της εμπειρίας που προέκυψε από την εφαρμογή της θεωρίας στην ανάπτυξη μιας ολοκληρωμένης εφαρμογής.

5.1 Συμπεράσματα

Σε αυτή την διπλωματική εργασία εξετάστηκε εκτενώς η υπηρεσιοστρεφής αρχιτεκτονική από όλες τις σκοπιές. Η ανάλυση της βιβλιογραφίας αποδεικνύει ότι στις μέρες μας η υπηρεσιοστρεφής αρχιτεκτονική αποτελεί την πιο ενδεδειγμένη λύση για την επιχειρηματική ολοκλήρωση καθώς παρέχει μια κοινή πλατφόρμα και ένα κοινό περιβάλλον εργασίας μεταξύ ετερογενών εφαρμογών που αναπτύχθηκαν με διαφορετικές τεχνολογίες από διάφορους οργανισμού. Η υπηρεσιοστρεφής αρχιτεκτονική ενισχύει την διαλειτουργικότητα μεταξύ διαφορετικών αρχιτεκτονικών αποκρύπτοντας την εσωτερική τους δομή με αποτέλεσμα την δημιουργία ευέλικτων και χαλαρά διασυνδεδεμένων κατανεμημένων συστημάτων.

Αν και, όπως τονίστηκε στο κεφάλαιο δύο, η υπηρεσιοστρεφής αρχιτεκτονική είναι ανεξάρτητη τεχνολογιών τα Web Services συνιστούν τον καλύτερο τρόπο υλοποίησης της . Το γεγονός αυτό οφείλεται στο γεγονός ότι στηρίζονται σε ευρέως αποδεκτά και χρησιμοποιούμενα πρότυπα όπως είναι το πρωτόκολλο SOAP, η WSDL διεπαφή που βασίζεται στην XML, η ιδέα του UDDI καταλόγου που διευκολύνει την ανακάλυψη και αξιοποίηση των υπηρεσιών αυτών.

Στην διπλωματική αυτή υλοποιήθηκε μια εφαρμογή η οποία καταδεικνύει τα θεμελιώδη χαρακτηριστικά των υπηρεσιοστρεφών εφαρμογών. Αποδεικνύεται η διαλειτουργικότητα των διαφορετικών πλατφόρμων, την επαναχρησιμοποίηση των χαλαρά συνδεδεμένων υπηρεσιών, και την κατανάλωση των υπηρεσιών από άλλες ενότητες του λογισμικού για να σχηματίσουν μια υπηρεσία που βασίζεται στην αλληλεπίδραση μεταξύ διαφορετικών εφαρμογών.

5.2 Μελλοντική εργασία

Η εφαρμογή που αναπτύχθηκε στα πλαίσια αυτής της διπλωματικής εργασίας μπορεί να αποτελέσει την αφετηρία για μια πολύ μεγαλύτερη, πολυπλοκότερη εφαρμογή. Την στιγμή αυτή η εφαρμογή συλλέγει ιστορικά δεδομένα για ομάδες και παίκτες από δύο πηγές και της αναμεταδίδει.

Έχει προβλεφθεί ήδη μελλοντικά οι πηγές άντλησής πληροφορίας να είναι περισσότερες, ώστε να εξάγεται ακόμα μεγαλύτερη ποικιλία πληροφοριών με σκοπό να αυξηθεί η αξιοπιστία της τελικής πληροφορίας εφόσον θα ελέγχεται σε σχέση με πολλές πηγές. Αυτή η ποικιλία αξιόπιστης πληροφορίας μπορεί να αξιοποιηθεί ώστε να παρέχονται στους καταναλωτές (consumers) στατιστικά τόσο των ομάδων όσο και παικτών, ενώ μπορεί μελλοντικά να χρησιμοποιηθεί ακόμα και από headhunter ή ιδιοκτήτες ομάδων με σκοπό να παρακολουθούν τα πρωταθλήματα και τους παίκτες σε παγκόσμια κλίμακα και να επιλέγουν νέους παίκτες για την ομάδα τους, να μελετούν τους αντιπάλους τους κ.τ.λ.

6 Βιβλιογραφία

- Θεμιστοκλέους Μ. και Μαντζάνα Β (2010), *Υπηρεσίες Παγκόσμιου Ιστού και Υπηρεσιοστρεφείς Αρχιτεκτονικές*
- Επαμεινώνδας, Ν (2012), *Ανάπτυξη εφαρμογής με την χρήση BPEL και Web Services*
- Τσούτσα. Π., (2010), *Αρχιτεκτονική SOA και Υπηρεσίες Ιστου.*
- Δάρα, Α. (2006), *Προσαρμοζόμενη αναζήτηση Διαδικτυακών Υπηρεσιών (Web Services) με υποστήριξη χαρακτηριστικών ποιότητας υπηρεσίας*
- Κατσιρντάκη Μ.(2012), *Ολοκλήρωση Ηλεκτρονικής Τραπαζικής*
- Λαμπαθάκη Φ. (2005), *Ολοκλήρωση Συστημάτων και Εφαρμογών με χρήση Web Services «Διαλειτουργικότητα Συστημάτω νμεταξύ δημόσιων φορέων»*
- Baker S. και Dobson S. (2005), Comparing Service-Oriented and Distributed Object Architectures
- Carter, S. (2007). The New Language of Business SOA & Web 2.0. IBM press.
- Newcomer, E., & Lomow, G. (2004). Understanding SOA with Web Services. Addison Wesley Professional.
- McGovern, J., Tyagi, S., Stevens E, M., & Mathew, S. (2003). Java Web Services Architecture . Morgan Kaufmann.
- Cerami, E. (2002). Web Services Essentials Distributed Applications with XML-RPC, SOAP, UDDI & WSDL. O'Reilly.
- CHAPPELL, D. A. (2002). Java Web Services. O'Reilly & Associates.
- Weerawarana, s. (2005). Web Services Platform Architecture. Pearson education.
- Channabasavaiah, K., Holley, K., & Tuggle , E. M. (2004). Migrating to a service oriented architecture. IBM.
- Endrei, M., Ang, J., Arsanjani, A., Chua, S., Comte, P., Krogdahl, l., και συν. (2004, April). Patterns: ServiceOriented Architecture and Web Services. Ανάκτηση Μάιος 23, 2011, από: http://www.redbooks.ibm.com/redbooks/pdfs/sg246303.pdf

- OASIS (2006), *Reference Model for Service Oriented Architecture 1.0* από: https://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf
- IBM open newsletter (2006), *Αφιέρωμα IBM Service Oriented Architecture* από: http://www-05.ibm.com/gr/pdf/OPEN_MARCH_06.pdf
- Ηλιούδης Χ, *Υπηρεσιοστρεφής Αρχιτεκτονική –SOA (Service Oriented Architecture)* από: http://aetos.it.teithe.gr/~iliou/cs4804/dialexeis/tmp/3.Service_Oriented_Architecture.pdf
- Γκουμόπουλος, Χ., *Κατανεμημένα Συστήματα*, από: http://www.icsd.aegean.gr/kaporisa/index_files/02_distsyst_basic_concepts.pdf
- Καραφασούλη, Κ., *Λειτουργικά Συστήματα* από: http://www.icsd.aegean.gr/lecturers/ckaraf/OperSyst/OS_DIAFAN_09_notes_bw.pdf
- The Open Group
- opap
- Παράλληλα και κατανεμημένα συστήματα
- Εισαγωγή στα Web services
- w3school
- http://en.wikipedia.org/wiki/JSON
- http://json.org/
- http://www.xml.com/pub/a/98/10/guide0.html?page=2#AEN58
- http://php.net/
- http://en.wikipedia.org/wiki/PHP
- https://github.com/yiisoft/yii2#readme
- http://httpd.apache.org/ABOUT_APACHE.html
- http://wiki.nginx.org/Main
- https://www.linux.com/
- http://www.linuxfoundation.org/
- https://www.mysql.com/
- http://getbootstrap.com/about/