Πανεπιστήμιο Πειραιώς
Τμήμα Ψηφιακών Συστημάτων

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΟ ΣΠΟΥΔΩΝ
«Κατεύθυνση Δικτυοκεντρικών Συστημάτων»

Διπλωματική Εργασία

Ανάπτυξη εφαρμογής με την χρήση BPEL και Web Services

Νικολόπουλος Επαμεινώνδας - ΑΜ:ΜΕ08094

Επιβλέπων: Μαρίνος Θεμιστοκλέους, Επίκουρος Καθηγητής

Πειραιάς 2012

Πανεπιστήμιο Πειραιώς

Περίληψη

Η παρούσα διπλωματική εργασία ασχολείται με την αυτοματοποίηση των επιχειρηματικών διαδικασιών μέσα στα πλαίσιο της υπερεσιοστραφής αρχιτεκτονικής SOA με την χρήση των τεχνολογιών Web services και BPEL. Η διαδικασία που αυτοματοποιείται είναι αυτή της διεκπεραίωσης μιας ηλεκτρονικής παραγγελίας προϊόντων από ένα κατάστημα. Σε αυτή την διαδικασία καλούνται να επικοινωνήσουν μεταξύ τους ετερογενή συστήματα εντός της ίδιας της επιχείρησης καθώς και ετερογενή συστήματα εκτός της επιχείρησης για να επιφέρουν το επιθυμητό αποτέλεσμα. Τέλος, παρουσιάζεται η εφαρμογή που αναπτύχθηκε και προσημειώνει την διαδικασία αυτή, αναλύεται η αρχιτεκτονική της και περιγράφονται οι τεχνολογίες που χρησιμοποιήθηκαν που επί το πλείστον αποτελούν τεχνολογίες αιχμής.

Λέξεις κλειδιά: Service Oriented Architecture (SOA), Web Services, Business Process Execution Language (BPEL), Business Process Management (BPM), BPEL4People, XML, WSDL, SOAP, UDDI

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

Περιεχόμενα

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

Ευρετήριο Σχημάτων

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

Ευρετήριο Πινάκων

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

1 Εισαγωγή

Στον αιώνα που ζούμε, η ικανότητα της γρήγορης προσαρμογής ενός οργανισμού στο δυναμικό περιβάλλον στο οποίο λειτουργεί, θεωρείται ως ένα πολύτιμο χαρακτηριστικό. Το επιχειρηματικό μοντέλο γίνεται ολοένα και πιο συνεργατικό, οι επιχειρήσεις έρχονται αντιμέτωπες με συνεχή ανταγωνισμό, και με μεγάλες επιχειρηματικές ευκαιρίες. Οι οργανισμοί συνάπτουν μεταξύ τους συνεταιριστικές σχέσεις για να είναι αποτελεσματικοί στις απαιτήσεις που θέτει η αγορά. Οι επιχειρηματικές διαδικασίες ακολουθούν τις απαιτήσεις τις αγοράς και ξεπερνούν τα όρια του οργανισμού, επιτρέποντας με αυτόν τον τρόπο στους οργανισμούς να χρησιμοποιούν-ανταλλάσουν κρίσιμες πληροφορίες με τους συνεργαζόμενους οργανισμούς.

Για να ακολουθήσουν οι οργανισμοί τις συνεχόμενες αλλαγές στις σχέσεις μεταξύ τον επιχειρήσεων και στις δια-οργανικές αλυσίδες αξίας, είναι ζωτικής σημασίας να αναπτύξουν προσαρμοστικά επιχειρησιακά συστήματα και αλυσίδες αξίας. Για να το επιτύχουν αυτό χρειάζονται μεθοδολογίες, και υποδομή που να υποστηρίζουν από άκρο-σε-άκρο την μοντελοποίηση της δια-οργανικής επιχειρηματικής διαδικασίας στο επιχειρησιακό επίπεδο, καθώς και τεχνολογίες - αρχιτεκτονικές που να υλοποιούν το μοντέλο που θα προκύψει στο επίπεδο των υπολογιστικών συστημάτων.

Το τρέχον τεχνολογικό περιβάλλον είναι γεμάτο από ένα πλήθος διαφορετικών τεχνολογιών, προτύπων και αρχιτεκτονικών που επιτρέπουν την ενοποίηση και την κάθετη ολοκλήρωση ετερογενών επιχειρησιακών συστημάτων. Η αρχιτεκτονική που είναι προσανατολισμένη στις υπηρεσίες (Service Oriented Architecture - SOA) επιτρέπει στους εταίρους να συνεργαστούν και διευκολύνει την σύνθεση δια-επιχειρηματικών διαδικασιών και βοηθάει τις επιχειρήσεις να ανταποκρίνονται γρήγορα και αποδοτικά ως προς το κόστος, στις αλλαγές των πληροφορικών τους συστημάτων.

Τα ERP (Enterprise Recourse Planning) καλύπτουν ένα μεγάλο μέρος των βασικών λειτουργιών ενός οργανισμού. Ωστόσο, άλλα πληροφορικά συστήματα χρειάζεται να έχουν πρόσβαση στις πληροφορίες που αποθηκεύονται σε αυτά, αυτό έχει ως συνέπεια την ανάπτυξη ακριβών λύσεων ολοκλήρωσης μεταξύ των ERP συστημάτων και των άλλων πληροφοριακών συστημάτων. Κάποιοι από τους μεγαλύτερους κατασκευαστές ERP συστημάτων έχουν προσθέσει στις πλατφόρμες τους την υποστήριξη Web services και οι υπόλοιποι οδηγούνται προς αυτή την κατεύθυνση προκειμένου να συγκλίνουν προς την SOA αρχιτεκτονική.

Τα Web Services είναι μια αναδυόμενη τεχνολογία και αποτελούν δομικά στοιχεία στην προσπάθεια για εταιρική ολοκλήρωση. Με την χρήση των WS οι εταιρίες μπορούν να αποκτήσουν τις πληροφορίες που χρειάζονται για να ανταποκριθούν στις τρέχουσες τους ανάγκες αποτελεσματικά, αυτό σημαίνει ότι με την ανάπτυξη μιας ολοκληρωμένης λύσης πληροφοριών στο Διαδίκτυο, τα συστήματα ERP των εταιρειών, δημοσιοποιούν πληροφορίες που δεν ήταν ποτέ πριν προσβάσιμες από άλλες επιχειρήσεις. Οι αγορές που δημιουργούνται με αυτόν τον τρόπο είναι εξ ορισμού πιο αποτελεσματικές, διότι επιτρέπουν στις εταιρείες να εστιάσουν τις προσπάθειές τους στην εξυπηρέτηση των πελατών και στη κερδοφορία. Οι δια-επιχειρησιακές διαδικασίες που δημιουργούνται με την χρήση των WS δημιούργησαν την ανάγκη για την δημιουργία προτύπων γλωσσών που να διαχειρίζονται –

ενορχηστρώνουν το σύνολο των WS που αλληλεπιδρούν μεταξύ τους. Η πιο διαδεδομένη πρότυπη γλώσσα που δημιουργήθηκε είναι η BPEL (Business Process Execution Language) για WS.

Η γλώσσα BPEL επιτρέπει στους οργανισμούς να αυτοματοποιήσουν τις επιχειρηματικές τους διαδικασίες μέσα από την ενορχήστρωση υπηρεσιών. Αναγκάζει τους οργανισμούς να σκέφτονται υπηρεσιο-κεντρικά: Οι υφιστάμενες λειτουργίες δημοσιεύονται σαν υπηρεσίες. Οι νέες εφαρμογές που δημιουργούνται μπορούν να χρησιμοποιήσουν μια ή περισσότερες υπηρεσίες και οι υπηρεσίες αυτές μπορούν να επαναχρησιμοποιηθούν από μία ή περισσότερες εφαρμογές.

Στην παρούσα διπλωματική αναπτύσσεται μια εφαρμογή η οποία προσομοιώνει την διαδικασία παραγγελίας προϊόντων μιας επιχείρησης. Η εφαρμογή βασίζεται στο διαδίκτυο για την ανταλλαγή των δεδομένων μεταξύ των διαφορετικών συστημάτων της επιχείρησης, των προμηθευτών, των τραπεζών και της μεταφορικής εταιρίας. Η εφαρμογή αποτελείται από Web services ενώ για την υλοποίηση της επιχειρησιακής λογικής, χρησιμοποιείται η γλώσσα BPEL και η BPEL4People. Τέλος έχει υλοποιηθεί μια σχεσιακή βάση δεδομένων για την αποθήκευση του συνόλου των δεδομένων.

1.1 Σκοπός και αντικείμενο της εργασίας

Σκοπός:

Σκοπός της εργασίας είναι να μελετήσει την αρχιτεκτονική SOA και να εστιάσει σε δύο δομικά της στοιχεία. Τα στοιχεία αυτά είναι τα Web Services και η ενορχήστρωση τους μέσω της γλώσσας BPEL.

Αντικείμενο εργασίας:

Αντικείμενο-Στόχος 1: Να μελετήσει την βιβλιογραφία και να κατανοήσει τις βασικές θεματικές περιοχές

Αντικείμενο-Στόχος 2: Να μελετήσει τις βασικές θεματικές περιοχές στην πράξη με την ανάπτυξη μιας εφαρμογής

Αντικείμενο-Στόχος 3: Να αναλύσει – σχολιάσει τα θέματα που μελέτησε στην πράξη και να εξάγει χρήσιμα συμπεράσματα

1.2 Δομή Εργασίας

Η παρούσα εργασία αποτελείται από πέντε κεφάλαια

Στο κεφάλαιο 1 γίνεται μια εισαγωγή στην θεματική περιοχή την οποία μελετά η παρούσα εργασία στον σκοπό και το αντικείμενο της εργασίας.

Το κεφάλαιο 2 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 με την χρήση της γλώσσας BPEL και η χορογραφία τους, καθώς και η διαχείριση επιχειρηματικών διαδικασιών.

Στο κεφάλαιο 3 παρουσιάζεται η θεωρία της μεθοδολογίας της έρευνας action research**.**

Στο κεφάλαιο 4 γίνεται ανάλυση της εφαρμογής που αναπτύχθηκε. Αναλύονται η αρχιτεκτονική, οι γλώσσες, οι εφαρμογές ανάπτυξης και η βάση δεδομένων που χρησιμοποιήθηκαν για την υλοποίηση της επιχειρηματικής διαδικασίας και γίνεται παρουσίαση της εφαρμογής που αναπτύχθηκε.

Το κεφάλαιο 5 συνοψίζει την εργασία και. παρουσιάζει τα κυριότερα συμπεράσματα.

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

2 Θεωρητικό υπόβαθρο

Σε αυτό το κεφάλαιο παρουσιάζονται οι τεχνολογίες που χρησιμοποιήθηκαν και η σχετική με αυτές θεωρία. Στην πρώτη ενότητα του κεφαλαίου εξετάζεται η Service Oriented Architecture (SOA) και η διακυβέρνηση της. Στην δεύτερη, τρίτη και πέμπτη ενότητα εξετάζεται η σχετική θεωρία σημαντικών δομικών της στοιχειών, όπως είναι τα Web services που ανήκουν στο στρώμα υπηρεσιών της αρχιτεκτονικής δομής της SOA, η χορογραφία, η ενορχήστρωση υπηρεσιών και η γλώσσα/πρότυπο Business Process Execution Language (BPEL) που ανήκουν στο στρώμα της σύνθεσης επιχειρησιακών διαδικασιών. Η BPEL είναι μια γλώσσα που χρησιμοποιεί τα Web services και ακολουθεί την σχετική με αυτήν θεωρία που εμπεριέχεται στην θεωρία της ενορχήστρωσης και της διαχείρισης των επιχειρηματικών διαδικασιών. Η διαχείριση επιχειρηματικών διαδικασιών και η σχέση της με την SOA εξετάζεται στην τέταρτη ενότητα αυτού του κεφαλαίου.

Πανεπιστήμιο Πειραιώς

2.1 Service Oriented Architecture (SOA)

Ένα από τα χαρακτηριστικά της Service Oriented Architecture (SOA) είναι ο κατακερματισμός των μεγάλων προβλημάτων σε μικρότερα επιμέρους προβλήματα, ακολουθώντας αυτή την λογική, θα εξεταστούν οι όροι service και service-oriented πριν την περαιτέρω ανάλυση της θεωρίας της SOA.

2.1.1 Service και Service-Oriented

Επειδή υπάρχουν πολλών ειδών service, στην SOA μας απασχολούν τα service που εμπλέκονται στην επιχειρηματική λογική ενός οργανισμού. Σύμφωνα με το The Open Group μια Υπηρεσία:

- Είναι μια λογική αναπαράσταση μιας επαναλαμβανόμενης επιχειρηματικής δραστηριότητας που έχει συγκεκριμένο αποτέλεσμα.
- Είναι αυτόνομη
- Μπορεί να είναι σύνθετη και να αποτελείται από άλλες υπηρεσίες.
- Ακολουθεί την λογική του μαύρου κουτιού για τους καταναλωτές της υπηρεσίας.

Καθώς ο όρος service-oriented υπάρχει εδώ και αρκετό καιρό και έχει χρησιμοποιηθεί μέσα σε διαφορετικά τεχνολογικά πλαίσια και για διαφορετικούς σκοπούς, μια σταθερά που έχει διαμορφωθεί από την αρχή της ύπαρξης του είναι το ότι αποτελεί μια ειδική προσέγγιση για ξεχωριστά προβλήματα. Αυτό βασίζεται στην λογική που απαιτεί η λύση ενός μεγάλου προβλήματος να μπορεί να επιμεριστεί και να διαχειριστεί καλύτερα εάν διασπαστεί σε μικρότερα κομμάτια. Καθένα από αυτά τα μικρότερα κομμάτια αντιστοιχεί σε κάποια ανησυχία ή σε ένα συγκεκριμένο μέρος του προβλήματος. Στις μέρες μας οι επιχειρηματικές διαδικασίες αντιπροσωπεύουν τέτοιου είδους προβλήματα τα οποία μπορεί να λυθούν με την χρήση της προσέγγισης του επιμερισμού σε επιχειρηματικές λειτουργίες. Με το διαδίκτυο να είναι η πιο βασική υποδομή αυτού του είδους η αποσύνθεση είναι επιτακτική ανάγκη καθώς οι επιχειρηματικές διαδικασίες διαπερνούν πολλαπλά όρια. Ως εκ τούτου απαιτείται η ενσωμάτωση και ολοκλήρωση των διάφορων εφαρμογών που χρησιμοποιούν διαφορετικές τεχνολογικές πλατφόρμες και μοντέλα δεδομένων.

2.1.2 Ορισμός SOA

Η SOA δεν αποτελεί μια νέα έννοια, η εταιρία SUN την περίοδο του 1990 είχε ορίσει την SOA για να περιγράψει το Jini το οποίο είναι ένα περιβάλλον για τον δυναμικό εντοπισμό και χρήση υπηρεσιών μέσα από το Διαδίκτυο. Τα Web services έχουν εκμεταλλευτεί την έννοια των υπηρεσιών που εισήχθηκε με την περιγραφεί του Jini. Τα Web services ακολουθούν την λογική της SOA, αρκετοί πιστεύουν ότι χρησιμοποιώντας τα την έχουν υιοθετήσει, αυτή η αντίληψη όμως είναι εσφαλμένη. Η SOA και τα Web services είναι δύο διαφορετικά πράγματα. Τα Web Services είναι η προτιμώμενη προτυποποιημένη τεχνολογία για το στρώμα υπηρεσιών της αρχιτεκτονικής δομής της SOA.

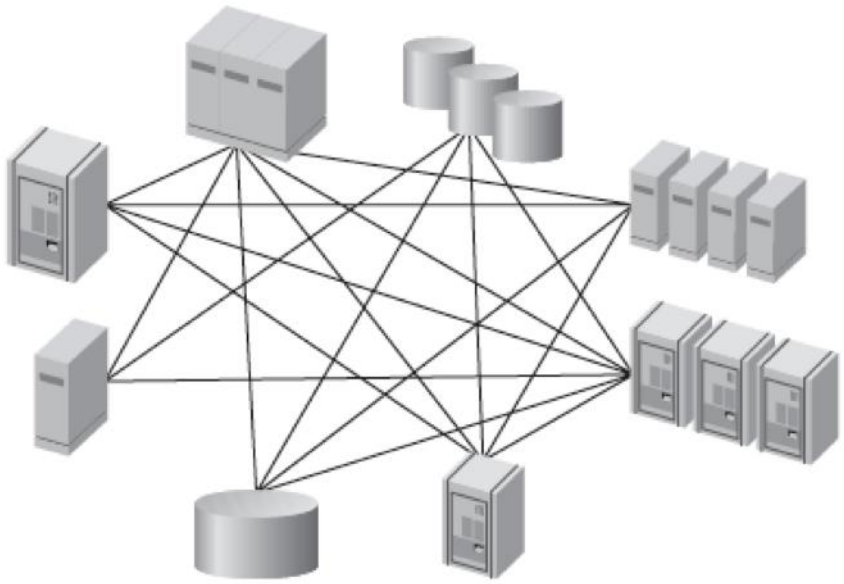
Πανεπιστήμιο Πειραιώς

Από τον πρώτο ορισμό που έδωσε η SUN την δεκαετία του 1990 μέχρι σήμερα έχουν προταθεί κατά καιρούς και αρκετοί άλλοι, ο επικρατέστερος ορισμός αυτή την στιγμή είναι ο αυτός που έχει διατυπωθεί από τον οργανισμό προτύπων OASIS και είναι ο ακόλουθος:

> *"Η SOA αρχιτεκτονική είναι ένα παράδειγμα για την οργάνωση και τη χρησιμοποίηση κατανεμημένων δυνατοτήτων οι οποίες μπορεί να είναι υπό τον έλεγχο διαφορετικών ιδιοκτησιακών χώρων (ownership domains). Παρέχει ένα ομογενοποιημένο τρόπο για την προσφορά, ανακάλυψη, αλληλεπίδραση και χρήση των δυνατότατων με σκοπό την παραγωγή επιθυμητών αποτελεσμάτων τα οποία είναι συνεπή με τις μετρήσιμες προϋποθέσεις και προσδοκίες"*
>
> *OASIS, 2006*

2.1.3 Γιατί SOA;

Στον ορισμό της SOA στην προηγούμενη υπό-ενότητα 2.1.2 γίνεται φανερό ότι τονίζονται η έννοια της ετερογένειας (σχήμα 1) και της διασποράς των κατανεμημένων συστημάτων ως η ανάγκη για την υιοθέτηση της SOA.



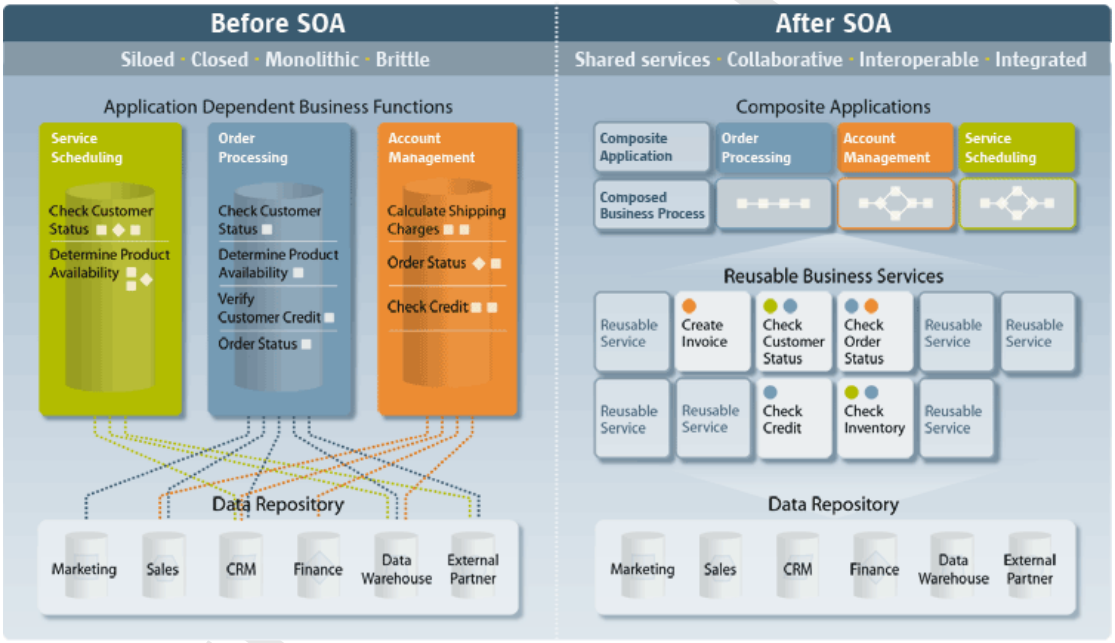
Σχήμα 1 Τα κατανεμημένα συστήματα είναι ετερογενή (Πηγή: Josuttis, 2007)

Μέσω των απλών παραδειγμάτων που παρουσιάζονται παρακάτω και στο σχήμα 2, μπορεί να φανεί η αναγκαιότητα της SOA για τον τελικό χρήστη και τον οργανισμό

- **Το πριν:** Στις συμβατικές αρχιτεκτονικές πληροφορικής, οι δραστηριότητες επιχειρηματικών διαδικασιών, οι εφαρμογές και τα

δεδομένα είναι συχνά "κλειδωμένα" σε ανεξάρτητα και ασύμβατα μεταξύ τους συστήματα, τα οποία είναι δαπανηρά στη συντήρησή τους και υποχρεώνουν τους χρήστες προκειμένου να εκτελέσουν συγκεκριμένες εργασίες να περιηγηθούν σε χωριστά δίκτυα, εφαρμογές και βάσεις δεδομένων. Επιπλέον οργανισμός χάνει την ικανότητα να έχει οποιοδήποτε προστιθέμενο κέρδος καθώς τα συστήματα αυτά έχουν κατασκευαστεί για να καλύπτουν συγκεκριμένες επιχειρησιακές ανάγκες και διαδικασίες. Αν προκύψουν νέες ανάγκες ή νέες διαδικασίες ο οργανισμός πρέπει είτε να κάνει εκτεταμένες αλλαγές στα συστήματα αυτά είτε να φτιάξει νέα από την αρχή.

- **Το μετά:** Με την SOA αρχιτεκτονική, οι χρήστες δε χρειάζεται πλέον να συνδέονται σε πολλά συστήματα, να αναζητούν σχετικά δεδομένα και να ενοποιούν τα αποτελέσματα μόνοι τους. Τα δεδομένα για τις δραστηριότητες επιχειρηματικών διαδικασιών προσφέρονται με τη μορφή μιας ενοποιημένης υπηρεσίας μέσω του δικτύου της επιχείρησης ή του διαδικτύου, μέσα από μία και μοναδική εφαρμογή. Έτσι ο οργανισμός αποκτά ευελιξία και ταχύτητα στην λήψη αποφάσεων.

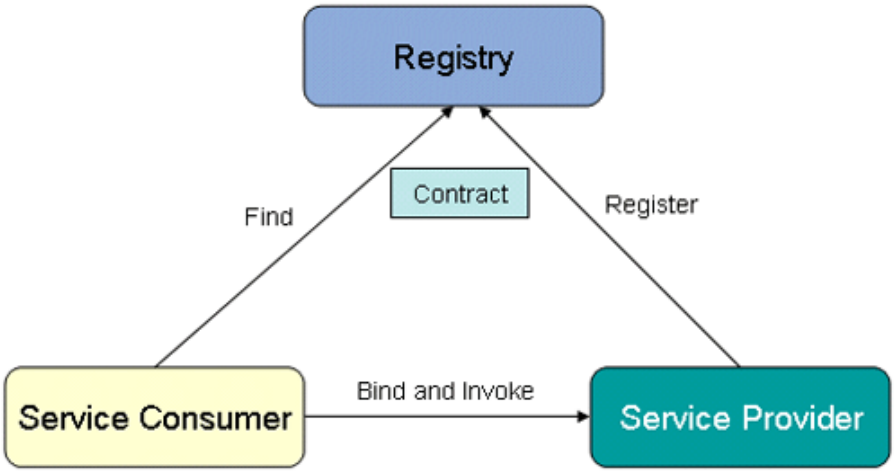


Σχήμα 2: Πριν και μετά την SOA (Πηγή: SUN, n.d).

2.1.4 Τα χαρακτηριστικά των υπηρεσιών SOA

Η SOA ακολουθεί το παράδειγμα αναζήτηση (find) - σύνδεση (bind) - εκτέλεση (execute). Για να καταστεί μια υπηρεσία διαθέσιμη προς εκτέλεση από τους πελάτες, ο πάροχος της υπηρεσίας θα πρέπει να την δηλώσει σε ένα δημόσιο μητρώο. Το μητρώο αυτό δεν είναι τίποτα άλλο παρά μια συλλογή από υπηρεσίες, τις περιγραφές τους και τις δικτυακές τους διευθύνσεις. Το μητρώο χρησιμοποιείται από τους καταναλωτές για την αναζήτηση υπηρεσιών που ταιριάζουν με τα κριτήρια απαιτήσεων που αυτοί θέτουν. Αν το μητρώο περιέχει την υπηρεσία που ταιριάζει

στα κριτήρια αναζήτησης, τους παρέχει την διεύθυνση της. Ο καταναλωτής πλέον μπορεί να έχει πρόσβαση στην υπηρεσία, να συνδεθεί με αυτήν καλώντας την στην διεύθυνση που του έχει δοθεί και τέλος να επικαλεσθεί τις μεθόδους της. Το προαναφερθέν παράδειγμα φαίνεται στο σχήμα 3.



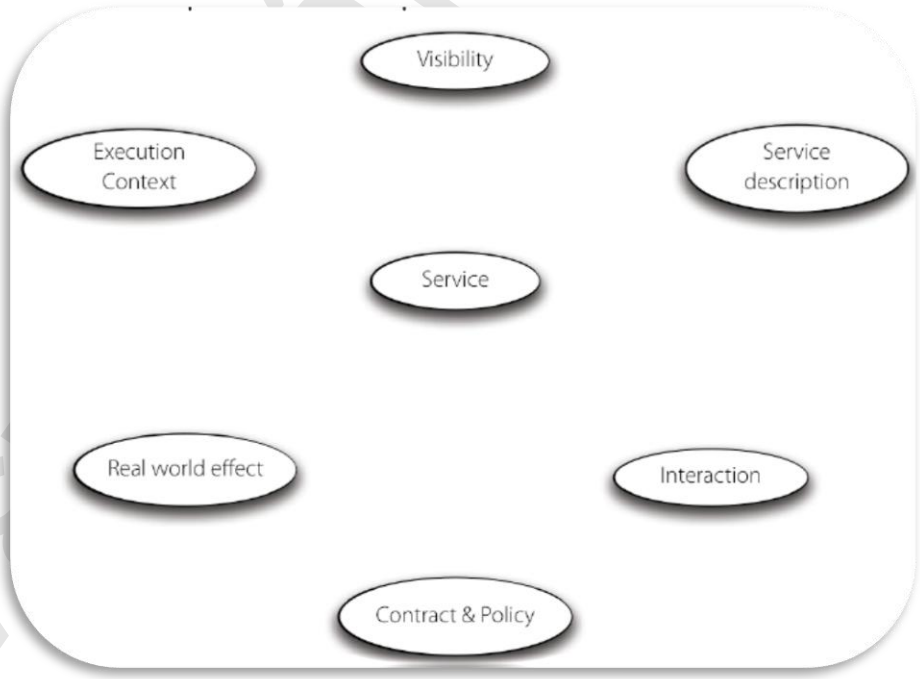
Σχήμα 3: Το παράδειγμα find-bind-execute (Πηγή: Mahmoud, 2005)

Το μοντέλο αναφοράς για την SOA του OASIS δίνει έμφαση στα χαρακτηριστικά που πρέπει να έχει μια υπηρεσία για να καταναλωθεί σχήμα 4. Στο πλαίσιο αυτό λοιπόν, μια υπηρεσία SOA πρέπει να έχει τα εξής χαρακτηριστικά:

- **Ορατότητα** (Visibility): Για να υπάρξει αλληλεπίδραση μεταξύ του καταναλωτή και του πάροχου τις υπηρεσίας πρέπει να "βλέπουν" ο ένας τον άλλο. Στην περίπτωση της SOA είναι σημαντικό να δοθεί έμφαση στην ορατότητα γιατί 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: Ο εκκινητής μιας αλληλεπίδρασης πρέπει να γνωρίζει τα άλλα μέρη.
  - Προθυμία: οι συμμετέχοντες πρέπει να είναι προδιατεθειμένοι να αλληλεπιδράσουν.
  - Πρόσβαση: οι συμμετέχοντες πρέπει να είναι σε θέση να αλληλεπιδράσουν.
- **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 ιδιωτικότητα, 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 4: Κύριες έννοιες του μοντέλου αναφοράς (Πηγή: OASIS, 2006)

2.1.5 Ισχυρή και χαλαρή συνδεσιμότητα

Σε ένα παραδοσιακό κατανεμημένο υπολογιστικό περιβάλλον και πριν την έλευση της SOA και των Web services, η δυσκολία της κλιμάκωσης, η μη υποστήριξη διαφοροποίησης, καθώς και η μη υποστήριξη συνεχούς βελτίωσης του κώδικα του των εφαρμογών ήταν κάποια από τα σημαντικά μειονεκτήματα που επηρέαζαν την ευελιξία και έπρεπε να επιλυθούν. Πηγή αυτού του προβλήματος ήταν οι ισχυρή συνδεσιμότητα (tight coupling) που είχαν μεταξύ τους οι κατανεμημένες εφαρμογές.

Η συνδεσιμότητα είναι ένα μέτρο για τον καθορισμό της εξάρτησης μεταξύ υπηρεσιών. Όσο πιο ισχυρή είναι η εξάρτηση μια υπηρεσίας με μια άλλη τόσο πιο ισχυρή είναι η συνδεσιμότητα μεταξύ τους. Μια ισχυρή εξάρτηση κάνει τις εφαρμογές μη 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. Η χαλαρή συνδεσιμότητα λοιπόν οδηγεί στην χαλαρή εξάρτηση δίνοντας σε ένα σύστημα μεγαλύτερη ευελιξία και μικρότερες επιπτώσεις σε τυχόν τροποποιήσεις ή λάθη. Η SOA και τα Web services, προάγουν την χαλαρή συνδεσιμότητα.

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

	Tight coupling	**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

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

Η ισχυρή συνδεσιμότητα αν και έχει αρκετά μειονεκτήματα δεν αποτελεί μια κακή λύση εν αντιθέσει σε κάποιες περιπτώσεις η χρήση της δίνει καλύτερα αποτελέσματα σε σχέση με χρήση της χαλαρής συνδεσιμότητα, όπως όταν οι εξαρτήσεις είναι κρίσιμης σημασίας για το σχεδιασμό.

Πανεπιστήμιο Πειραιώς

2.1.6 Αρχιτεκτονική Δομή

Τα πρόσφατα συστήματα είναι δομημένα με δυο επίπεδα αρχιτεκτονικής δομής (two-tier layers), στα οποία ο πελάτης έχει άμεση πρόσβαση στη βάση δεδομένων και στα δικτυακά API, χωρίς να υπάρχει κάποιο λογικό μοντέλο μεταξύ αυτού και των παραπάνω. Αυτή η προσέγγιση των δύο επιπέδων αρχιτεκτονικής δομής μπορεί ακόμα να χρησιμοποιηθεί σε συστήματα λογισμικού, όπου η ανάπτυξη είναι μικρού μεγέθους και πρωτότυπη. Επίσης αυτή η προσέγγιση, μπορεί να εφαρμοστεί σε κάποιες ενότητες προηγμένων συστημάτων για την παροχή ορισμένων λειτουργιών. Ωστόσο, μια ανάπτυξη εφαρμογής με δυο επίπεδα αρχιτεκτονικής δομής είναι περιορισμένη σε συστήματα μικρής διάρκειας ζωής και δεν υποστηρίζει ευέλικτα APIs. Δεν παρέχει επαρκή απομόνωση του κώδικα της εφαρμογής από τον πελάτη και αυτό είναι που κάνει την αρχιτεκτονική μη ευέλικτη και μη κλιμακούμενη κατά την χρήση της από πολλούς ταυτόχρονους χρήστες με ό,τι συνεπάγεται αυτό.

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 προσέγγιση στην ανάπτυξη λογισμικού για ευέλικτη χρήση των πόρων του συστήματος και την διαχείριση δεδομένων.

Η αρχιτεκτονική SOA είναι βασισμένη στην ανάπτυξη εφαρμογών ν-στρωμάτων στην οποία οι υπηρεσίες βρίσκονται πάνω από τα στοιχεία (components) που είναι αρμόδια για την παροχή συγκεκριμένων λειτουργιών και για την εξασφάλιση των απαιτήσεων ποιότητας των υπηρεσιών**,** όπως φαίνεται στο σχήμα 5.

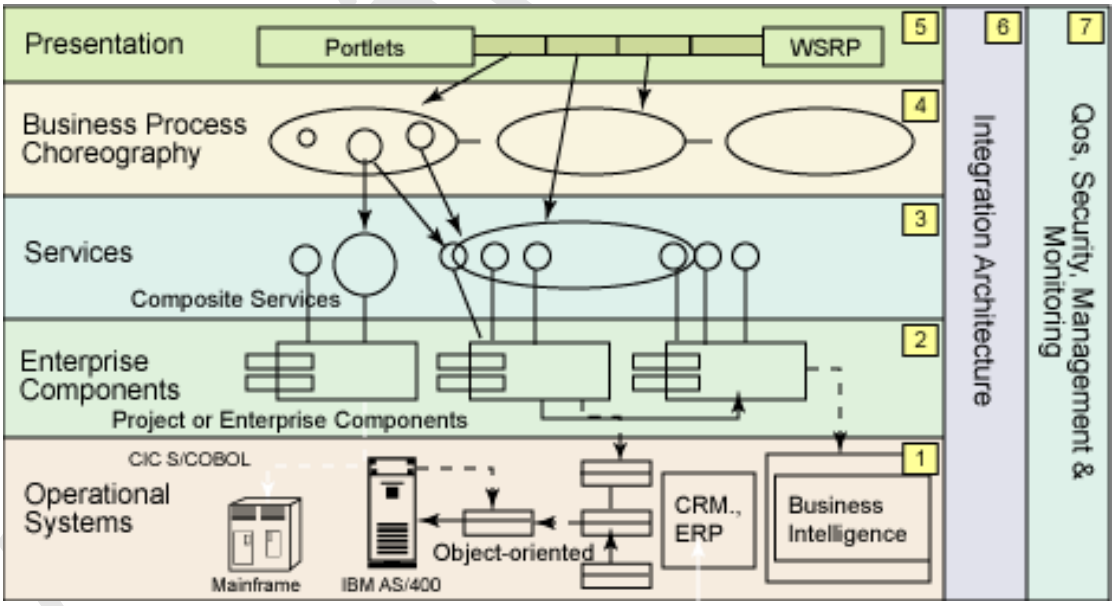
Κάθε στρώμα στο SOA έχει συγκεκριμένα αρχιτεκτονικά χαρακτηριστικά, όπως περιγράφονται παρακάτω:

- **Στρώμα επιχειρησιακών συστημάτων (Operational Systems):** Αυτό το στρώμα περιέχει τις υπάρχουσες εφαρμογές, όπως τα πακέτα εφαρμογών CRM και το ERP, τα αντικειμενοστραφή συστήματα και οι εφαρμογές επιχειρηματικής ευφυΐας. Οι εφαρμογές αυτές παρέχουν το υπόβαθρο για τις υπηρεσίες και κάθε μια από αυτές έχει το δικές τις ιδιόκτητες δομές, βάσεις δεδομένων και πρόσβαση σε άλλους πόρους του συστήματος.
- **Στρώμα επιχειρησιακών στοιχείων (Enterprise Components):** Αυτά είναι εξειδικευμένα στοιχεία (components) που παρέχουν στις υπηρεσίες συγκεκριμένες λειτουργίες και καλύπτουν συγκεκριμένες απαιτήσεις τους. Πρόκειται για τα περιουσιακά στοιχεία των επιχειρήσεων που χρησιμεύουν στην εφαρμογή υπηρεσιών και άλλων αναγκών του συστήματος, όπως είναι η διαχείριση του, η διαθεσιμότητα του και η εξισορρόπηση του φορτίου των υπηρεσιών του.
- **Στρώμα υπηρεσιών (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

Πανεπιστήμιο Πειραιώς

Διαδίκτυο. Συνήθως χρησιμοποιούνται τα Web Services (ενότητα 2.2) σε αυτό το στρώμα.

- **Στρώμα σύνθεσης επιχειρησιακών διαδικασιών (Business Process Composition):** Οι υπηρεσίες μπορούν να ενωθούν και να αλληλεπιδράσουν σαν να είναι μια ενιαία εφαρμογή μέσω της ενορχήστρωσης ή της χορογραφίας υπηρεσιών (ενότητα 2.3), η οποία εν τέλη θα είναι σε θέση να υποστηρίζει συγκεκριμένες περιπτώσεις χρήσης και επιχειρησιακές διαδικασίες. Στο σχήμα 5 αυτό το στρώμα ονομάζεται Business Process Choreography.
- **Στρώμα παρουσίασης ή πρόσβασης (Presentation):** Παρέχει τις διεπαφές χρήστη για τις υπηρεσίες και τις συντεθειμένες εφαρμογές. Αν και το στρώμα παρουσίασης δεν αποτελεί πρωτεύουσα ανησυχία της SOA αρχιτεκτονικής απλά συμπεριλαμβάνεται στην αρχιτεκτονική για την πιθανή ανάπτυξη προτύπων τα οποία θα μπορέσουν να αξιοποιήσουν τα Web services στο στρώμα εφαρμογών ή παρουσίασης.
- **Στρώμα ενσωμάτωσης (Integration):** 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).
- **Στρώμα ποιότητας των υπηρεσιών (QoS):** Αυτό το στρώμα παρέχει τα στοιχεία που απαιτούνται για την παρακολούθηση, την ασφάλεια, την διαχείριση και την διατήρηση της ποιότητας των υπηρεσιών.



Σχήμα 5: Τα στρώματα της SOA αρχιτεκτονικής (Πηγή: Arsanjani, 2005)

2.1.7 Πλεονεκτήματα από την υιοθέτηση της.

Η χρήση του SOA έχει πολλά οφέλη που βοηθούν έναν οργανισμό να προσαρμοστεί με επιτυχία στο σημερινό δυναμικό επιχειρηματικό τοπίο. Ένας από του κύριους στόχους μιας επιχείρησης είναι να μπορεί να αντεπεξέλθει με επιτυχία στις συνεχώς μεταβαλλόμενες επιχειρηματικές συνθήκες. Επίσης είναι απαραίτητο να μπορεί να διατηρήσει το ανταγωνιστικό της πλεονέκτημα στην σχέση κόστους-αποτελεσματικότητας που συνεπάγεται σε υψηλό ROI. Παρακάτω αναφέρονται κάποια από τα βασικά πλεονεκτήματα την SOA αρχιτεκτονικής.

- **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.
- **Επαναχρησιμοποίηση:** Οι υπηρεσίες έχουν δημιουργηθεί έτσι ώστε να καλύπτουν άμεσα τις απαιτήσεις του επιπέδου εφαρμογών ενώ ταυτόχρονα να υποστηρίζουν σε κάποιο βαθμό την επαναχρησιμοποίηση τους από μελλοντικούς εν δυνάμει πελάτες. Αυτό ελαχιστοποιεί την προσπάθεια που χρειάζεται για την ανάπτυξη νέων υπηρεσιών-εφαρμογών έχοντας ως φυσικό επακόλουθο την μείωση του κόστους.
- **Γρηγορότερη εναρμόνιση με την αγορά:** Με τις υπηρεσίες του πυρήνα της επιχείρησης να είναι εκτεθειμένες στο δίκτυο η ανάπτυξη και η χρήση νέων υπηρεσιών γίνεται πιο γρήγορα μειώνοντας δραματικά τον χρόνο που χρειάζεται η επιχείρηση να εναρμονιστεί με τα νέα δεδομένα της αγοράς, καθιστώντας έτσι την αντίδραση της πιο ευέλικτη και αποτελεσματική κάθε φορά που θα χρειαστεί να αλλάξουν οι επιχειρησιακές της ανάγκες. Αυτό επιτυγχάνεται γιατί οι νέες υπηρεσίες επαναχρησιμοποιούν τις υφιστάμενες υπηρεσίες και τα συστατικά τους, μειώνοντας έτσι το σχεδιασμό, την ανάπτυξη, τη δοκιμή και το χρόνο εγκατάστασης-υιοθέτησης.
- **Μετασχηματισμός των δεδομένων:** Με την SOA μπορεί να παρασχεθεί μετατροπή δεδομένων από μια μορφή σε μία άλλη, μέσω αυτοματοποιημένων αντιστοιχίσεων πεδίων. Η μετατροπή δεδομένων από ειδικά γραμμένο λογισμικό είναι αρκετά δαπανηρή. Η χρήση γενικών (generic) μεταφραστών δεδομένων μπορεί να επιφέρει σημαντική μείωση κόστους.
- **Χρήση έξω-επιχειρησιακών πόρων:** Η SOA αρχιτεκτονική επιτρέπει την σύνδεση του οργανισμού με εξωτερικούς εταίρους, 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
- **Καλύτερη ευθυγράμμιση μεταξύ επιχείρησης και πληροφοριακών συστημάτων:** Η ευθυγράμμιση επιτυγχάνεται όταν όλες τις λειτουργίες των

Πανεπιστήμιο Πειραιώς

επιχειρήσεων (συμπεριλαμβανομένων και των IT) υποστηρίζουν κοινούς στόχους και αποτελέσματα. Οι υπηρεσίες που παρέχονται από το IT ενός οργανισμού που χρησιμοποιούν μια προσέγγιση SOA 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 και τη SOA 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.

2.1.8 Παγίδες κατά την υιοθέτηση της

Κάθε αρχιτεκτονική έχει τα δικά της μειονεκτήματα το ίδιο και η SOA, ο δρόμος για την υιοθέτηση της είναι δύσκολος και κρύβει παγίδες σύμφωνα που αξίζει να αναφερθούν. Παγίδες κρύβονται όταν:

- Κάποιος προσπαθεί να κατασκευάσει μια SOA αρχιτεκτονική σαν μια παραδοσιακή κατανεμημένη αρχιτεκτονική. Πιθανότατα είναι το νούμερο ένα λάθος που αντιμετωπίζουν οι οργανισμοί. Στην προσπάθεια τους για την επίτευξη της SOA κατασκευάζουν υπηρεσιοστραφής υπηρεσίες με τον ίδιο τρόπο με τον οποίο κατασκευάζουν παραδοσιακά κατανεμημένα συστήματα. Η SOA δεν είναι ίδια με άλλες προσεγγίσεις, ούτε μπορούν τα εφαρμοστούν σε αυτή τα χαρακτηριστικά ισχυρής συνδεσιμότητας.
- Δεν υπάρχει προτυποποίηση της SOA. Η SOA όπως και κάθε άλλη αρχιτεκτονική απαιτεί την επιβολή εσωτερικής προτυποποίησης για να μπορούν να γίνουν αντιληπτά πλήρως τα πλεονεκτήματα της. Δεν μπορεί να σχεδιάζεται και υλοποιείται μια SOA για ένα τμήμα οργανισμού χωρίς να λαμβάνονται υπόψη τα άλλα τμήματα του οργανισμού που θα αλληλεπιδράσει. Ο σχεδιασμός πρέπει να αφορά όλον τον οργανισμό και όχι μέρη του.
- Το σχέδιο μετάβασης έχει παραληφθεί. Οι πιθανότητες επιτυχούς μετάβασης θα έχουν εκμηδενιστεί χωρίς την χρήση ενός σχεδίου μετάβασης, γιατί η έκταση που τα τελικά σημεία πρόσβασης υπηρεσιών θα έχουν τοποθετηθεί σε μια επιχείρηση μπορεί να οδηγήσει στην αναθεώρηση του περιβάλλοντος του IT της. Οι επιδράσεις μια φτωχά εκτελεσμένης μετάβασης μπορεί να είναι σημαντικές.
- Δεν ξεκινούν με την θεμελιώδη XML αρχιτεκτονική: Στον σημερινό κόσμο της SOA τα πάντα ξεκινούν με τα Web services. Η δήλωση αυτή έχει καθιερωθεί σε ορισμένους οργανισμούς, αλλά δεν είναι εντελώς αληθής. Στον κόσμο της σημερινής SOA τα πάντα ξεκινούν με 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 SOA. Έτσι, δίνεται μεγάλη προσοχή στο τρόπο με τον οποίο τα δεδομένα μεταφέρονται μεταξύ των υπηρεσιών, ο τρόπος όμως με

Πανεπιστήμιο Πειραιώς

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 SOA και οι αδυναμίες του θα έχουν επιπτώσεις σε οποιοδήποτε λύση κατασκευαστεί πάνω σε αυτό.

- Δεν αντιλαμβάνονται τις απαιτήσεις επιδόσεων της SOA. Η χαλαρή συνδεσιμότητα έχει το τίμημα της. 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 υπόψη του, 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.

Πανεπιστήμιο Πειραιώς

2.1.9 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:

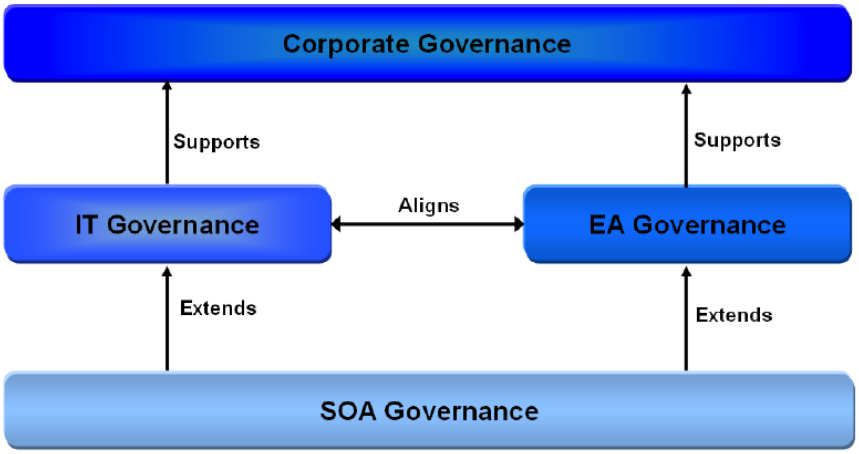
> *"Η Διακυβέρνηση SOA θα πρέπει να θεωρείται ως η εφαρμογή της Εταιρικής Διακυβέρνησης της Διακυβέρνηση IT και της Διακυβέρνηση της EA προς τη Service Oriented Architecture. Στην πραγματικότητα η SOA Διακυβέρνηση επεκτείνετε στην Διακυβέρνηση IT και στην Διακυβέρνηση EA, διασφαλίζοντας ότι πληρούνται 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 το σχήμα 6 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.

Πανεπιστήμιο Πειραιώς

- 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.



Σχήμα 6: Οι σχέσεις μιας Διακυβέρνησης SOA (Πηγή: The Open Group, 2009)

Από τον παραπάνω ορισμό γίνεται αντιληπτό ότι η SOA αρχιτεκτονική για είναι επιτυχημένη πρέπει να έχει μια μορφή Διακυβέρνησης. Μιας Διακυβέρνησης η οποία δεν εκτείνεται μόνο στα όρια της SOA αλλά και σε άλλα τμήματα ενός οργανισμού. Η SOA λοιπόν πρέπει να είναι ευθυγραμμισμένη με την επιχειρησιακή δομή του οργανισμού. Σύμφωνα με το Carter (2007), εάν η επιχειρησιακή δομή μιας εταιρίας και η αρχιτεκτονική του IT δεν είναι σε συγχρονισμό ή ευθυγράμμιση, η εταιρεία δεν μπορεί να αποδώσει τα μέγιστα ή να οδηγηθεί στην επιθυμητή καινοτομία.

2.1.9.1 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,

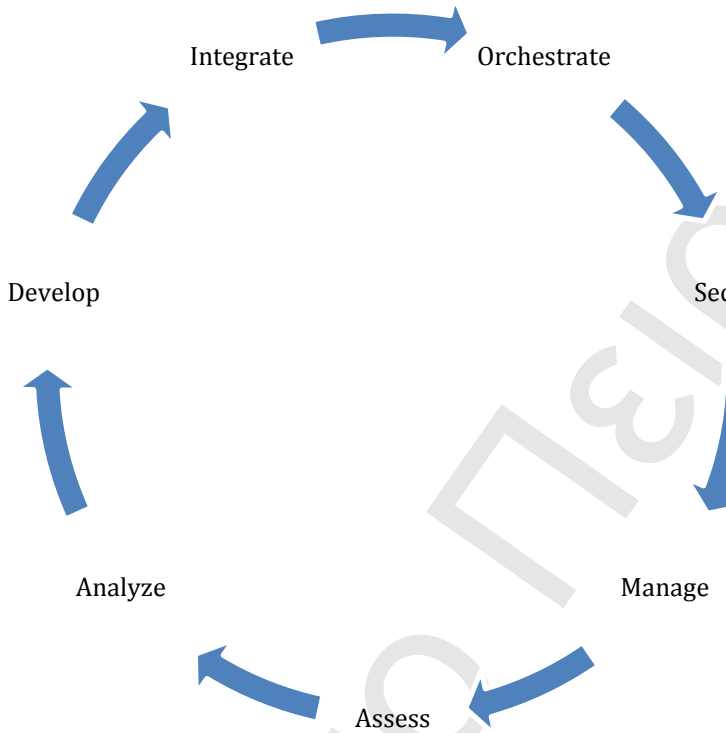
  που ορίστηκαν στο προηγούμενο στάδιο. Επιπλέον, σε αυτό το στάδιο γίνεται εκπαίδευση και ανάπτυξη πάνω σε αναμενόμενες συμπεριφορές, πρακτικές.

- 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.

2.1.10 Κύκλος ζωής SOA

Ο δρόμος προς την SOA αρχιτεκτονική αποτελείται από στάδια. Τα στάδια αυτά είναι: το στάδιο της ανάπτυξης, το στάδιο της ενσωμάτωσης, το στάδιο της ενορχήστρωσης και χορογραφίας, το στάδιο της διασφάλισης, το στάδιο της διαχείρισης, το στάδιο της πρόσβασης και τέλος το στάδιο της ανάλυσης. Όλα μαζί να στάδια συνθέτουν μια προσέγγιση κύκλου ζωής για την SOA σχήμα 7.

Πανεπιστήμιο Πειραιώς



Σχήμα 7: Μια προσέγγιση ενός κύκλου ζωής για την SOA.

Αναλυτικά:

- **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 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

Πανεπιστήμιο Πειραιώς

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.

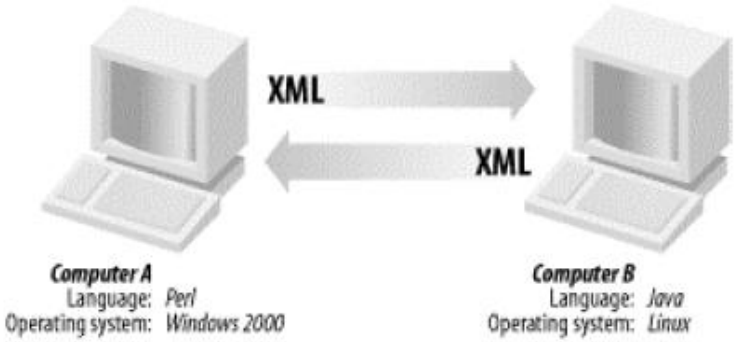
Ακολουθώντας ένα κύκλο ζωής, όπως αυτόν που παρουσιάζεται σε αυτή τη προσέγγιση, εξασφαλίζει στον οργανισμό ότι οι υπηρεσίες του κατά την ανάπτυξη τους θα έχουν την κατάλληλη ασφάλεια, αξιοπιστία και διαθεσιμότητα.

Πανεπιστήμιο Πειραιώς

2.2 Web services

Τα Web services προσφέρουν μια κατανεμημένη υπολογιστική προσέγγιση για την ολοκλήρωση των ετερογενών εφαρμογών των κατανεμημένων συστημάτων μέσω του Διαδικτύου. Τα ανοιχτά πρότυπα στα οποία βασίζονται υποστηρίζουν την διαλειτουργικότητα μεταξύ των διαφορετικών λύσεων των κατασκευαστών.

Ένα παράδειγμα μιας πολύ απλής εφαρμογής (σχήμα 8) ενός Web Service χρησιμοποιεί τις τεχνολογίες XML και HTTP που υποστηρίζονται σε μεγάλη κλίμακα από φυλλομετρητές ιστού και εξυπηρετητές ιστού.



Σχήμα 8:Ένα απλό Web Service (Πηγή: Cerami, 2002)

Τα Web services είναι μια αναδυόμενη τεχνολογία και αποτελούν δομικά στοιχεία στην προσπάθεια για εταιρική ολοκλήρωση. Στο Web services Glossary του οργανισμού W3C (W3C, 2004)δίνεται ο εξής 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*

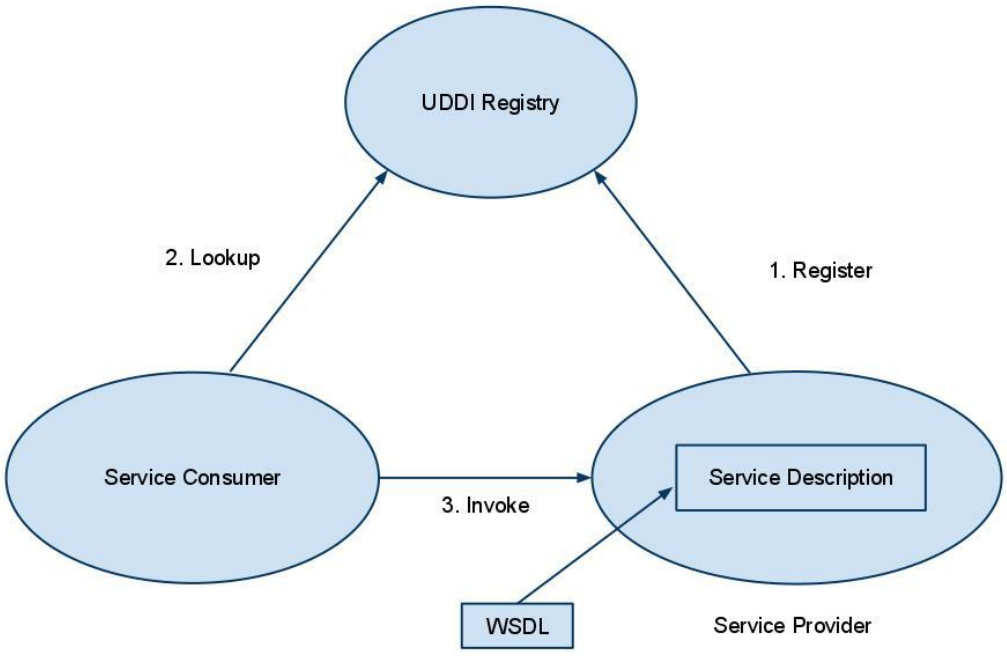
Στο ορισμό διαφαίνεται το βασικό χαρακτηριστικό των Web Service που είναι η διαλειτουργικότητα και η τεχνολογίες που εμπλέκονται για την επίτευξη αυτής.

2.2.1 Ρόλοι

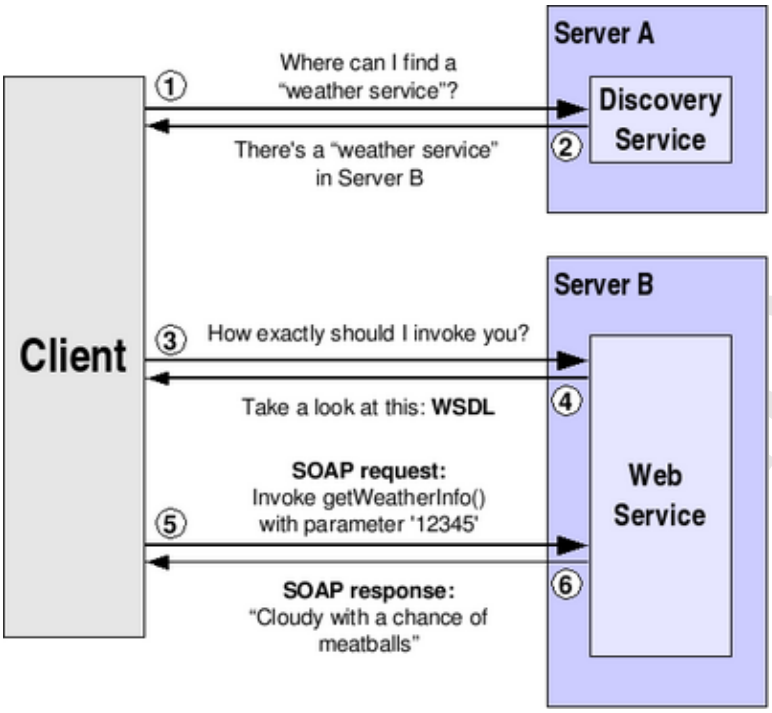
Στην αρχιτεκτονική των WS όπως και στην SOA υπάρχουν τρεις κύριοι ρόλοι και είναι παρόμοιοι:

- **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.
- **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.
- **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 9 αποτυπώνει τους κύριους ρόλους ενός WS και πως αλληλεπιδρούν μεταξύ τους και το σχήμα 10 δείχνει μια τυπική κλήση ενός WS



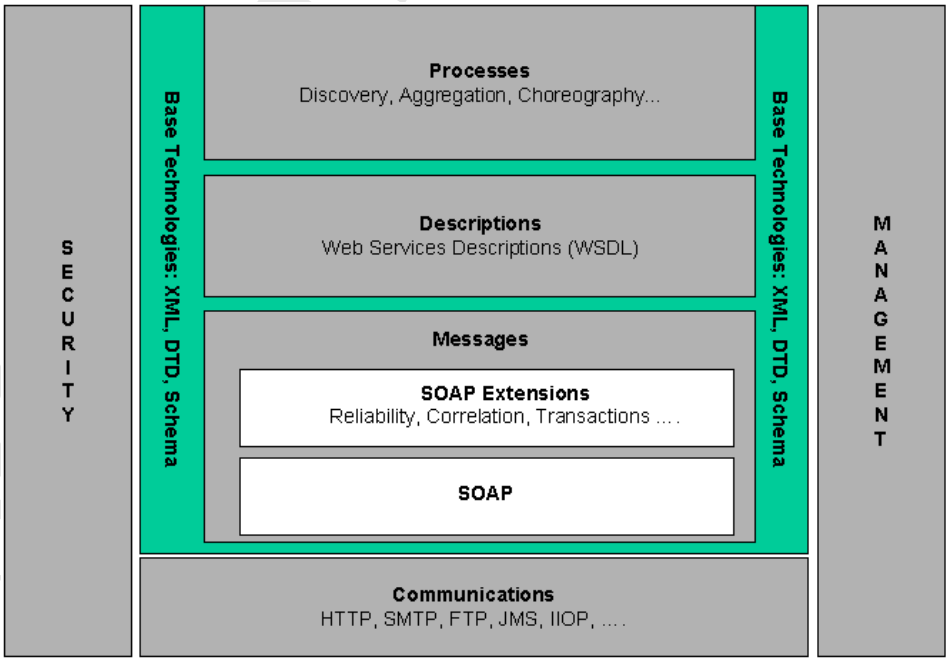
Σχήμα 9: Οι ρόλοι ενός Web Service (Πηγή: Gunathilake, 2011)



Σχήμα 10: Ένα τυπικό παράδειγμα κλήσης ενός Web service (Πηγή: Sotomayor, 2005).

2.2.2 Αρχιτεκτονική Στοίβα

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



Σχήμα 11: Η αρχιτεκτονική Στοίβα των Web Services (Πηγή: W3C, 2004)

- **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.

2.2.3 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.

2.2.3.1 Extensible Markup Language (XML)

Η XML λύνει τη βασική απαίτηση μιας τεχνολογίας που εμφανίζεται σε πολλά μέρη. Με την προσφορά ενός ευέλικτου, και με την δυνατότητα επέκτασης της μορφή των δεδομένων, προτύπου, η XML μειώνει σημαντικά το φορτίο της ανάπτυξης των πολλών τεχνολογιών που απαιτούνται για να εξασφαλιστεί η επιτυχία των Web services.

Πανεπιστήμιο Πάτρας

Ακολουθεί Ορισμός:

> *"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 όπως φαίνεται και στο σχήμα 11**.** Τα μηνύματα (SOAP) οι περιγραφές (WSDL) και οι διαδικασίες (Ανακάλυψης, συγκέντρωσης, χορογραφίας, ενορχήστρωσης, κλπ) βασίζονται σε αυτή.

2.2.3.2 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).

Πανεπιστήμιο Πειραιώς

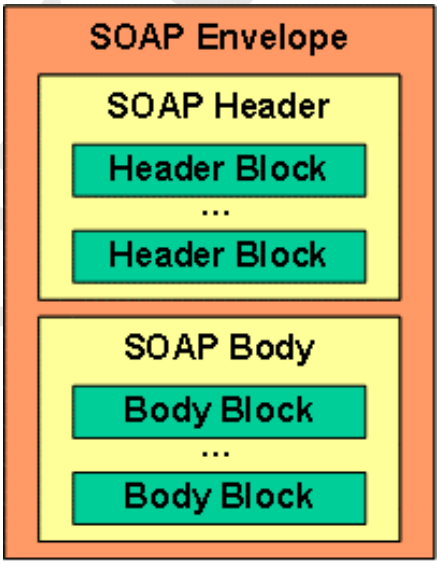
Ακολουθεί Ορισμός:

> *"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*

Η παρούσα προδιαγραφή παρέχει τις λεπτομέρειες μόνο για δύο MEPs: το HTTP και το SMTP. Τα υπόλοιπα χαρακτηριστικά γνωρίσματα αφήνονται να καθοριστούν από άλλες προδιαγραφές.

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 12. Ένα μήνυμα έχει δύο βασικά μέρη:

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



Σχήμα 12:Δομή μηνύματος SOAP (Πηγή: W3C, 2004)

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

Πανεπιστήμιο Πειραιώς

- 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.
- Και 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.

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.

2.2.3.3 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

Πανεπιστήμιο Πειραιώς

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 .

Ακολουθεί ορισμός:

> *"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*

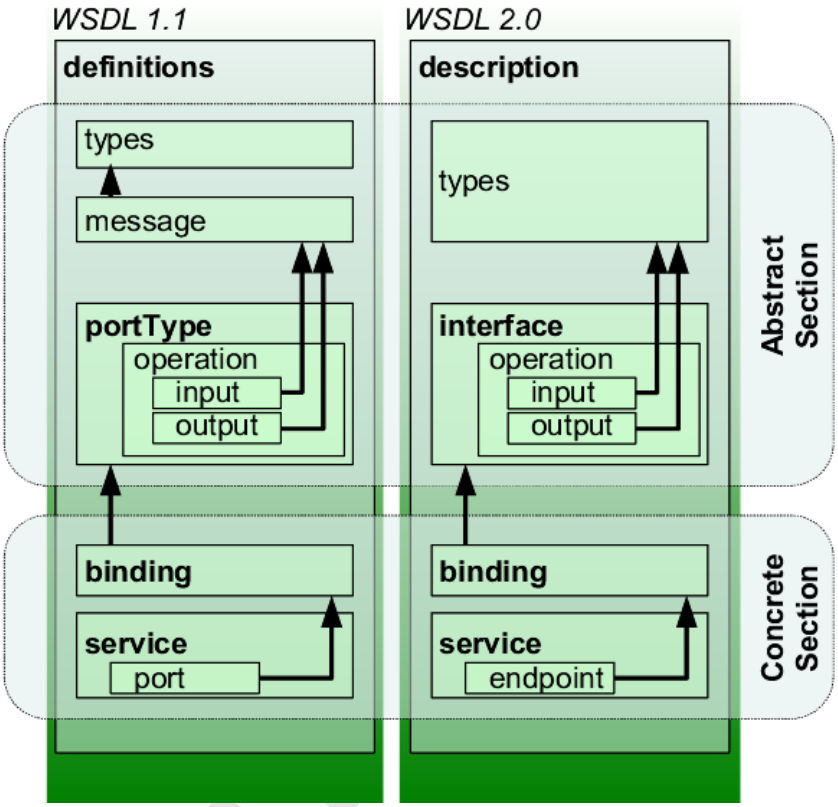
Ένα έγγραφο WSDL 1.1 χρησιμοποιεί τα παρακάτω αντικείμενα για τον ορισμό των web services :

- 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.

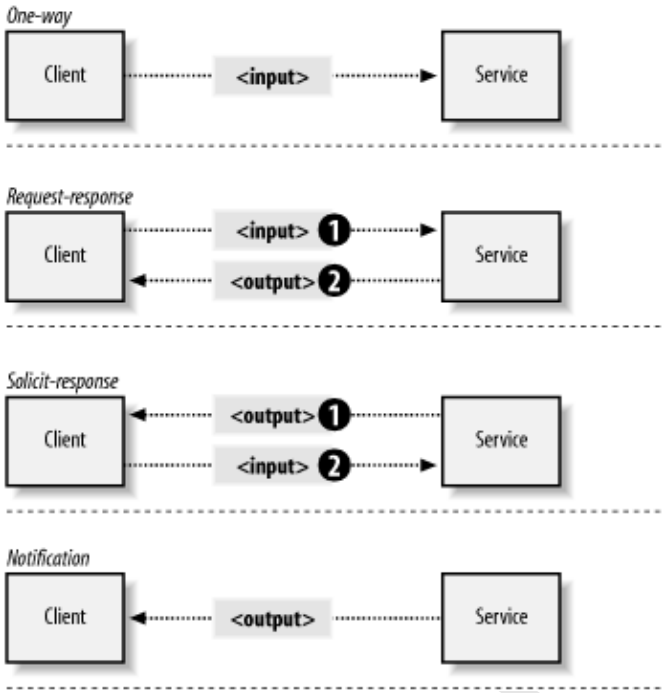
Πανεπιστήμιο Πειραιώς



Σχήμα 13:Αναπαράσταση των εννοιών που καθορίζονται στα έγγραφα WSDL 1.1 και WSDL 2.0 (Πηγή: Wikipedia, 2011)

Στο WSDL 1.1 υποστηρίζονται τρία πρότυπα ανταλλαγής μηνυμάτων σχήμα 14:

- 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 14: Πρότυπα ανταλλαγής μηνυμάτωνWSDL 1.1 (Πηγή: Weerawarana S. , Curbera, Leymann, Storey, & Ferguson, 2005)

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"

2.2.4 Universal Description, Discovery and Integration (UDDI)

Το UDDI ως τεχνική προδιαγραφή παρέχει μια μέθοδο για δημοσίευση και εύρεση των περιγραφών μιας υπηρεσίας. Είναι μια κεντρική υπηρεσία καταλόγου, όπου τα web services μπορούν να καταχωρηθούν και να προσδιοριστούν σε έναν παροχέα υπηρεσιών. Είναι μια πρωτοβουλία των εταιρειών Microsoft, IBM και Arriba που το Σεπτέμβριο του 2000 εξέδωσαν την έκδοση 1.0 του UDDI, η οποία επέτρεπε στις

επιχειρήσεις τη γρήγορη και δυναμική εύρεση καθώς και συναλλαγή με κάθε άλλη υπηρεσία, Μετέπειτα έδειξαν και άλλες εταιρίες ενδιαφέρον και προστέθηκαν και αυτές στην πρωτοβουλία. Ακολούθησε το UDDI 2.0 το Μάιο του 2001, με επιπλέον χαρακτηριστικά, όπως η μοντελοποίηση για την υποστήριξη σύνθετων οργανισμών, πιο ισχυρή κατηγοριοποίηση και υποστήριξη αναγνωριστικού για τους πελάτες, ενισχυμένη αναζήτηση, διεθνοποίηση, αναπαραγωγή με βάση τους peer. Η τελευταία έκδοση του UDDI είναι η έκδοση 3.0.2.

Ορισμός του UDDI όπως δίνεται από το OASIS περιληπτικά είναι ο ακόλουθος: Τα Web services Τα web services έχουν νόημα μόνο όταν δυνητικοί χρήστες μπορούν να βρουν πληροφορίες ικανές ώστε να επιτρέψουν την εκτέλεσή τους. Το UDDI που βασίζεται στην XML και στο SOAP για την επικοινωνία είναι αυτό που επιτρέπει τον καθορισμό ενός συνόλου από υπηρεσίες που υποστηρίζουν την περιγραφή και την ανακάλυψη των πάροχων των Web services, το διαθέσιμων Web services, καθώς και τις τεχνικές λεπτομέρειες που χρειάζονται για την κλήση ενός Web service.

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 ξεκινήσουν 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".

Το UDDI έχει διαφορετικές χρήσεις, με βάση την προοπτική που κάποιος το χρησιμοποιεί. Από την σκοπιά ενός business analyst, το UDDI είναι παρόμοιο με μια μηχανή αναζήτησης του Διαδικτύου για επιχειρηματικές διαδικασίες. Οι τυπικές μηχανές αναζήτησης, οργανώνουν και δημιουργούν ευρετήρια με τις διευθύνσεις των ιστοσελίδων, ωστόσο, σε ένα business analyst δεν αρκεί σαν πληροφορία μόνο μια απλή διεύθυνση ιστοσελίδας. Ένας Business Analyst μπορεί να περιηγηθεί σε ένα ή περισσότερα UDDI μητρώα για να δει τις διάφορες επιχειρήσεις που εκθέτουν τα web service καθώς και τις προδιαγραφές των υπηρεσιών αυτών. Οι προγραμματιστές χρησιμοποιούν το UDDI Programmer's API για δημοσιεύσουν web service και για να κάνουν ερωτήσεις στην υπηρεσία καταλόγου για να ανακαλύψουν web service τα οποία καλύπτουν διάφορα κριτήρια.

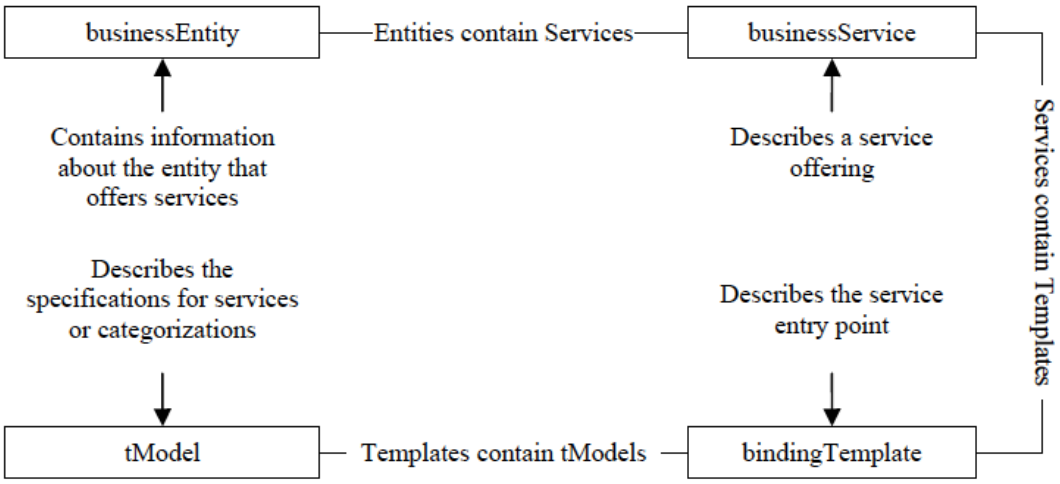
Ο παρακάτω πίνακας περιγράφει περιληπτικά τις προσφερόμενες υπηρεσίες του UDDI καθώς και κάποιες από τις βασικές δομές δεδομένων όπως αυτές περιγράφονται στο πρότυπο. Επίσης μπορεί να φανεί και η σχέση που έχουν μεταξύ τους οι βασικές αυτές δομές δεδομένων στο σχήμα 15.

Πανεπιστήριο Πειραιώς

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 (Πηγή: Vasudevan, 2001)

Πανεπιστήμιος

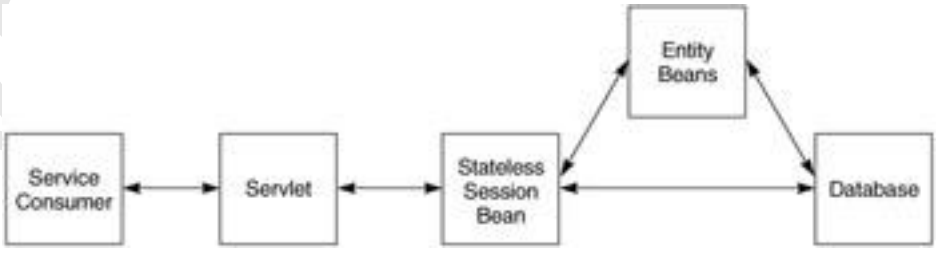


Σχήμα 15: Μέθοδοι UDDI (Πηγή: McGovern, Tyagi, Stevens E, & Mathew, 2003)

2.2.5 Σενάρια υλοποίησης

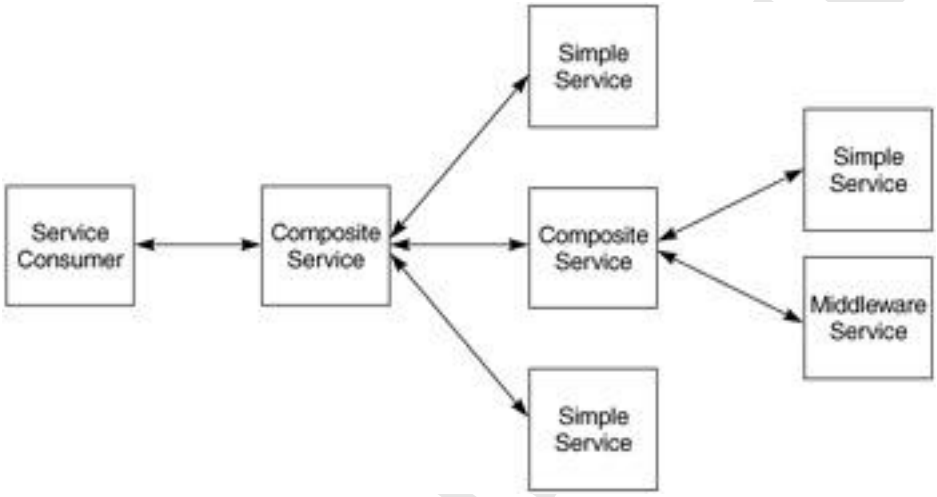
Τα σενάρια εφαρμογής της τεχνολογίας των Web services επιλέγονται ανάλογα με τις απαιτήσεις του περιβάλλοντος για το οποίο θα αναπτυχθούν. Οι ανάγκες κάθε οργανισμού δεν είναι ίδιες, επίσης κάθε οργανισμός δεν έχει την ίδια τεχνολογική υποδομή, τα σενάρια υλοποίησης λοιπόν θα είναι προσαρμοσμένα στις εκάστοτε απαιτήσεις του οργανισμού. Τα σενάρια μπορούν να κατηγοριοποιηθούν σε τέσσερις βασικούς τύπους υλοποίησης: απλής υπηρεσίας, σύνθετης υπηρεσίας, ενδιάμεσης υπηρεσίας και τέλος ενός διαδρόμου υπηρεσιών. Αναλύονται παρακάτω και βασίζονται στις λειτουργίες και τεχνολογίες της Java.

Απλή υπηρεσία: Είναι ένα service-enabled στοιχείο (ή servlet) που έχει άμεση πρόσβαση σε μια βάση δεδομένων ή σε άλλους πόρους. Με απλά λόγια είναι ένα web service το οποίο εξωτερικεύει – διαθέτει μέσω του Διαδικτύου μια εσωτερική απλή διαδικασία-λειτουργία του οργανισμού. Οι απλές υπηρεσίες είναι η συνηθέστερη επιλογή για την ανάπτυξη μικρών, διακριτών λειτουργιών. Μια απλή υπηρεσία είναι σαν αυτή που φαίνεται στο σχήμα 16**,** ο οργανισμός έχει ένα Stateless Session Bean που εκτελεί μια λειτουργία του οργανισμού έχοντας πρόσβαση σε μια βάση δεδομένων, η λύση είναι να αναπτυχθεί ένα Servlet και να μετατρέψει αυτήν την υπηρεσία του οργανισμού σε Web service.



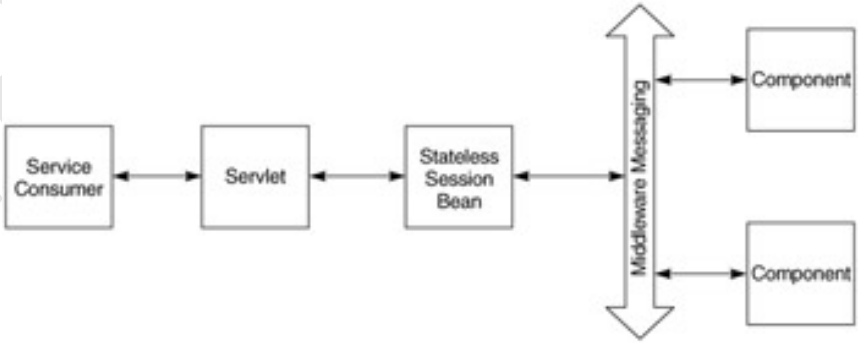
Σχήμα 16: Σενάριο απλού Web service (Πηγή: McGovern, et al., 2003)

Σύνθετη υπηρεσία: Είναι μια υπηρεσία η οποία συνθέτει σε ένα Web service τις λειτουργίες των απλών υπηρεσιών, άλλων σύνθετων υπηρεσιών και άλλων ενδιάμεσων υπηρεσιών. Ο πελάτης μπορεί να χρησιμοποιήσει της λειτουργίες όλων υπηρεσιών ξεχωριστά, ή μπορεί να χρησιμοποιήσει την έξοδο του συνδυασμού όλων των λειτουργιών των υπηρεσιών. Το τελικό αποτέλεσμα/τα που λαμβάνει ο πελάτης εξαρτώνται από τις εκάστοτε απαιτήσεις και την υλοποίηση του Web service. Ένα παράδειγμα μιας σύνθετης υπηρεσίας φαίνεται στο σχήμα 17.



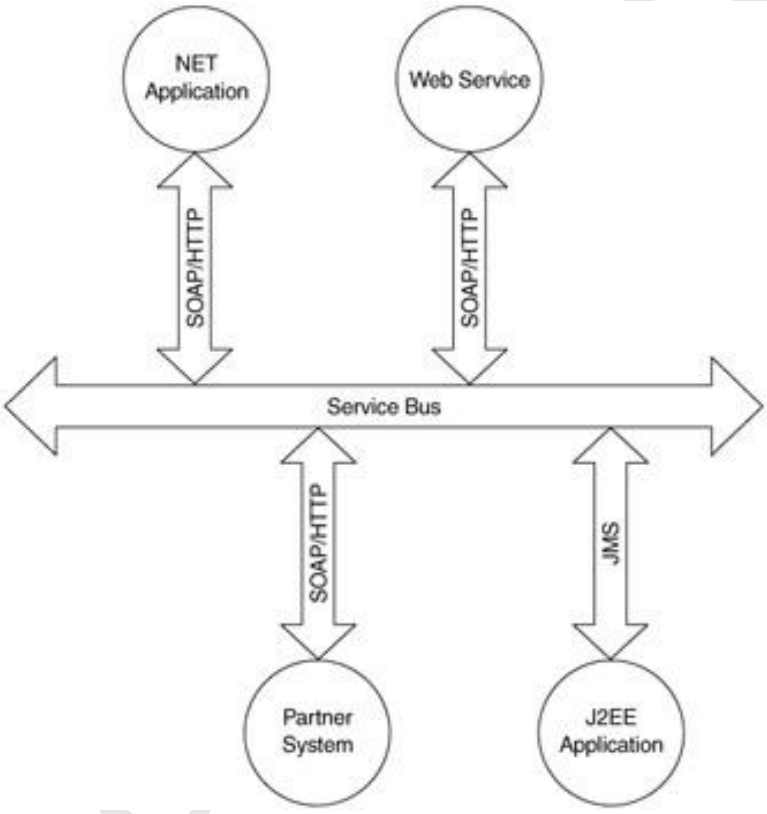
Σχήμα 17: Σενάριο σύνθετου Web service (Πηγή: McGovern, et al., 2003)

Ενδιάμεση υπηρεσία: Ένα web service το οποίο τοποθετεί τις αιτήσεις στον ενδιάμεσο διάδρομο είναι ένα παράδειγμα υλοποίησης μιας ενδιάμεσης υπηρεσίας. Πολλοί οργανισμοί έχουν εγκατεστημένα ενδιάμεσα συστήματα προσανατολισμένα σε μηνύματα(Messaging Oriented Middleware(MOM)). Ένα MOM σύστημα παρέχει μια έμμεση σύνδεση μεταξύ του αποστολέα ενός μηνύματος και του παραλήπτη του μηνύματος αυτού. Ο αποστολέας τοποθετεί ένα μήνυμα στην ουρά αναμονής, η οποία είναι αντιστοιχισμένη σε ένα τελικό σημείο. Ο αποστολέας δεν είναι ενήμερος για την φυσική τοποθεσία ή την υλοποίηση του τελικού αυτού σημείου. Ένα παράδειγμα μιας σύνθετης υπηρεσίας φαίνεται στο σχήμα 18.



Σχήμα 18: Σενάριο ενδιάμεσου Web service(Πηγή: McGovern, et al., 2003)

Διάδρομος υπηρεσιών: Ο διάδρομος υπηρεσιών είναι παρόμοιος με μια λύση Enterprise Application Integration(EAI). Επιτρέπει στα Web services να επικοινωνήσουν μεταξύ του μέσω τρίτων. Αποτελεί μια μέθοδος επικοινωνίας με πολυδιανομή (multicast) όπου ένας αιτών μπορεί να στείλει αιτήσεις σε πολλές υπηρεσίες ταυτόχρονα αλλά δεν είναι εν γνώση του ποίες άλλες υπηρεσίες καταναλώνουν την αίτηση του. Το λογισμικό διαχείρισης του διάδρομου υπηρεσιών είναι αυτός που δρομολογεί αυτά τα μηνύματα. Ο διάδρομος υπηρεσιών επίσης παρέχει και άλλες δυνατότητες όπως είναι η μετατροπή της μορφής των αιτήσεων από μια σε μια άλλη. Ένα παράδειγμα του διαδρόμου υπηρεσιών φαίνεται στο σχήμα 19.



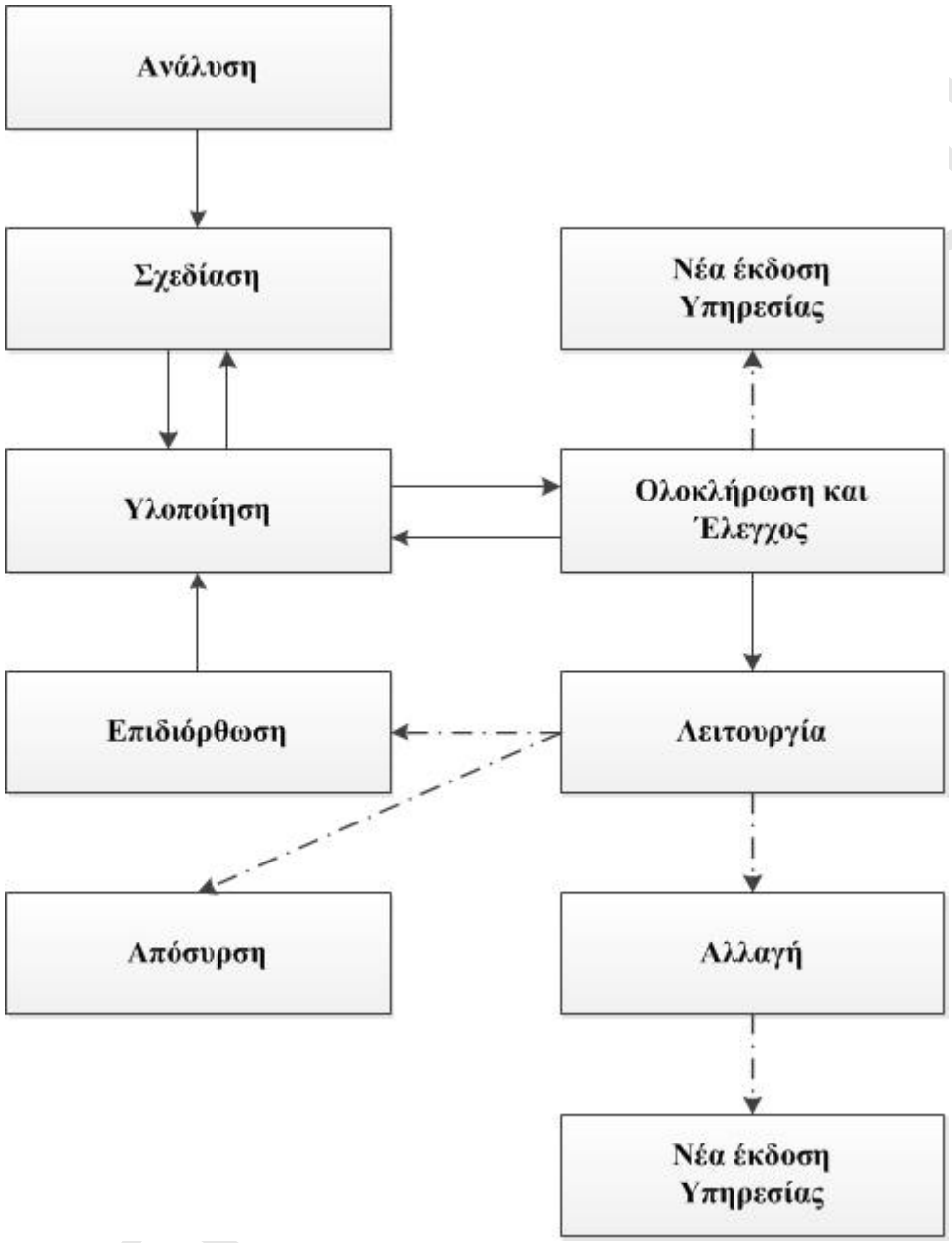
Σχήμα 19: Σενάριο διαδρόμου υπηρεσιών (Πηγή: McGovern, et al., 2003)

2.2.6 Κύκλος Ζωής

Για τα Web service δεν έχει καθιερωθεί κάποιος κοινά αποδεκτός κύκλος ζωής παρόλο που έχουν προταθεί αρκετοί στην βιβλιογραφία. Ένας ολοκληρωμένος κύκλος ζωής παρουσιάζεται από του Θεμιστοκλέους και Ματζάνα (2010) και αποτελείται από εννιά στάδια. Τα στάδια αυτά είναι τα εξής:

- 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.

Πανεπιστήμιο Πειραιώς



Σχήμα 20: Κύκλος ζωής Web service (Πηγή: Θεμιστοκλέους & Μαντζάνα, 2010)

2.2.7 Πλεονεκτήματα

Το να βασιστεί μια ανάπτυξη SOA στα Web services σαν μέσο για την εφαρμογή της, της προσφέρει κάποια σημαντικά οφέλη. Τα οφέλη των Web service είναι τα εξής:

- Είναι αυτόνομα: Από την στιγμή πού ένα WS έχει γίνει deploy και είναι έτοιμο προς εκτέλεση, ο πελάτης μπορεί να το καταναλώσει χωρίς την ανάγκη οποιασδήποτε εγκατάστασης λογισμικού στο μηχάνημα του καθώς ένας

πελάτης HTTP είναι συνήθως προεγκατεστημένος στα περισσότερα λειτουργικά συστήματα.

- Είναι αυτοπεριγραφόμενα: μια διεπαφή μπορεί να δημοσιοποιηθεί μέσω ενός WSDL εγγράφου. Ο πελάτης και ο εξυπηρετητής χρειάζεται μόνο να αναγνωρίσουν το περιεχόμενο και την δομή των μηνυμάτων αίτησης και απόκρισης. Η ορισμός της δομής ενός μηνύματος ταξιδεύει μαζί με το μήνυμα.
- Είναι αφαιρετικά: Απλά Web Service μπορούν να συγκεντρώνονται για να σχηματίσουν πιο σύνθετα Web Service για να προσφέρουν στον πελάτη μια πιο καλύτερη-ολοκληρωμένη υπηρεσία. Ο σχηματισμός αυτών τον σύνθετων Web Service προσφέρει την δυνατότητα στον πελάτη να δει μια επιχειρησιακή υπηρεσία πιο αφαιρετικά.
- Είναι προσβάσιμα μέσω του Διαδικτύου: Τα Web Service δημοσιεύονται, εντοπίζονται και μπορούν να κληθούν μέσω του Διαδικτύου, επίσης χρησιμοποιούν τα προτυποποιημένα πρωτόκολλα του.
- Είναι ανεξάρτητα: Τα περισσότερα Web Service σήμερα βασίζονται στην XML που είναι μία γλώσσα ανεξάρτητη πλατφορμών. Επίσης μια εφαρμογή πελάτη μπορεί γραμμένη με οποιαδήποτε γλώσσα προγραμματισμού μπορεί να έχει πρόσβαση σε ένα Web Service γραμμένο αντίστοιχα από οποιαδήποτε γλώσσα προγραμματισμού, αυτό τα καθιστά ανεξάρτητα γλωσσών προγραμματισμού. Και τέλος μπορούν να κληθούν από οποιοδήποτε προτυποποιημένο πρωτόκολλο δικτύου (συνήθως το HTTP), αυτό τα καθιστά ανεξάρτητα πρωτοκόλλων
- Είναι ανοικτά και βασίζονται σε πρότυπα: Η τεχνολογία των Web Service βασίζεται σε ανοιχτά πρότυπα, καθιστώντας με αυτόν τον τρόπο εύκολη την διαλειτουργικότητα μεταξύ των Web Service.
- Είναι δυναμικά: Τα Web Service μπορεί να ανακαλυφθούν και να καταναλωθούν στο χρόνο εκτέλεσης χωρίς να χρειάζεται οποιαδήποτε γνώση για τον χρόνο μεταγλώττισης τους, όπως συμβαίνει στην πλειοψηφία των άλλων τεχνολογιών.
- Μπορούν συντεθούν: Τα Web Service μπορούν να συγκεντρωθούν και να συνθέσουν μια μεγαλύτερη υπηρεσία. Για την σύνθεση αυτής της μεγαλύτερης υπηρεσίας μπορεί να χρησιμοποιηθούν οι διαδικασίες ενορχήστρωσης και χορογραφίας υπηρεσιών.
- Επιτρέπουν γρήγορη ένταξη: Ο πελάτης ενός Web service μπορεί από την στιγμή που αναζητήσει και εντοπίσει ένα Web service που καλύπτει τις απαιτήσεις του, να το καλέσει και να το εντάξει στο σύστημα του.

2.2.8 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:

Πανεπιστήμιο Πειραιώς

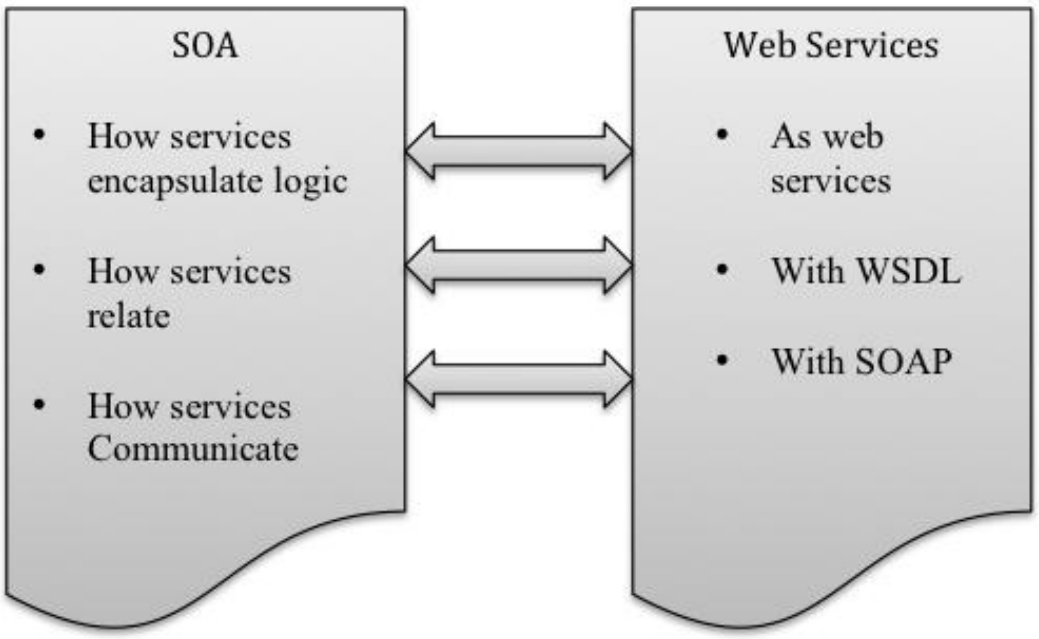
- Η δυναμική σύνδεση: Ένα Web service απαιτεί τα περιεχόμενα ενός μητρώο υπηρεσιών να είναι έμπιστα. Προς το παρόν μόνο τα ιδιωτικά μητρώα υπηρεσιών μπορούν να έχουν τέτοιο έλεγχο στα περιεχόμενα τους.
- Η ανταπόκριση στις απαιτήσεις: Κάθε φορά που δημιουργείται μια γενική υπηρεσία για να διαχειριστεί 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 παραμέτρους των 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 η προσθήκη νέων μεθόδων ή η αλλαγή της εσωτερικής λογικής των web services 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 συναλλαγές και άλλα, τα οποία προσφέρονται από άλλες προσεγγίσεις εδώ και αρκετά χρόνια. Τα web service προσπαθούν να καλύψουν το σημαντικό κενό αυτό με την εισαγωγή πρόσθετων προδιαγραφών (WS-*).

Πανεπιστήμιο Πειραιώς

2.2.9 Web Services 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, περιορίζονται στο στρώμα των υπηρεσιών σχήμα 5 και σχήμα 21. Υπάρχουν όμως και άλλα σημαντικά στρώματα στην αρχιτεκτονική της SOA που πρέπει να δοθεί εξίσου μεγάλο βάρος και σε αυτά 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 με τα 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.



Σχήμα 21:Web Services και SOA

2.3 Χορογραφία και ενορχήστρωση υπηρεσιών

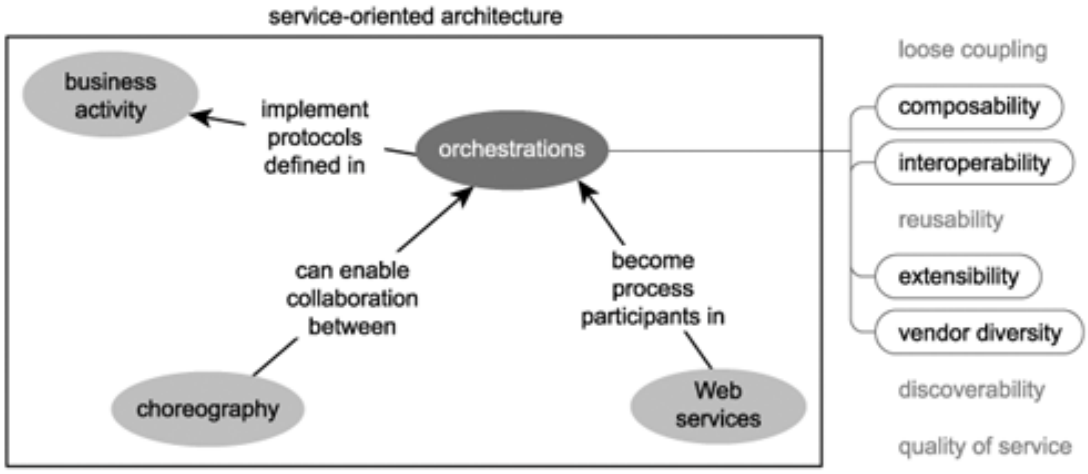
Ο συνδυασμός Web services για την δημιουργία υψηλού επιπέδου δια-οργανικών επιχειρησιακών διαδικασιών χρειάζεται πρότυπα τα οποία θα μοντελοποιούν τις αλληλεπιδράσεις. Οι προσεγγίσεις της χορογραφίας (choreography) και της ενορχήστρωσης (orchestration) υπηρεσιών ανήκουν στο στρώμα σύνθεσης επιχειρησιακών διαδικασιών της αρχιτεκτονικής SOA (βλ. σχήμα 5) και είναι τα πρότυπα που μοντελοποιούν τις παραπάνω αλληλεπιδράσεις. Οι επόμενες ενότητες αναλύουν την χορογραφία και την ενορχήστρωση των υπηρεσιών με λεπτομέρεια

2.3.1 Ενορχήστρωση Υπηρεσιών

Για την ενορχήστρωση δεν υπάρχει κάποιος κοινά αποδεκτός ορισμός, ο όρος ενορχήστρωση σύμφωνα με τον Peltz (2003) είναι ο τρόπος με τον οποίο τα web services αλληλεπιδρούν μεταξύ τους, εκτελώντας ένα μοντέλο διαδικασιών – μια επιχειρησιακή λογική. Επίσης αναφέρει πως οι αλληλεπιδράσεις αυτές μεταξύ των web services μπορεί είτε να εκτείνονται ανάμεσα σε οργανισμούς , είτε ανάμεσα σε εφαρμογές. Ο Erl (2005) αναφέρει στον ορισμό που παραθέτει για την ενορχήστρωση, πως επιτρέπει στη επιχειρησιακή λογική να εκφραστεί μέσα από υπηρεσίες. Ο Josuttis (2007) στον δικό του ορισμό αναφέρει πως η ενορχήστρωση υπηρεσιών ονομαζέται η σύνθεση νέων υπηρεσιών από ήδη υπάρχουσες υπηρεσίες. Ο Peltz (2003) όπως και ο Josuttis (2007) εστιάζουν και οι δύο στην σημασιολογία της λέξης ενορχήστρωση αναφέρουν πως με την ενορχήστρωση μια διαδικασία ελέγχεται πάντα από τη προοπτική του ενός από τα συμβαλλόμενα μέρη, π.χ σε μία ορχήστρα το ρόλο αυτό αναλαμβάνει ο μαέστρος οποίος μέσω της καθοδήγησης και του συνδυασμού διαφορετικών μουσικών οργάνων οδηγεί την ορχήστρα στην εκτέλεση πιο σύνθετων διαδικασιών πιθανώς από αυτές που θα εκτελούσε από μόνο ένα όργανο. Το ίδιο ακριβώς που συμβαίνει με τα ενορχηστρωμένα όργανα μπορεί να συμβεί και με τις ενορχηστρωμένες υπηρεσίες. Από τα παραπάνω εξαγώγιμο συμπέρασμα είναι ότι η ενορχήστρωση των υπηρεσιών μπορεί να βοηθήσει το IT να ευθυγραμμιστεί με την επιχειρησιακή λογική του οργανισμού ακόμα και αν αυτή εκτείνεται εκτός των συνόρων του.

Στο τεχνολογικό κομμάτι ένα μοντέλο χορογραφίας περιγράφει τις ενέργειες επικοινωνίας και τις εσωτερικές ενέργειες στις οποίες εμπλέκεται μια υπηρεσία. Οι εσωτερικές ενέργειες περιλαμβάνουν μετασχηματισμούς των δεδομένων καθώς και κλήσεις στις εσωτερικές μονάδες ενός λογισμικού. Επίσης μπορεί να περιέχει ενέργειες επικοινωνίας ή εξαρτήσεις μεταξύ των ενεργειών επικοινωνίας οι οποίες εμφανίζονται σε οποιαδήποτε διεπαφή Web service και προέρχονται από τρίτα συνεργαζόμενα μέρη. Οι ενορχηστρώσεις επίσης καλούνται “εκτελέσιμες διαδικασίες” από την στιγμή που πρόκειται να εκτελεστούν από μια μηχανή ενορχήστρωσης. Οι μηχανές ενορχήστρωσης μπορούν να ενσωματωθούν πλήρως στα υπηρεσιοστραφή περιβάλλοντα καθιστώντας τις κατάλληλες για την SOA αρχιτεκτονική. Στο σχήμα 22 φαίνονται η αλληλεπιδράσεις της ενορχήστρωσης με άλλα στοιχεία της SOA αρχιτεκτονικής καθώς και ποία είναι τα πλεονεκτήματα (τονισμένα με κύκλο στα δεξιά του σχήματος 22) που απορρέουν από την χρήσης της.

Πανεπιστήμιο Πειραιώς



Σχήμα 22: Η αλληλεπίδραση της ενορχήστρωσης με άλλα στοιχεία της SOA (Πηγή: Erl, 2005)

Κάποιοι ισχυρίζονται πως η ενορχήστρωση είναι αυτή που προσδίδει στα Web services την στρατηγική τους αξία. Τα Web services έχουν εγγενή αξία γιατί είναι εύκολα υλοποιήσιμα από τους προγραμματιστές και μπορούν να χρησιμοποιηθούν για να εξαλείψουν οποιοδήποτε πρόβλημα διαλειτουργικότητας μεταξύ διαφορετικών τύπων λογισμικού/συστημάτων. Η ενορχήστρωση είναι αυτή που καλείται από κάποιον οργανισμό για να δοθεί λύση στα προβλήματα που προκύπτουν από την διαχείριση των επιχειρηματικών διαδικασιών, ενδεικτικά προβλήματα είναι οι ασύμβατοι τύποι δεδομένων, το ταίριασμα σημασιολογικών δεδομένων κτλ.

Η γλώσσα/πρότυπο μέσω της οποίας επιτυγχάνεται η ενορχήστρωση βασίζεται στην XML και είναι η BPEL από το OASIS. Η BPEL επικράτησε μεταξύ άλλων προτύπων που αναπτύχθηκαν για αυτό τον σκοπό. Αναλύεται με λεπτομέρεια στην ενότητα 2.5.

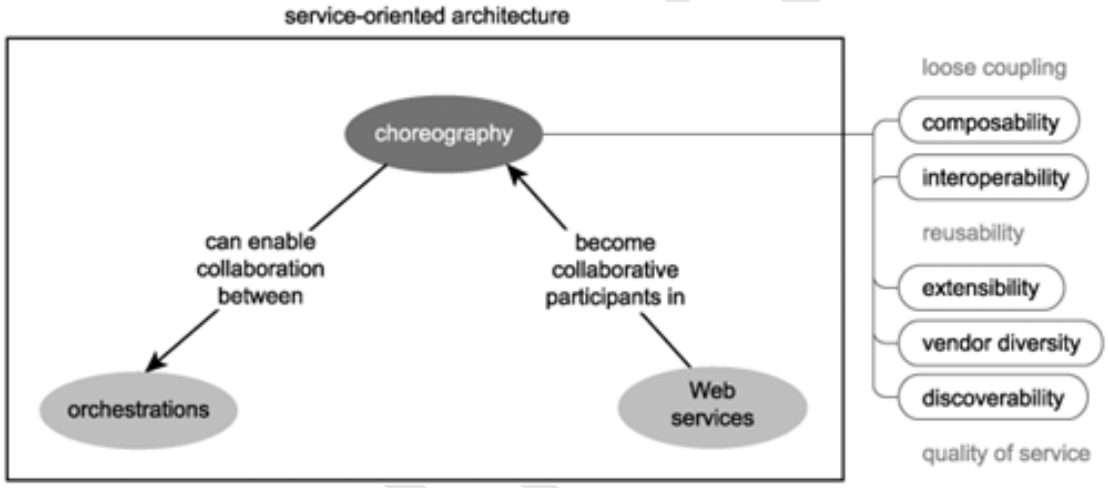
2.3.2 Χορογραφία υπηρεσιών

Όπως και για την ενορχήστρωση έτσι και με την χορογραφία δεν υπάρχει κάποιος κοινά αποδεκτός ορισμός. Ο όρος χορογραφία σύμφωνα με τον Peltz (2003) και τον Jossutis (2007) είναι ο τρόπος με τον οποίο παρακολουθείται μια ακολουθία μηνυμάτων μεταξύ των διαφορετικών μερών που συνεργάζονται για τη εκτέλεση μιας διαδικασίας. Κάθε μέρος της διαδικασίας είναι υπεύθυνο για ένα ή περισσότερα βήματα της και οι ανταλλαγές των μηνυμάτων μεταξύ των Web services είναι συνήθως δημόσιες. Ιδιοκτήτης της διαδικασίας συνήθως δεν είναι κάποιο από τα εμπλεκόμενα μέρη και μπορεί κάποιο από αυτά να μην γνωρίζει ή να μην κατανοεί την διαδικασία στο σύνολο της. Από τα παραπάνω προκύπτει ότι η χορογραφία στηρίζεται στην συνεργασία των διαφορετικών μερών.

Ένα μοντέλο χορογραφίας περιγράφει την συνεργασία μεταξύ μιας συλλογής υπηρεσιών για την επίτευξη ενός κοινού στόχου. Συλλέγει σφαιρικά τις επικοινωνίες που εμπλέκουν οι συμμετέχουσες υπηρεσίες για την επίτευξη αυτού του κοινού στόχου και της εξαρτήσεις μεταξύ αυτών των αλληλεπιδράσεων συμπεριλαμβανομένων των αιτιωδών εξαρτήσεων και/ή των εξαρτήσεων ελέγχου ροής(π.χ μια δεδομένη αλληλεπίδραση πρέπει να συμβεί πριν μια άλλη, ή μια

αλληλεπίδραση είναι η αιτία για μία άλλη), των εξαρτήσεων αποκλεισμού (ότι η μια δεδομένη αλληλεπίδραση εξαιρεί ή αντικαθιστά μια άλλη), των εξαρτήσεων ροής δεδομένων, των αλληλεπιδράσεων συσχέτισης, τους χρονικούς περιορισμούς, των εξαρτήσεων συναλλαγών, κλπ.

Η χορογραφία, μπορεί να συμβάλει στην υλοποίηση της SOA εκτός των συνόρων ενός οργανισμού. Ενώ υποστηρίζει εγγενώς τις δυνατότητες της επαναχρησιμοποίησης και της επεκτασιμότητας, η χορογραφία μπορεί επίσης να αυξήσει την οργανωσιακή ευκινησία και τον οργανωσιακό εντοπισμό. Οι οργανισμοί είναι σε θέση να συμμετάσχουν σε πολλαπλές online συνεργασίες, που μπορούν να επεκτείνουν δυναμικά ή ακόμη και να τροποποιήσουν τις σχετικές επιχειρηματικές διαδικασίες που ενοποιούνται με τις χορογραφίες. Με το να είναι σε θέση να περνούν πληροφορίες γύρω από κανάλια, οι συμμετέχουσες υπηρεσίες μπορεί να κάνουν γνωστή την ύπαρξη στους οργανισμούς που αλληλεπιδρούν για την ύπαρξη των άλλων οργανισμών με τους οποίους είχαν ήδη επαφή.



Σχήμα 23: Η αλληλεπίδραση της χορογραφίας με άλλα στοιχεία της SOA (Πηγή: Erl, 2005)

Η γλώσσα/πρότυπο μέσω της οποίας θα επιτυγχάνεται η χορογραφία δεν έχει ακόμα πλήρως οριστεί, μια από τις επικρατέστερες είναι η WS-CDL(Web Services Choreography Description Language) από το W3C. Σε ακαδημαϊκό επίπεδο έχουν διατυπωθεί άλλες γλώσσες χορογραφίας όπως παράδειγμα η Let's Dance, η BPEL4Chor και η MAP.

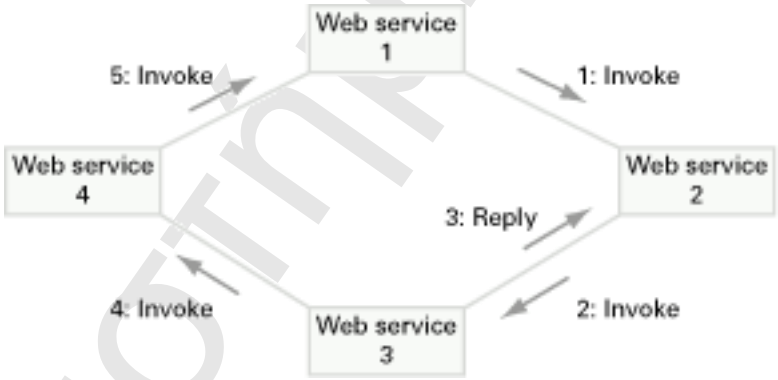
2.3.3 Σύγκριση ενορχήστρωσης και χορογραφίας

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 24).

Σχήμα 24:Σύνθεση υπηρεσιών με ενορχήστρωση (Πηγή: Juric M. B., n.d)

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 25).



Σχήμα 25: Σύνθεση υπηρεσιών με την ενορχήστρωση (Πηγή: Juric M. B., n.d)

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.

Πανεπιστήμιο Πειραιώς

2.4 Διαχείριση Επιχειρηματικών Διαδικασιών (Business Process Management)

Αυτή η ενότητα θα ξεκινούσε κανονικά με την διαχείριση επιχειρηματικών διαδικασιών(Business Process Management (BPM)) αλλά πριν τον ορισμό της διαχείρισης επιχειρηματικών διαδικασιών και την ανάλυση της, είναι καλό να δοθεί ο ορισμός της επιχειρηματικής διαδικασίας και κάποια στοιχεία για αυτή.

2.4.1 Ορισμός Επιχειρηματικής Διαδικασίας (Business Process)

Για την επιχειρηματική διαδικασία δεν υπάρχει κάποιος κοινά αποδεκτός ορισμός, Σύμφωνα με τον Davenport (1993) μια διαδικασία είναι ένα σύνολο δομημένων και μετρήσιμων δραστηριοτήτων που αποσκοπούν στην παραγωγή ενός καθορισμένου αποτελέσματος για έναν πελάτη. Σύμφωνα με τους Harrington (1991) και Hammer & Champy (1993), μια διαδικασία είναι κάθε δραστηριότητα ή ομάδα δραστηριοτήτων που λαμβάνει μια είσοδο, προσθέτει αξία σε αυτή και παρέχει μια έξοδο για ένα πελάτη.

Γενικό συμπέρασμα από τα παραπάνω είναι πως και οι τέσσερις συμφωνούν ότι μια επιχειρηματική διαδικασία είναι: (i) ένα σύνολο δραστηριοτήτων, (ii)έχουν είσοδο (iii) και η έξοδος τους έχει προστιθέμενη αξία. Είναι σημαντικό για μια επιχειρηματική διαδικασία να είναι σωστά σχεδιασμένη για να παρέχει ανταγωνιστικό πλεονέκτημα και η έξοδος της να μπορεί να προσδώσει στον οργανισμό την μέγιστη προστιθέμενη αξία. Για αυτό τον σκοπό μια επιχειρηματική διαδικασία θα πρέπει συνεχώς να παρακολουθείται, να ανανεώνεται και να βελτιστοποιείται σύμφωνα με τις τρέχουσες ανάγκες της αγοράς και της επιχείρησης.

Υπάρχουν τέσσερα είδη επιχειρηματικών διαδικασιών:

- Απλές (single-function): Αφορούν συγκεκριμένες λειτουργίες που εκτελούνται από ένα ορισμένο τμήμα ενός οργανισμού
- Δια-τμηματικές (cross-functional): Διατρέχουν διάφορες λειτουργίες ή τμήματα ενός οργανισμού. Περισσότερα από 1 άτομα είναι υπεύθυνα για την εκτέλεση τους
- Κοινές στο πλαίσιο του οργανισμού (common across the enterprise)
- Δι-επιχειρησιακές (inter-organizational): Οι οποίες απεικονίζουν αμφίδρομη επικοινωνία ανάμεσα σε διαφορετικούς οργανισμούς

Από την παραπάνω κατηγοριοποίηση των επιχειρηματικών διαδικασιών γίνεται αντιληπτό ότι μια επιχειρηματική διαδικασία μπορεί να αποτελεί μέρος μιας μεγαλύτερης επιχειρηματικής διαδικασίας, οδηγώντας στο συμπέρασμα ότι μπορεί γίνει σύνθεση ή κατακερματισμός των επιχειρηματικών διαδικασιών.

2.4.2 Ορισμός BPM

Όπως και με τις επιχειρηματικές διαδικασίες έτσι και με τη διαχείριση επιχειρηματικών διαδικασιών δεν υπάρχει κάποιος κοινά αποδεκτός ορισμός. Στην

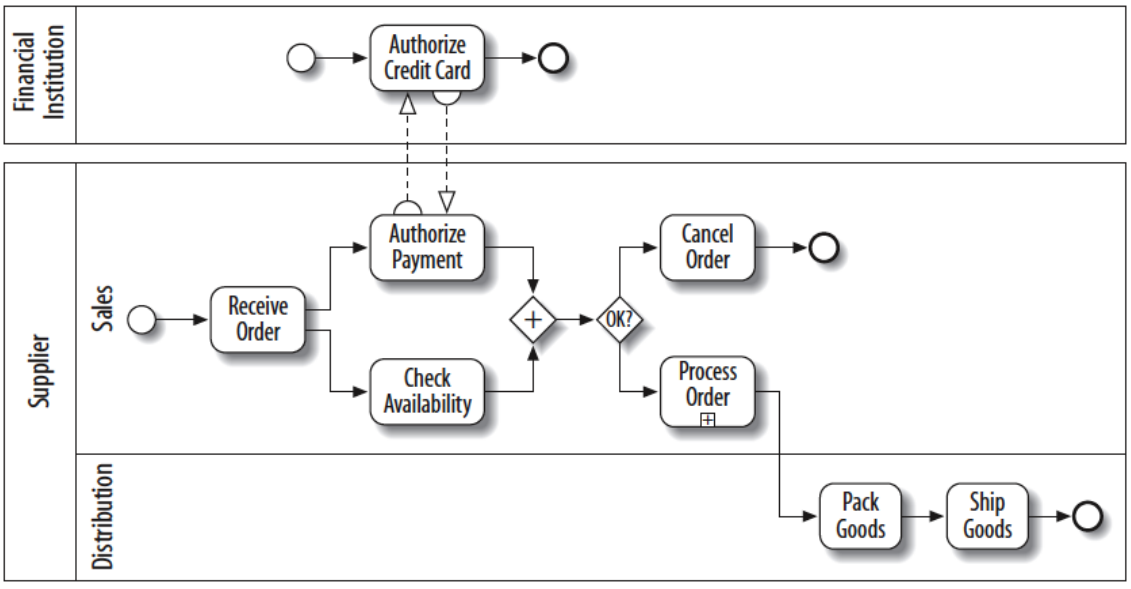
Πανεπιστήμιο Πειραιώς

παρούσα διπλωματική υιοθετείται ο ορισμός του Carter (2007) που σύμφωνα με αυτόν η διαχείριση επιχειρηματικών διαδικασιών είναι μια πειθαρχία που συνδυάζει τις δυνατότητες του λογισμικού και τις επιχειρηματικές δεξιότητες ατόμων, συστημάτων και πληροφοριών για να επιταχύνει το χρόνο των διαδικασιών βελτιστοποίησης, διευκολύνοντας κατά αυτόν το τρόπο την επιχειρησιακή καινοτομία.

2.4.3 Πρότυπα και Εργαλεία για την BPM

Τα πρότυπα τα οποία έχουν αναπτυχθεί για την υποστήριξη της διαχείρισης των επιχειρηματικών διαδικασιών είναι πολλά, αλλά τα πιο σημαντικά είναι:

- Η **BPMN** (Business Process Modeling Notation): Παρέχει σε μια επιχείρηση την δυνατότητα κατανόησης των εσωτερικών της επιχειρηματικών διαδικασιών μέσω της γραφικής μοντελοποίησης και δίνει σε οργανισμούς την ικανότητα να επικοινωνήσουν με αυτές τις διαδικασίες με πρότυπο τρόπο. Επιπλέον, ο γραφικός συμβολισμός θα διευκολύνει την κατανόηση των επιδόσεων συνεργασίας και των επιχειρηματικών συναλλαγών μεταξύ των οργανισμών. Αυτό θα εξασφαλίσει ότι οι επιχειρήσεις θα αντιλαμβάνονται τον ίδιο τον εαυτό τους καθώς και τους συμμετέχοντες στην επιχείρηση τους και θα επιτρέπουν στους οργανισμούς να προσαρμοστούν γρήγορα στις νέες εσωτερικές και B2B επιχειρηματικές συνθήκες. Ένα παραδείγματα της BPMN παρουσιάζεται στο σχήμα 26.
- Και η **BPEL** (Business Process Execution Language): Ένα πρότυπο το οποίο εστιάζει στην εκτέλεση των επιχειρηματικών διαδικασιών καθώς και στην επικοινωνία μεταξύ συστημάτων και αναλύεται με μεγαλύτερη λεπτομέρεια στην ενότητα της 2.5 παρούσας διπλωματικής.

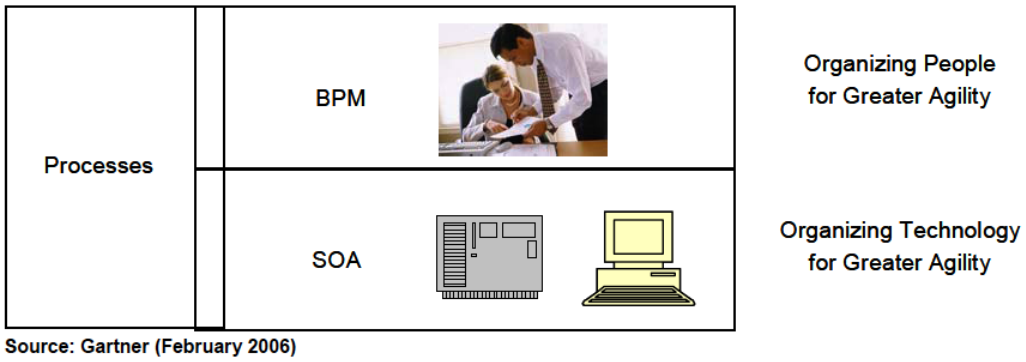


- Σχήμα 26:Παράδειγμα ενός BPMN διαγράμματος (Πηγή: Josuttis, 2007)

Μια συλλογή εργαλείων που είναι σημαντική για τη διαχείριση επιχειρηματικών διαδικασιών είναι το **BAM** (Business Activity Monitoring). Το BAM επιτρέπει την διαχείριση των συναθροίσεων (aggregations), των ειδοποιήσεων και των προφίλ για την παρακολούθηση των σχετικών με την επιχείρηση μετρήσεων (που ονομάζονται και αλλιώς βασικοί δείκτες επιδόσεων (Key Performance Indicators(KPI)). Επιτρέπει την ορατότητα από την μια άκρη των επιχειρηματικών διαδικασιών στην άλλη, παρέχοντας ακριβείς πληροφορίες σχετικά με την κατάσταση και τα αποτελέσματα των διαφόρων λειτουργιών, διαδικασιών και συναλλαγών ώστε να μπορούν, μέσα σε μια επιχείρηση να εξεταστούν οι προβληματικές περιοχές και να επιλυθούν τα όποια ζητήματα προκύψουν.

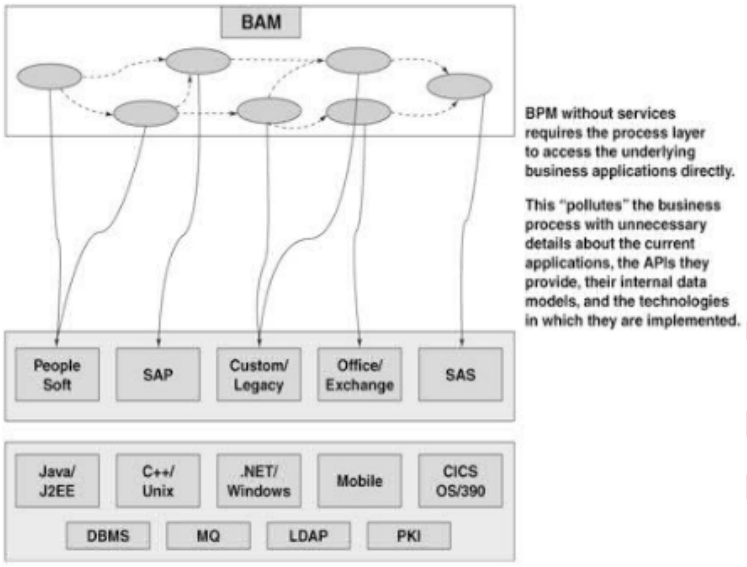
2.4.4 BPM και SOA

Σύμφωνα με όσα γράφουν Hill, et al., (2006) σε μία έρευνα του Gartner, η SOA και η BPM μοιράζονται τον ίδιο στόχο, που είναι η ευελιξία της επιχείρησης, η BPM διοργανώνει τους ανθρώπους για μεγαλύτερη ευελιξία, ενώ η SOA οργανώνει την τεχνολογία για μεγαλύτερη ευελιξία (βλ. σχήμα 27). Η SOA υποστηρίζει τον συντονισμό κατανεμημένων συστημάτων υποστηρίζοντας έτσι τις επιχειρηματικές διαδικασίες και δεν πρέπει να συγχέεται με αυτές.



Σχήμα 26: BPM και SOA (Πηγή: Hill, et al., 2006)

Η αρχή της SOA είναι οι υπηρεσίες, η SOA εκθέτει τις υπηρεσίες της προς κατανάλωση και το BPM το οποίο απαιτεί υπηρεσίες για την ολοκλήρωση/μοντελοποίηση μιας διαδικασίας τις καταναλώνει. Ο συνδυασμός SOA και BPM είναι ιδανικός. Κάθε διαδικασία μοντελοποιείται σαν μια σειρά από μικρά ανεξάρτητα μέρη. Αυτά τα μέρη συνήθως υλοποιούνται σαν υπηρεσίες. Η BPM βοηθάει στην δημιουργία μοντέλων διαδικασιών με την κλήση αυτών των υπηρεσιών.



Σχήμα 27: BPM χωρίς υπηρεσίες (Πηγή: Numnonda, 2009).

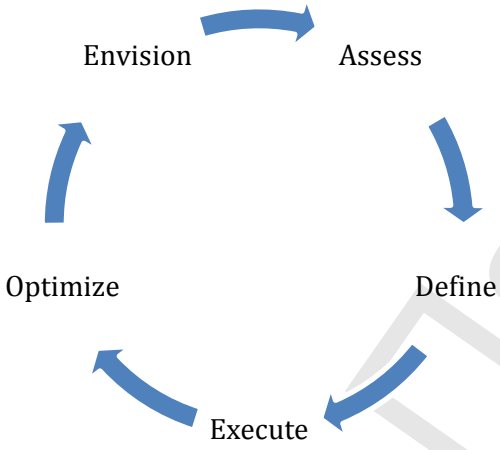
Όπως φαίνεται και από το σχήμα 28 μια BPM χωρίς την χρήση υπηρεσιών θα μπορούσε να "μολύνει" την επιχειρηματική διαδικασία με αχρείαστες λεπτομέρειες για τις τωρινές εφαρμογές, τα API που παρέχουν, τα εσωτερικά μοντέλα δεδομένων, και τις τεχνολογίες με τις οποίες έχουν υλοποιηθεί. Δηλαδή μια ισχυρής συνδεσιμότητας υλοποίηση.

Το στρώμα υπηρεσιών που λείπει από το σχήμα προσφέρει στο στρώμα επιχειρηματικών διαδικασιών τα εξής πλεονεκτήματα:

- Μια γραμμή με επιχειρηματικές υπηρεσίες παρέχει coarse-grain επιχειρηματική λειτουργικότητα (μπορεί δηλαδή να προσφέρει πολλές υπηρεσίες που σχετίζονται μεταξύ τους) που αντιστοιχίζεται στις επιχειρηματικές εργασίες μιας επιχειρηματικής διαδικασίας.
- Μια επιχειρηματική διαδικασία δεν ευθύνεται ότι δεν γνωρίζει τις λεπτομέρειες σχετικά με την υποκείμενη εφαρμογή και τις τεχνολογικές πλατφόρμες.
- Οι υπηρεσίες μητρώου και εντοπισμού που παρέχονται από το επίπεδο υπηρεσιών εξασφαλίζουν ότι το επίπεδο των επιχειρηματικών διαδικασιών μπορεί να εντοπίσει και να έχει πρόσβαση στις υπηρεσίες δυναμικά.
- Ένα μοντέλο δεδομένων του επιπέδου υπηρεσιών ορίζεται με βάση το επιχειρηματικό τομέα και είναι ανεξάρτητο από το μοντέλο δεδομένων που χρησιμοποιείται από οποιαδήποτε συγκεκριμένη εφαρμογή.
- Ένα μοντέλο ασφάλειας του επιπέδου-υπηρεσιών παρέχει single sign-on και έλεγχο πρόσβασης που βασίζονται σε ρόλους για να εξασφαλίσει ότι οι εργασίες επεξεργασίας είναι εξουσιοδοτημένες να χρησιμοποιούν τις υπηρεσίες.

2.4.5 Κύκλος Ζωής

Όπως φαίνεται από το παρακάτω σχήμα 29 ένας κύκλος ζωής για την διαχείριση των επιχειρηματικών διαδικασιών μπορεί να περιγραφεί από τα εξής στάδια:



Σχήμα 28: Ένας κύκλος ζωής της διαχείρισης των επιχειρηματικών δραστηριοτήτων.

- **Στάδιο Οραματισμού (Envision):** Σε αυτό το στάδιο οι στόχοι της επιχείρησης έχουν εμπεδωθεί και είναι καταγεγραμμένοι. Αναλύονται οι βασικοί δείκτες επιδόσεων των επιχειρηματικών στόχων και με τη συνδυασμένη γνώση των στόχων και των απαιτήσεων απόδοσης, αναπτύσσεται/συλλαμβάνεται ένα όραμα για την BPM λύση. Μπορεί να απαιτηθεί επίσης σε αυτό το στάδιο αλλαγή στην στρατηγική διαχείρισης.
- **Στάδιο Αξιολόγησης (Assess):** Αναλύεται η τωρινή κατάσταση του οργανισμού. Αξιολογούνται οι οργανωτικές ικανότητες για να διαπιστωθεί αν θα μπορέσουν να υιοθετήσουν τις νέες επιχειρηματικές διαδικασίες, οι οποίες μπορεί να διαπεράσουν τα σύνορα του οργανισμού και να απαιτήσουν την ισοπέδωση της οργανωσιακής δομής και ιεραρχίας. Επίσης αξιολογούνται οι υπάρχουσες επιχειρησιακές μετρήσεις και τα επιχειρησιακά μέτρα, καθώς και οι υπάρχουσες υποδομές του IT, η υπάρχουσα τεχνολογική υποδομή και η τεχνολογική στοίβα του οργανισμού. Αφού γίνουν όλες οι αξιολογήσεις για την επερχόμενη αλλαγή που περιγράφεται στο στάδιο του οραματισμού, ο κύκλος ζωή προχωράει στο επόμενο στάδιο.
- **Στάδιο Ορισμού (Define):** Είναι το μελλοντικό σταθερό στάδιο του οργανισμού, κατά το οποίο θα έχουν αναπτυχθεί (σχεδιασμός, υλοποίηση, εγκατάσταση και διαχείριση) οι επιχειρηματικές διαδικασίες. Σε αυτό το στάδιο ορίζονται οι απαιτήσεις σε όλα όσα αξιολογήθηκαν στο προηγούμενο στάδιο για να υποστηριχθούν οι επιχειρηματικές διαδικασίες. Επίσης γίνεται προσημείωση των διαδικασιών για τον εντοπισμό τυχόν σημείων συμφόρησης. Η διαδικασία διακυβέρνησης και το πλαίσιο τροποποιούνται και εξευγενίζονται για την υποστήριξη του σκοπού, των προτεραιοτήτων και της χρηματοδότησης. Τροποποιήσεις ορίζουν επίσης τα εισαγωγικά κριτήρια για την πιστοποίηση των διαδικασιών και την εφαρμογή τους, στο περιβάλλον

Πανεπιστήμιο Πειραιώς

του πελάτη, όπου θα βρίσκονται πρόσωπο με πρόσωπο με τους στόχους επιδόσεων.

- **Στάδιο Εκτέλεσης (Execute):** Ο υψηλού επιπέδου ορισμός της επιχειρηματικής και αρχιτεκτονικής και της αρχιτεκτονικής του IT καθώς και των συστατικών τους, έχει ουσιαστικά μοντελοποιηθεί, κατασκευαστεί, ενσωματωθεί, αναπτυχθεί και παρακολουθηθεί στους αντίστοιχους χρόνους εκτέλεσης τους. Σε γενικές γραμμές:
  - Οι επιχειρηματικές διαδικασίες είτε έχουν επανασχεδιαστεί είτε είναι σχεδιασμένες από την αρχή.
  - Ως ένα από τους μηχανισμούς για τον προσδιορισμό των υπηρεσιών χρησιμοποιούνται τα μοντέλα αποσύνθεσης διεργασιών.
  - Η διαδικασία έχει ενσωματωθεί και συνδεθεί με τη χρήση υπηρεσιών και άλλων στοιχείων του IT.
  - Η διαδικασία έχει αναπτυχθεί (deployed) σε μια μηχανή εκτέλεσης διαδικασιών και οι διεργασίες που εκτελούνται παρακολουθούνται για την επίδοση τους και για άλλες συμφωνίες του επίπεδου υπηρεσιών (Service Level Agreements (SLA)) καθώς και για άλλους βασικούς δείκτες επιδόσεων.

  Οποιαδήποτε στιγμή απαιτηθούν από αυτό το στάδιο εργαλεία και προϊόντα για την επιτυχή εκτέλεσή του, έχει εγκατασταθεί και ρυθμιστεί η τεχνολογική στοίβα, που υποστηρίζει κάθε φάση της διαδικασίας ανάπτυξης. Έχουν ξεκινήσει επίσης οι απαιτούμενες οργανωσιακές αλλαγές, όπως αυτές έχουν οριστεί στο προηγούμενο στάδιο. Κατά την διάρκεια αυτής της φάσης αρχικοποιείται και αναπτύσσεται η εφαρμογή του πλαισίου διακυβέρνησης διαδικασιών.

- **Στάδιο Βελτιστοποίησης (Optimize):** Κατά το στάδιο αυτό παρακολουθούνται, διαχειρίζονται και βελτιστοποιούνται οι διάφορες πτυχές της αρχιτεκτονικής της επιχείρησης, για να επιτευχθεί καλύτερη απόδοση και για να πληρούν τις μετρήσεις της επιχείρησης και του IT, οι οποίες χρησιμοποιούνται για να καθοριστεί η επιτυχία των εργασιών των επιχειρήσεων. Τα αποτελέσματα από τις εκτελέσιμες διαδικασίες συνήθως συγκεντρώνονται και αναλύονται. Η ανάλυση συνήθως αποκαλύπτει πληροφορίες οι οποίες ανατροφοδοτούν το στάδιο του οραματισμού, όπου οι στόχοι και οι προτεραιότητες των επιχειρήσεων μπορούν να επανεξεταστούν σε πραγματικό χρόνο βάση του τότε λειτουργικού περιβάλλοντος της επιχείρησης.

Χρειάζονται πρότυπα και εργαλεία για καλυφθούν οι απαιτήσεις σε όλα τα βήματα του κύκλου ζωής, μια λύση μπορεί να είναι ο συνδυασμός των BPMN, SOA και BAM.

Πανεπιστήμιο Πειραιώς

2.5 Business Process Execution Language (BPEL)

Η υιοθέτηση της αυτοματοποίησης των επιχειρηματικών διαδικασιών χρειάζεται κάποιο πρότυπο και κάποια γλώσσα τα οποία θα παρέχουν την δυνατότητα και τις κατευθύνσεις σε ένα οργανισμό να εκφράσει τις επιχειρηματικές διαδικασίες κατά ένα προτυποποιημένο τρόπο και με μία κοινά αποδεκτή γλώσσα. Αυτή η γλώσσα είναι η BPEL. Αφού αναλύθηκαν στις προηγούμενες ενότητες η SOA, τα Web services, η ενορχήστρωση των υπηρεσιών και η διαχείριση επιχειρηματικών διαδικασιών ήρθε η στιγμή της ανάλυσης της BPEL.

Η IBM, η Microsoft και η BEA το 2002, συνεργάστηκαν για την ανάπτυξη του πρώτου προτύπου της BPEL της BPEL4WS. Το BPEL4WS αντιπροσώπευε μια σύγκλιση των ιδεών που εμπεριέχονταν στις προδιαγραφές των XLANG και WSFL, οι οποίες προέρχονταν από την Microsoft και την IBM αντίστοιχα. Οι XLANG και WSFL αντικαταστάθηκαν σε εκείνο το σημείο από την προδιαγραφή BPEL4WS 1.0. Στην συνέχεια προστέθηκαν και συνέβαλαν στην ανάπτυξη της προδιαγραφής οι εταιρίες SAP και Siebel Systems και τον Απρίλιο του 2003 κατέθεσαν την έκδοση 1.1 της BPEL4WS στον OASIS για την προτυποποίηση της. Τον Απρίλιο του 2007 ο OASIS δημοσιοποίησε το πρότυπο WSBPEL 2.0 ή WS-BPEL 2.0 το οποίο αποτελεί το τελευταίο πρότυπο της BPEL.

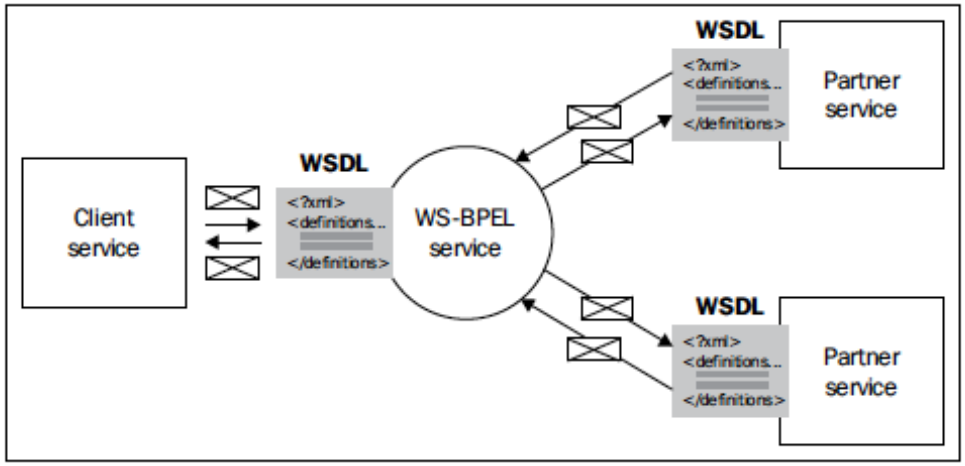
Από την τεχνολογική σκοπιά, η γλώσσα BPEL στηρίζεται στα πρότυπα XML, XPath και WSDL. Στα μηνύματα WSDL, η δομή και η σύνταξη της γλώσσας XML παρέχουν το μοντέλο δεδομένων που χρησιμοποιείται από την γλώσσα BPEL. Το πρότυπο XPath παρέχει υποστήριξη για την διερεύνηση των δεδομένων, ενώ όλοι οι εξωτερικοί συνεργάτες και οι δραστηριότητες αναπαρίστανται σαν υπηρεσίες WSDL (βλ. σχήμα 30**).**

Με την BPEL οι επιχειρηματικές διαδικασίες μπορούν να περιγραφούν με δύο τρόπους τον αφαιρετικό και τον εκτελέσιμο, συνοπτικά:

- Με το μοντέλο των εκτελέσιμων επιχειρηματικών διαδικασιών μπορεί να περιγραφεί με λεπτομέρειες, η πραγματική συμπεριφορά ενός συμμετέχοντος σε μια επιχειρηματική επικοινωνία. Και μπορεί να εκτελεστεί από μια μηχανή ενορχήστρωσης.
- Με το μοντέλο των αφαιρετικών επιχειρηματικών διαδικασιών μπορούν να περιγραφούν οι διαδικασίες που καθορίζουν την αμοιβαία δημόσια ανταλλαγή μηνυμάτων μεταξύ των συμμετεχόντων, χωρίς όμως να αποκαλυφθεί η εσωτερική συμπεριφορά τους. Οι αφαιρετικές διαδικασίας δεν προορίζονται να εκτελεστούν.

Η παρούσα διπλωματική θα ασχοληθεί μόνο με της εκτελέσιμες επιχειρηματικές διαδικασίες.

Πανεπιστήμιο Πειραιώς



Σχήμα:Οι αλληλεπιδράσεις της μηχανής BPEL (Πηγή: Vasiliev, 2007)

2.5.1 Χαρακτηριστικά

Κάποια από τα βασικά χαρακτηριστικά της BPEL όπως αυτά περιγράφονται από τους Papazoglou (2008) και Juric, Mathew, & Sarang (2006)είναι τα εξής:

- Βασίζεται στη XML αυτό την καθιστά, ανεξάρτητη πλατφορμών και κατασκευαστών
- Επιτρέπει την σύνθεση υπηρεσιών και περιγράφει την λογική των επιχειρηματικών διαδικασιών μέσω της αυτής της σύνθεσης
- Διαχειρίζεται σύγχρονες και ασύγχρονες λειτουργίες κλήσης σε υπηρεσίες και διαχειρίζεται τις απαντήσεις από τις κλήσεις αυτές έστω και αν αυτές επιστρέφουν σε μεταγενέστερα χρονικά διαστήματα.
- Επιτρέπει την μοντελοποίηση της συνεργασίας επιχειρηματικών διαδικασιών μέσω των συνδέσεων συνεργατών (partnerLinks).
- Επιτρέπει την μοντελοποίηση του ελέγχου εκτέλεσης μιας επιχειρηματικής διαδικασίας.
- Επιτρέπει τον προγραμματισμό δραστηριοτήτων βασιζόμενη στον χρόνο εκτέλεσης και ορίζει τη σειρά εκτέλεσης τους.
- Υποστηρίζει πλαίσια και μπορεί μέσω πολλών τέτοιων πλαισίων να δομήσει μια επιχειρηματική διαδικασία.
- Συσχετίζει αιτήσεις εντός και μεταξύ των επιχειρηματικών διαδικασιών
- Επιτρέπει την διαχείριση σφαλμάτων και παρέχει πλούσιο λεξιλόγιο σφαλμάτων.
- Διατηρεί πολλές δραστηριότητες συναλλαγών μακράς διαρκείας, οι οποίες μπορεί να είναι επίσης διακοπτόμενες.
- Μπορεί να επαναφέρει δραστηριότητες οι οποίες είτε είχαν διακοπεί είτε είχαν αποτύχει, ελαχιστοποιώντας με αυτό τον τρόπο την εργασία που χρειάζεται για να επαναληφθούν.
- Επιτρέπει την χειραγώγηση και των μετασχηματισμό των δεδομένων.

2.5.2 Ο Πυρήνας της BPEL

Μια διαδικασία BPEL αποτελείται από βήματα. Κάθε βήμα καλείται δραστηριότητα. Η BPEL υποστηρίζει βασικές δομημένες δραστηριότητες. Οι βασικές δραστηριότητες αντιπροσωπεύουν βασικές δομές οι οποίες χρησιμοποιούνται για συνήθεις εργασίες, όπως είναι αυτές που αναφέρονται παρακάτω:

- Αναμονή του πελάτη για την κλήση μιας επιχειρηματικής διαδικασίας μέσω της αποστολής μηνύματος χρησιμοποιώντας την <receive>
- Αναπαραγωγή μιας απάντησης για μια σύγχρονη λειτουργία, χρησιμοποιώντας την <reply>
- Κλήση άλλων Web services χρησιμοποιώντας την <invoke>
- Χειραγώγηση των μεταβλητών των δεδομένων χρησιμοποιώντας την <assign>
- Ένδειξη των λαθών και των εξαιρέσεων, χρησιμοποιώντας την <throw>
- Αναμονή για κάποιο χρονικό διάστημα, χρησιμοποιώντας την <wait>
- Τερματισμός ολόκληρης της διαδικασίας χρησιμοποιώντας την <terminate>

Αφού αναφέρθηκαν ποίες είναι οι βασικές δραστηριότητες της BPEL τώρα το επόμενο βήμα για κάποιον που θέλει να κατασκευάσει μια πιο σύνθετη επιχειρηματική διαδικασία, είναι να τις συνδυάσει. Η BPEL σαν γλώσσα υποστηρίζει αρκετές δομημένες δραστηριότητες για αυτόν τον σκοπό. Οι πιο σημαντικές είναι οι εξής:

- Η <process> η οποία περικλείει όλες τις υπόλοιπες δραστηριότητες της BPEL, και συνεπώς αποτελεί την ρίζα της BPEL.
- Η <sequence> για τον ορισμό ενός συνόλου δραστηριοτήτων οι οποίες θα κληθούν να εκτελεστούν σε σειρά ακολουθίας.
- Η <flow> για τον ορισμό ενός συνόλου δραστηριοτήτων οι οποίες θα κληθούν να εκτελεστούν παράλληλα.
- Η <switch> μια υπό συνθήκη δομή για την εκτέλεση κλαδιών.
- Η <while> για τον καθορισμό επαναλήψεων.
- Η <pick> η οποία δίνει την δυνατότητα της επιλογής μιας διαδρομής από ένα σύνολο εναλλακτικών διαδρομών.

Κάθε BPEL διαδικασία επίσης θα ορίσει μια σειρά από συνδέσμους συνεργατών χρησιμοποιώντας το <partnerLink> και θα δηλώσει τις μεταβλητές χρησιμοποιώντας το <variable>. Στον **πίνακα 3** παρουσιάζεται ένα απλοϊκό παράδειγμα κώδικα BPEL.

Στο **Παράρτημα Α** της παρούσας διπλωματικής περιέχονται με περισσότερη λεπτομέρεια οι προαναφερθείσες δραστηριότητες, καθώς και άλλα σημαντικά στοιχεία (elements) της γλώσσας BPEL.

Πανεπιστήμιο Πειραιώς

```
<process>
  …
  <partnerLinks>
  ...
  </partnerLinks>

  <variables>
  ...
  </variables>

  <sequence>
    …
    <receive>
    …
    </receive>

    <assign>
    …
    </assign>

    <invoke>
    …
    </invoke>

    <reply>
    …
    </reply>
   …
  </sequence>
 …
</process>
```

Πίνακας 3: Παράδειγμα κώδικα BPEL

2.5.3 Σύγχρονη και ασύγχρονη επικοινωνία

Μια διαδικασία BPEL μπορεί να κατηγοριοποιηθεί με βάση τον τρόπο επικοινωνίας που έχει επιλεγεί για κληθεί μια σύνδεση συνεργάτη. Υπάρχουν δυο τρόποι επικοινωνίας ο σύγχρονος και ο ασύγχρονος. Υπάρχει όμως και οι περιπτώσεις που μπορεί να χρησιμοποιηθούν και οι δύο τρόποι επικοινωνίας κατά την κλήση μιας σύνδεσης συνεργάτη καθώς η σύνδεση μπορεί να περιέχει πολλές μεθόδους/λειτουργίες, οπότε ανάλογα με την μέθοδο/λειτουργία να επιλέγεται και ο τρόπος επικοινωνίας ή απάντησης πιο συγκεκριμένα, όπως αναφέρεται στην τελευταία παράγραφο αυτής της ενότητας. Αναλυτικότερα:

Πανεπιστήμιο Πειραιώς

- Μια σύγχρονη διαδικασία BPEL επιστρέφει την απάντηση στον πελάτη άμεσα αφού έχει τελειώσει την επεξεργασία. Ο πελάτης καθ' όλη την διάρκεια την εκτέλεσης της διαδικασίας παραμένει σε κατάσταση αναμονής χωρίς να μπορεί να συνεχίσει την προγραμματιστική ροή της δικιάς του εφαρμογής.
- Από την άλλη πλευρά μια ασύγχρονη διαδικασία BPEL δεν θέτει τον πελάτη σε κατάσταση αναμονής και ο πελάτης μπορεί να συνεχίσεις την ροή της προγραμματιστικής διαδικασίας που αυτός εκτελεί. Για την επιστροφή της απάντησης στον πελάτη η διαδικασία BPEL χρησιμοποιεί ένα callback. Αξίζει να σημειωθεί ότι μια τέτοια διαδικασία BPEL δεν είναι υποχρεωμένη να επιστρέψει κάποια απάντηση.

Οι περισσότερες διαδικασίες που καλείται η BPEL να μοντελοποιήσει στην πραγματική ζωή είναι ασύγχρονες και χρειάζονται μεγάλα χρονικό διάστημα για να ολοκληρωθούν.

Η διαφορά μεταξύ σύγχρονης και ασύγχρονης BPEL διαδικασίας στο τεχνικό κομμάτι, έγκειται στον τρόπο απάντησης στον πελάτη της διαδικασίας. Μια σύγχρονη διαδικασία BPEL χρησιμοποιεί το <reply> για να αποστείλει στον πελάτη την απάντηση, ενώ μια ασύγχρονη διαδικασία BPEL χρησιμοποιεί το <invoke> για την αντίστοιχη λειτουργία.

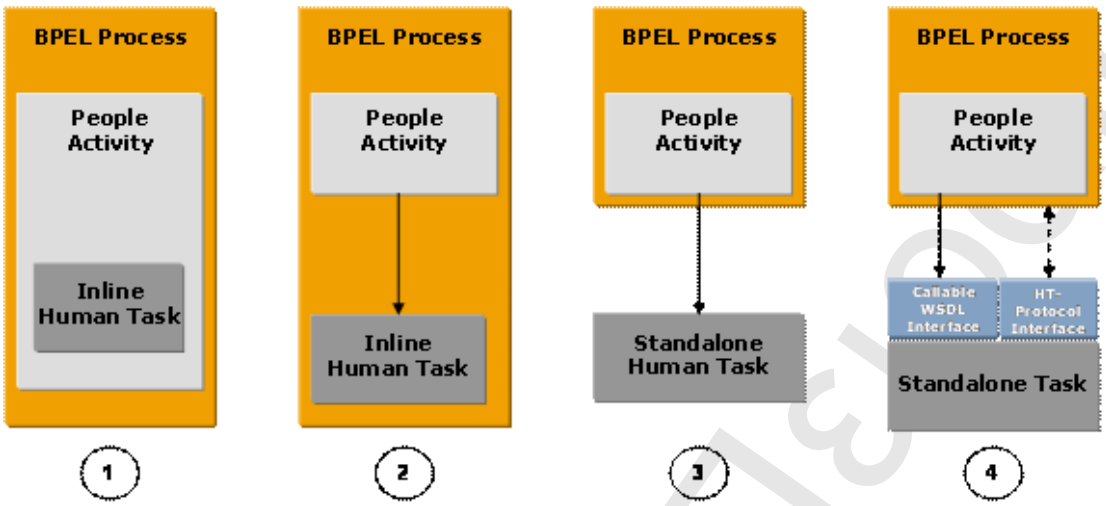
2.5.4 Business Process Execution Language for People (BPEL4People)

Η BPEL απευθυνόταν αμιγώς στην δημιουργία επιχειρηματικών διαδικασιών μεταξύ υπολογιστικών συστημάτων, έχοντας αυτή την μηχανο-κεντρική προσέγγιση απέκλειε την εισαγωγή ανθρώπινων/φυσικών δραστηριοτήτων μέσα στην δομή της επιχειρηματικής διαδικασίας. Οι επιχειρηματικές διαδικασίες όμως κάπου μέσα στην ροή τους μπορεί να χρειάζονται την ανθρώπινη παρέμβαση και αυτό αποτελούσε ένα σημαντικό μειονέκτημα της BPEL. Το μειονέκτημα αυτό αντιμετωπίστηκε κατά την προδιαγραφή μιας επέκτασης της, της BPEL4People (Business Process Execution Language for People).

Οι εταιρίες IBM και SAP από κοινού πρότειναν σε ένα white paper τον Ιούλιο του 2005 την BPEL4People και ακολούθησαν ύστερα από κοινού οι Active Endpoints, Adobe, BEA και Oracle την δημοσιοποίηση των προδιαγραφών WS-BPEL Extension for People (BPEL4People), Version 1.0 τον Ιούνιο του 2007 σαν συνέχεια του πρώτου whitepaper, περιγράφοντας πως μπορούν να εκτελεστούν οι ανθρώπινες αλληλεπιδράσεις και κατατέθηκαν στον OASIS για προτυποποίηση. Ο OASIS τον Αύγουστο του 2010 δημοσιοποίησε την WS-BPEL Extension for People (BPEL4People) Specification Version 1.1.

Στο τεχνολογικό κομμάτι στη προδιαγραφή του BPEL4People αναπτύχθηκαν οι δραστηριότητες ανθρώπων ως βασικές δραστηριότητες. Οι δραστηριότητες ανθρώπων χρησιμοποιούνται για την ενσωμάτωση των ανθρώπινων αλληλεπιδράσεων στο πλαίσιο των BPEL διαδικασιών. Το παρακάτω σχήμα 31 απεικονίζει διαφορετικούς τρόπους που θα μπορούσαν να ενταχθούν οι ανθρώπινες αλληλεπιδράσεις (συμπεριλαμβανομένων των ανθρώπινων εργασιών και ανακοινώσεων) η προδιαγραφή αναφέρεται σε αυτούς χρησιμοποιώντας την λέξη αστερισμός (constellation).

Πανεπιστήμιο Πειραιώς



Σχήμα 29:Οι ανθρώπινες δραστηριότητες της BPEL4People (Πηγή: OASIS, 2010)

Οι αστερισμοί 1 και 2 δείχνουν μοντέλα αλληλεπίδρασης στα οποία οι εργασίες είναι ορισμένες στο εσωτερικό, σαν να αποτελούν μέρος μιας διαδικασίας BPEL. Αναλυτικά:

1) Πρώτος αστερισμός: Μια εσωτερική εργασία μπορεί να οριστεί ως μέρος μιας δραστηριότητας ατόμων. Η χρήση στην περίπτωση αυτή περιορίζεται στη δραστηριότητα των ανθρώπων που θα περιλαμβάνονται.

2) Δεύτερος αστερισμός: Μια εργασία μπορεί να οριστεί ως μια ανώτερου επιπέδου δομή μιας διαδικασίας BPEL. Στην περίπτωση αυτή, η ίδια εργασία μπορεί να χρησιμοποιηθεί στο πλαίσιο περισσοτέρων από μίας ανθρώπινων δραστηριοτήτων, αποτελώντας έτσι ένα σημαντικό πλεονέκτημα αν κάποιος το δει από την οπτική της επαναχρησιμοποίησης. Οι διαδικασίες και οι ειδοποιήσεις της BPEL4People μέσω αυτού του αστερισμού καθίστανται φορητές μεταξύ των μηχανών εκτέλεσης BPEL4People.

3) Τρίτος αστερισμός: Εμφανίζει μια αυτόνομη εργασία μέσα στο ίδιο το περιβάλλον, χωρίς τις προδιαγραφές κλήσης σχετικών με την εργασία διεπαφών Web services. Έτσι, η κλήση της εργασίας είναι μια συγκεκριμένη και αυτόνομη εφαρμογή. Αυτός ο αστερισμός είναι παρόμοιος με στον αστερισμό 2, εκτός του το ότι ο ορισμός της εργασίας γίνεται ανεξάρτητα από οποιαδήποτε διαδικασία. Ως αποτέλεσμα, η εργασία και οι ειδοποιήσεις δεν έχουν άμεση πρόσβαση στο πλαίσιο της διαδικασίας.

4) Τέλος ο τέταρτος αστερισμός: Εμφανίζει τη χρήση μιας αυτόνομης εργασίας από ένα διαφορετικό περιβάλλον. Η βασική διαφορά σε σύγκριση με τον τρίτο αστερισμό είναι ότι η εργασία έχει μια διεπαφή Web service, η οποία μπορεί να κληθεί μέσω των πρωτοκόλλων των Web services. Επιπλέον, το πρωτόκολλο συντονισμού WS-HumanTask χρησιμοποιείται για την επικοινωνία μεταξύ διαδικασιών και εργασιών. Χρησιμοποιώντας αυτόν τον μηχανισμό η διαδικασία μπορεί να εκτελεί λειτουργίες του κύκλου ζωής πάνω στην εργασία. Οι

Πανεπιστήμιο Πειραιώς

διαδικασίες της BPEL4People που χρησιμοποιούν τις εργασίες με αυτόν τον τρόπο είναι φορητές μεταξύ των μηχανών BPEL4People.

2.5.5 Κύκλος ζωής διαδικασιών BPEL

Τα στάδια από το σχεδιασμό έως την εκτέλεση των διαδικασιών BPEL, περιλαμβάνουν τον ορισμό των διαδικασιών BPEL, τον ορισμό των δομών WSDL των διαδικασιών BPEL και τον ορισμό όλων των απαραιτήτων για την μηχανή BPEL αρχείων. Ο κύκλος ζωής μιας διαδικασίας BPEL από τον σχεδιασμό μέχρι την εκτέλεση αποτελείται από τα επόμενα επτά στάδια:

- Στο πρώτο στάδιο ο αναλυτής επιχειρήσεων είναι αυτός που καταγράφει και καθορίζει τη χρησιμοποιώντας την μέθοδο της μοντελοποίησης. Η μέθοδος αυτή μπορεί να χρησιμοποιεί τη σημειογραφία BPMN. Υπάρχουν εργαλεία σε αυτό το στάδιο που μέσω της BPMN μπορούν να παράγουν κομμάτια κώδικα BPEL.
- Στο δεύτερο στάδιο ο αναλυτής επιχειρήσεων ή ο υπεύθυνος ανάπτυξης λογισμικού μπορεί να προσδιορίζει τον επιπλέον κώδικα BPEL που πρέπει να αναπτυχθεί έτσι ώστε η διαδικασία να αναπαριστά με ακρίβεια την καθορισμένη διαδικασία του προηγούμενου επιπέδου.
- Στο τρίτο στάδιο ο κώδικας της διαδικασίας έχει αναπτυχθεί η διαδικασία σε αυτό το σημείο είναι έτοιμη να εφαρμοστεί σε μια μηχανή BPEL και να εκτελεστεί.
- Στο τέταρτο στάδιο η διαδικασία ενοποιείται με τους ορισμούς των στοιχείων WSDL εφαρμόζεται στη μηχανή εκτέλεσης BPEL, τα endpoinds δημιουργούνται και από αυτό το στάδιο και μετά η διαδικασία είναι έτοιμη προς κλήση.
- Στο πέμπτο στάδιο μια αίτηση του πελάτη προς την μηχανή εκτέλεσης ξεκινάει τη διαδικασία και τρέχει τη ροή ελέγχου. Μια επιχειρηματική διαδικασία στην BPEL είναι ο ορισμός ενός προτύπου που μπορεί να χρησιμοποιηθεί για την δημιουργία στιγμιότυπων αυτής της επιχειρηματικής διαδικασίας. Τα μηνύματα που αποστέλλονται σε μία διαδικασία είτε δημιουργούν ένα νέο στιγμιότυπο όταν φτάνουν στο σημείο εκκίνησης της διαδικασίας, είτε συνεχίζουν ένα ήδη υπάρχον στιγμιότυπο, με το οποίο είναι συνδεδεμένα βάση κάποιου αναγνωριστικού (αυτό επιτυγχάνεται μέσω του message correlation), όταν αυτά παραλαμβάνονται από κάποιο άλλο σημείο μέσα στην διαδικασία το οποίο δεν αποτελεί το σημείο εκκίνησης. Η μηχανή εκτέλεσης είναι υπεύθυνη για την αρχικοποίηση ενός στιγμιότυπου της διαδικασίας, ελέγχει κάθε πτυχή της και διαχειρίζεται τις συναλλαγές της με τις υπόλοιπες υπηρεσίες.
- Στο έκτο στάδιο η διαδικασία είναι πλέον εκτελεσμένη (επιτυχώς ή ανεπιτυχώς), σε αυτό το στάδιο απελευθερώνονται όλοι οι πόροι του συστήματος που έχει δεσμεύσει σε η μηχανή εκτέλεσης για αυτήν την διαδικασία.
- Στο έβδομο και τελευταίο στάδιο γίνεται εξαγωγή των δεδομένων και των πληροφοριών που σχετίζονται με την διαδικασία σε βάσεις δεδομένων της

Πανεπιστήμιο Πειραιώς

μηχανής εκτέλεσης, για μελλοντική χρήση καθώς και επισκόπηση των βημάτων της διαδικασίας.

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

3 Μεθοδολογία Έρευνας

Σε αυτό το κεφάλαιο παρουσιάζεται η θεωρία της μεθοδολογίας της έρευνας που στην παρούσα διπλωματική είναι η Action Reasearch και οι δύο προσεγγίσεις της συγκεκριμένης μεθοδολογίας.

Πανεπιστήμιο Πειραιώς

3.1 Action Research

Πρόκειται για μια μέθοδος επίλυσης προβλημάτων που έχει ως στόχο να συμβάλει στις πρακτικές ανησυχίες των ανθρώπων σε άμεσες, πραγματικές και προβληματικές καταστάσεις καθώς και στη συμβολή της επίτευξης των στόχων των κοινωνικών επιστημών καθώς στοχεύει στην επίλυση πραγματικών προβλημάτων.

3.2 Διαδικασίες Action Research

Για την Action Research μεθοδολογία υπάρχουν περισσότερες από μια διαφορετικές προσεγγίσεις. Δύο από τις κυριότερες προσεγγίσεις για την Action Research διαδικασία παρουσιάζονται παρακάτω.

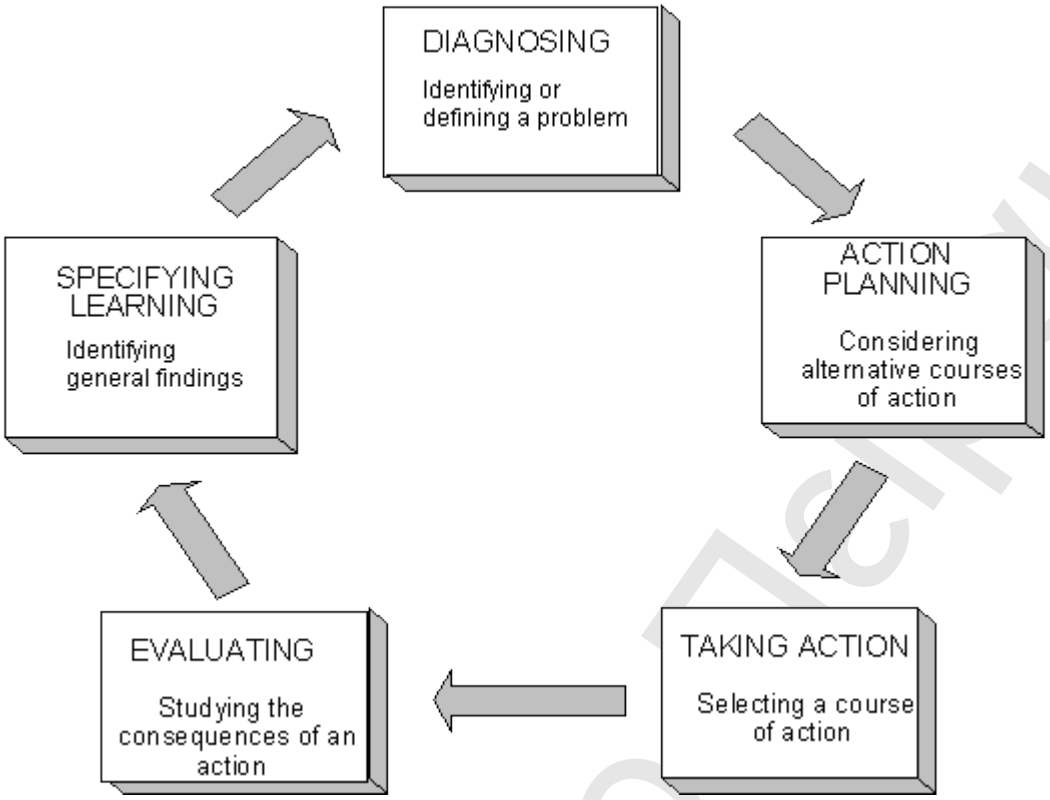
3.2.1 Διαδικασία Susman

Σύμφωνα με την προσέγγιση του Susman η διαδικασία του Action Research αποτελεί μια κυκλική διαδικασία η οποία περιέχει πέντε στάδια όπως φαίνεται και στο σχήμα 32. Τα στάδια αυτά είναι τα ακόλουθα:

- Το στάδιο της διάγνωσης (Diagnosing) όπου γίνεται ο προσδιορισμός του προβλήματος
- Το στάδιο του προσδιορισμού της λύσης (Action Planning) κατά το οποίο εξετάζονται εναλλακτικοί τρόποι δράσης.
- Το στάδιο της επιλογής του τρόπου δράσης (Taking Action) επιλέγοντας έναν από τη προηγούμενο στάδιο.
- Το στάδιο της αξιολόγησης (Evaluating) κατά το οποίο γίνεται αξιολόγηση των επιπτώσεων του επιλεγμένου τρόπου δράσης
- Το στάδιο του καθορισμού της μάθησης (Specifying Learning) κατά το οποίο εντοπίζονται οι όποιες γενικές διαπιστώσεις

Όντας κυκλική διαδικασία η διαδικασία του Susman επιτρέπει στα στάδια να επαναλαμβάνονται όσες φορές χρειαστεί μέχρι να δοθεί η βέλτιστη λύση στο πρόβλημα.

Πανεπιστήμιο Πειραιώς



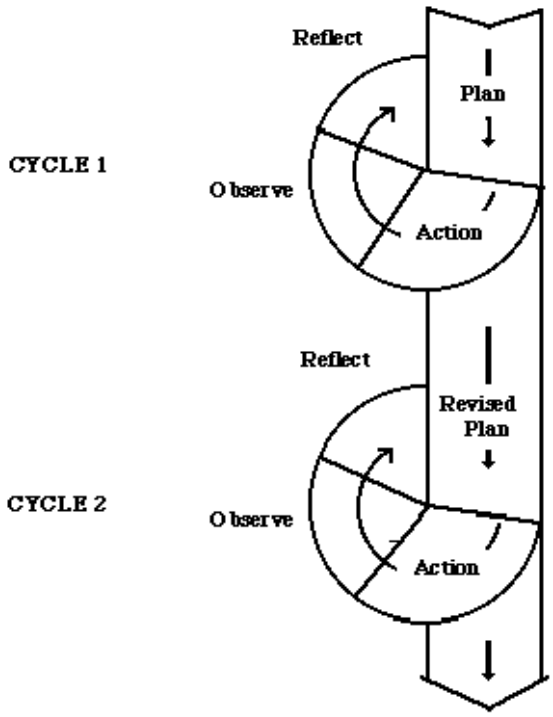
Σχήμα 30: Διαδικασία Action Research σύμφωνα με τον Susman (Πηγή: O'Brien, 1998)

3.2.1 Διαδικασία Kemmis

Ο Stephen Kemmis δίνει μια πιο απλή διαδικασία για την μεθοδολογία του Action Research. Η Διαδικασία που παρουσίασε ο Kemmis έχει και αυτή κυκλικό χαρακτήρα και κάθε κύκλος αποτελείται από τέσσερα στάδια. Τα τέσσερα στάδια είναι τα ακόλουθα:

- Το στάδιο του προσδιορισμού (Plan)
- Το στάδιο επιλογής δράσης (Action)
- Το στάδιο της παρατήρησης (Observe)
- Το στάδιο των επιπτώσεων (Reflect) βάση των όποιων δράσεων και παρατηρήσεων επιλέχθηκαν και καταγράφηκαν αντίστοιχα.

Όπως φαίνεται και από το σχήμα 33 για την επίλυση κάποιου προβλήματος στην διαδικασία που προτείνεται από τον Kemmis μπορεί να προστεθούν παραπάνω από έναν κύκλοι.



Σχήμα 31: Διαδικασία Action Research σύμφωνα με τον Kemmis (Πηγή: O'Brien, 1998)

4 Υλοποίηση

Σε αυτό το κεφάλαιο γίνεται ανάλυση της εφαρμογής που αναπτύχθηκε. Αναλύονται και αναφέρονται η αρχιτεκτονική της, οι γλώσσες, οι εφαρμογές ανάπτυξης και η βάση δεδομένων που χρησιμοποιήθηκαν για την υλοποίηση της επιχειρηματικής διαδικασίας και γίνεται παρουσίαση της εφαρμογής που αναπτύχθηκε.

Πανεπιστήμιο Πειραιώς

4.1 Λογισμικό

Για την υλοποίηση και τον σχεδιασμό της αρχιτεκτονικής SOA, της BPEL, των Web Services και των επιχειρηματικών διαδικασιών έχουν αναπτυχθεί ένα σύνολο εργαλείων από διαφορετικούς κατασκευαστές.

Ενδεικτικά κάποιοι ικανοί εξυπηρετητές για την SOA αρχιτεκτονική είναι ο WebSphere της IBM, Ο JBoss της Redhat, ο NetWeaver της SAP, ο Weblogic της Oracle, και ο BizTalk της Microsoft και από Open Source ο GlassFish της Oracle. Οι προαναφερθέντες εξυπηρετητές μπορούν με την εγκατάσταση και παραμετροποίηση κατάλληλων plugins και επεκτάσεων να εκτελέσουν το πλήθος των τεχνολογιών που εμπεριέχονται στην SOA αρχιτεκτονική. Από τους παραπάνω εξυπηρετητές επιλέχθηκαν ο Weblogic τον οποίο αξιοποιεί η SOA Suite 11g της Oracle η οποία εμπεριέχει τις τεχνολογίες της SOA και BPM και ο GlassFish Server Open Source Edition που μπορεί να φιλοξενήσει Web Services υλοποιημένα με την αντικειμενοστραφή γλώσσα προγραμματισμού Java.

Πανεπιστήμιο Πειραιώς

Για την ανάπτυξη της παρούσας διπλωματικής τα εργαλεία ανάπτυξης IDE (Integrated development Environment) που χρησιμοποιήθηκαν ήταν το NetBeans και το JDeveloper της Oracle. Το NetBeans χρησιμοποιήθηκε για την ανάπτυξη των Web Services με την χρήση την γλώσσας προγραμματισμού Java, και το JDeveloper για την ανάπτυξη των διαδικασιών BPEL και BPEL4People/BPM.

NetBeans

Οι βάσεις δεδομένων που χρησιμοποιήθηκαν για την υποστήριξη της υλοποίησης της παρούσας διπλωματικής ήταν η Open Source Mysql καθώς και η Oracle Express Edition Database. Για την ανάπτυξη των βάσεων δεδομένων χρησιμοποιήθηκαν τα εργαλεία Mysql Query Browser, και Mysql WorkBench.

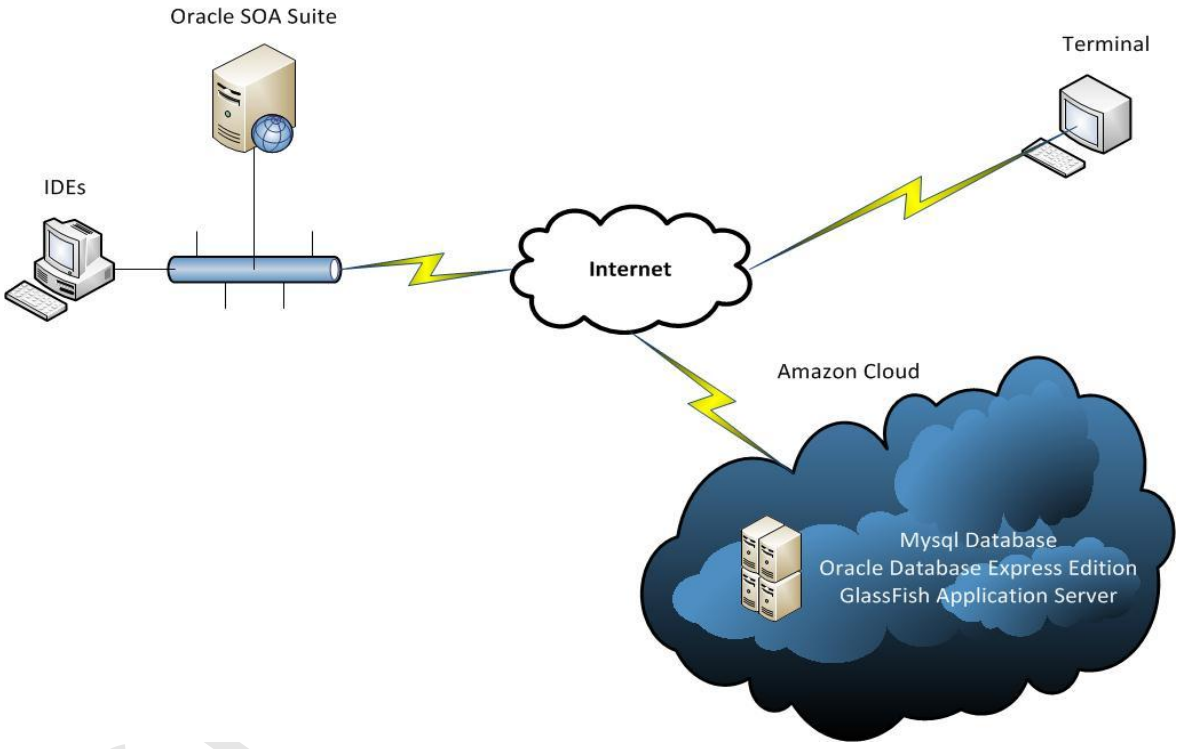
Η επιλογή των εξυπηρετητών, των εργαλείων ανάπτυξης, και των βάσεων δεδομένων έναντι άλλων, έγινε με γνώμονα την ευκολία εύρεσης τους, την υποστήριξη τους, το πλήθος τεχνολογιών/λύσεων που καλύπτουν και αφορούν την SOA αρχιτεκτονική και την ευκολία εκμάθησης και χρήσης και εγκατάστασης τους.

Πανεπιστήμιο Πειραιώς

4.2 Υλικό και τοπολογία

Για την υλοποίηση της εφαρμογής από οι απαιτήσεις σε υλικό ήταν τρείς υπολογιστές και αυτό γιατί τα εργαλεία της Oracle SOA Suite 11g, και JDeveloper ήταν ιδιαίτερα απαιτητικά σε μνήμη καθώς και σε επεξεργαστική ισχύ. Η τρείς υπολογιστές είχαν λειτουργικό σύστημα της Microsoft, η τοπολογία των υπολογιστών και τα εργαλεία/εφαρμογές που έτρεχαν όπως φαίνονται και στο σχήμα 34 είναι οι εξής:

- Ένας εξυπηρετητής στο cloud της Amazon Web Services στον οποίον εκτελούνταν οι βάσεις δεδομένων Mysql, Oracle Database Express Edition, και ο GlassFish ο οποίος φιλοξενούσε τα Web Services της εφαρμογής.
- Ένας εξυπηρετητής στο γραφείο μου στον οποίον εκτελούνταν το Oracle SOA Suite 11g και φιλοξενούσε τις BPEL, BPEL4People και BPM διαδικασίες.
- Και ένας υπολογιστής στον οποίο εκτελούνταν τα εργαλεία JDeveloper, NetBeans, Mysql Query Browser και Mysql WorkBench.



Σχήμα 32: Τοπολογία και εφαρμογές ανά υπολογιστή/εξυπηρετητή

4.3 Ανάπτυξη εφαρμογής

Στην ανάπτυξη της εφαρμογής λήφθηκε υπόψη η προσέγγιση της SOA αρχιτεκτονικής και κύριος στόχος της είναι να αναδειχτούν κάποια από τα πλεονεκτήματα της, όπως είναι η επαναχρησιμοποίηση, η διαλειτουργικότητα, οι χαλαροί δεσμοί μεταξύ των διαφορετικών στοιχείων της εφαρμογής κ.α.

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:

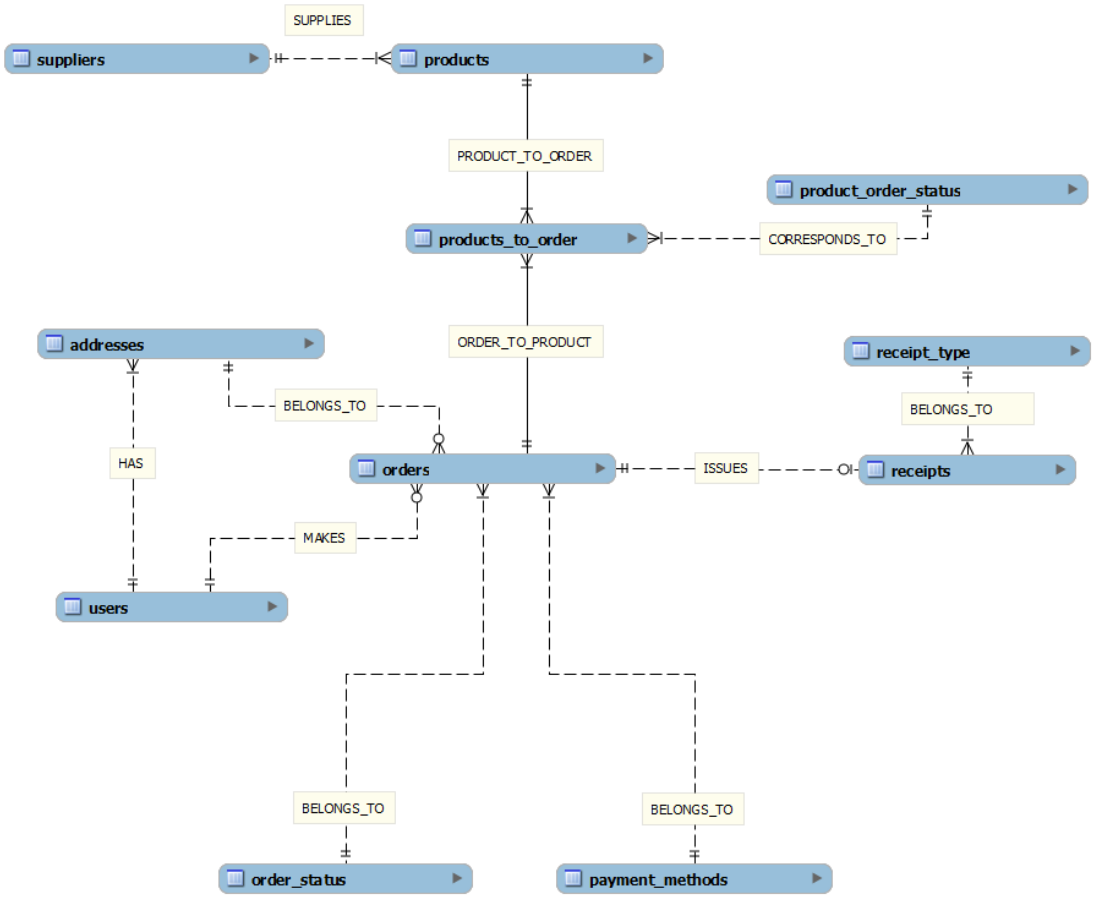
- Τα δεδομένα (Database)
- Τα απλά Web-services
- Τα σύνθετα Web Services (σύνθετες επιχειρηματικές διαδικασίες BPEL)
- Και το User Interface

Τα επίπεδα αναλύονται ένα-ένα στις επόμενες ενότητες.

4.3.1 Βάση δεδομένων

Σημαντικό στοιχείο της εφαρμογής που αναπτύχθηκε αποτελεί το πρώτο επίπεδο της, το επίπεδο των δεδομένων. Για την διαχείριση τους αναπτύχθηκε μια βάση δεδομένων, παρακάτω ακολουθεί η τεκμηρίωση της με τα διαγράμματα και τα μοντέλα.

Πανεπιστήμιο Πειραιώς

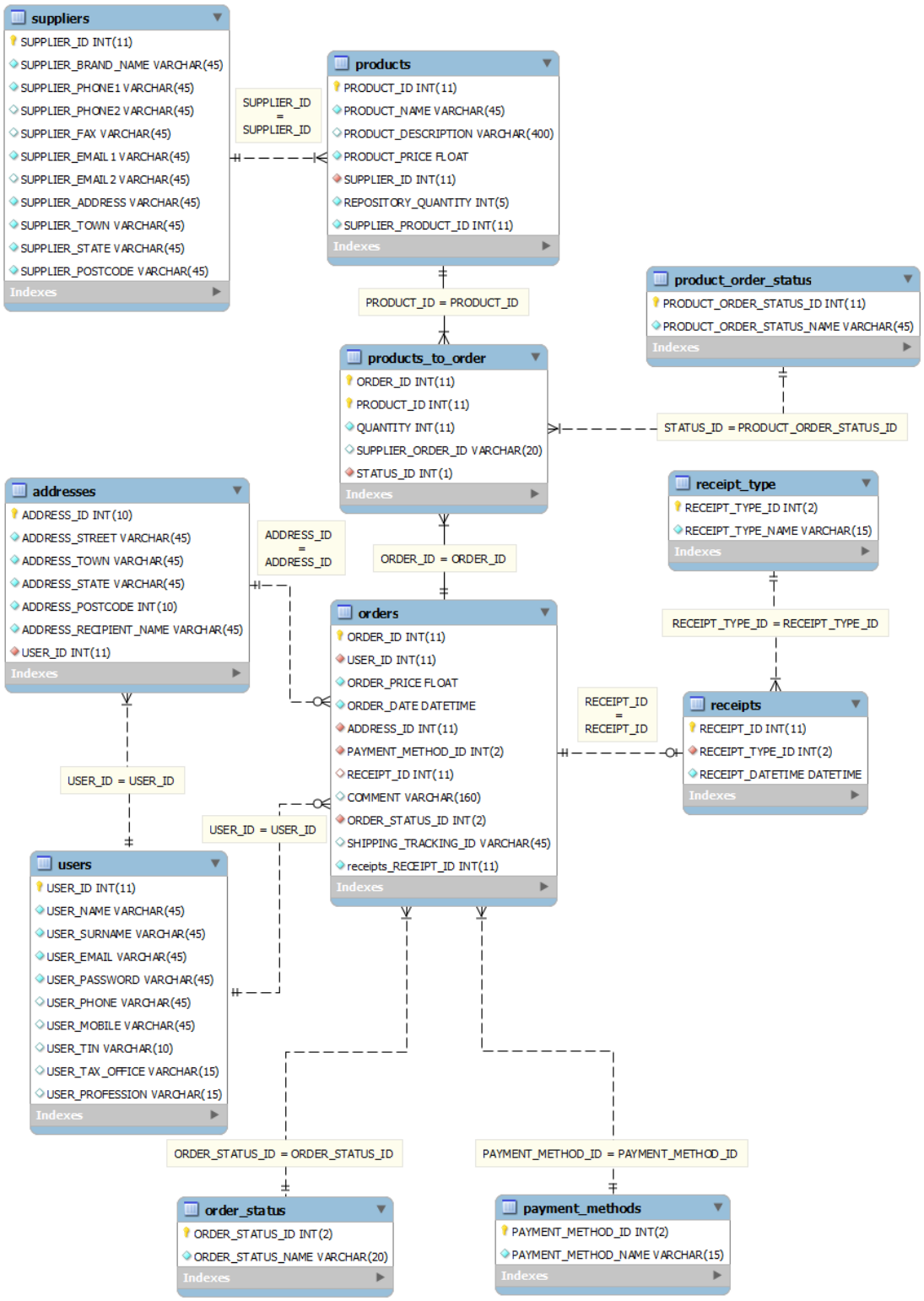


Entity - Relationship	Παραγγελία Προϊόντων	12/02/2012	V 1.0
(1/1)			

Παραγγελία Προϊόντων

- Κάθε πελάτης (USERS) μπορεί να πραγματοποιήσει μηδέν ή περισσότερες παραγγελίες (ORDERS) και μπορεί να έχει μια ή περισσότερες διευθύνσεις (ADDRESSES).
- Κάθε διεύθυνση (ADDRESS) μπορεί να αντιστοιχηθεί σε μία ή περισσότερες παραγγελίες (ORDERS).
- Κάθε προμηθευτής (SUPPLIER) μπορεί να προμηθεύει ένα ή περισσότερα προϊόντα.
- Κάθε παραγγελία (ORDER) μπορεί να έχει ένα ή περισσότερα προϊόντα (PRODUCTS) και κάθε προϊόν μπορεί να ανήκει σε μία ή περισσότερες παραγγελίες. Η αντιστοιχία αυτή καταχωρείται στο (PRODUCTS_TO_ORDER) .
- Κάθε προϊόν μιας παραγγελίας (PRODUCTS_TO_ORDER) πρέπει να έχει μια κατάσταση προϊόντος (PRODUCT_ORDER_STATUS) και κάθε κατάσταση προϊόντος μπορεί να ανήκει σε ένα ή περισσότερα προϊόντα της παραγγελίας.
- Για κάθε παραγγελία μπορεί να εκδοθούν μία ή καμία παραστατικό (RECEIPTS).
- Κάθε παραστατικό πρέπει να ανήκει σε κάποιο τύπο παραστατικού (RECEIPT_TYPE) και ένας τύπος παραστατικού μπορεί να αντιστοιχηθεί σε περισσότερες από μια αποδείξεις.
- Κάθε μέθοδος πληρωμή (PAYMENT_METHOD) μπορεί να αντιστοιχηθεί σε μία ή περισσότερες παραγγελίες και κάθε παραγγελία πρέπει να έχει μια μέθοδο πληρωμής
- Κάθε κατάσταση παραγγελίας (ORDER_STATUS) μπορεί να αντιστοιχηθεί σε μία ή περισσότερες παραγγελίες και κάθε παραγγελία πρέπει να έχει μια κατάσταση παραγγελίας.

Πανεπιστήμιο Πειραιώς

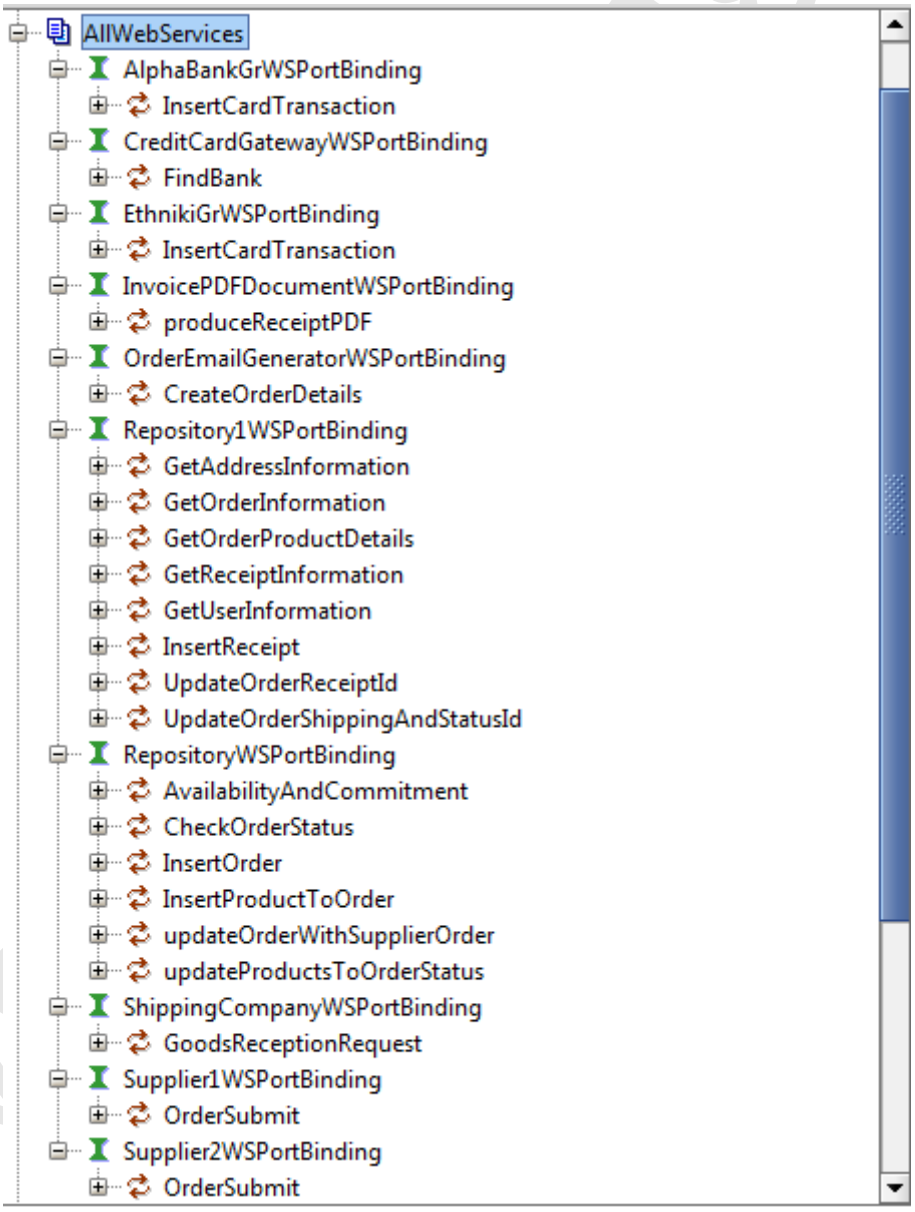


Relational Model	Παραγγελία Προϊόντων	12/02/2012	V 1.0
(1/1)			

4.3.2 Απλά Web services

Το επίπεδο των απλά Web Services αποτελείται από web services τα οποία εκτελούν απλές επιχειρηματικές διαδικασίες. Το επίπεδο αυτό είναι αυτό που έρχεται σε αλληλεπίδραση με την/τις βάση-εις δεδομένων. Στο επίπεδο αυτό έχουν υλοποιηθεί web services που εκτελούν μικρές και απλές μεθόδους, αρκετές από αυτές τις μεθόδους αλληλεπιδρούν με τις βάσεις δεδομένων. Στο επίπεδο αυτό οι συγκεκριμένες αυτές μέθοδοι (μονάδες κώδικα) επιτρέπουν σε εξωτερικά συστήματα να τις καλέσουν μέσω τον endpoint και να τις χρησιμοποιήσουν είτε αυτόνομα είτε να τις εντάξουν σε Coarse Web Services όπως θα περιγράφεται στην επόμενη ενότητα.

Για την παρούσα διπλωματική αναπτύχθηκαν αρκετές μέθοδοι η πλειονότητα των οποίων αφορούν αλληλεπιδράσεις με την βάση δεδομένων και δημοσιεύθηκαν με την μορφή των Web Services.



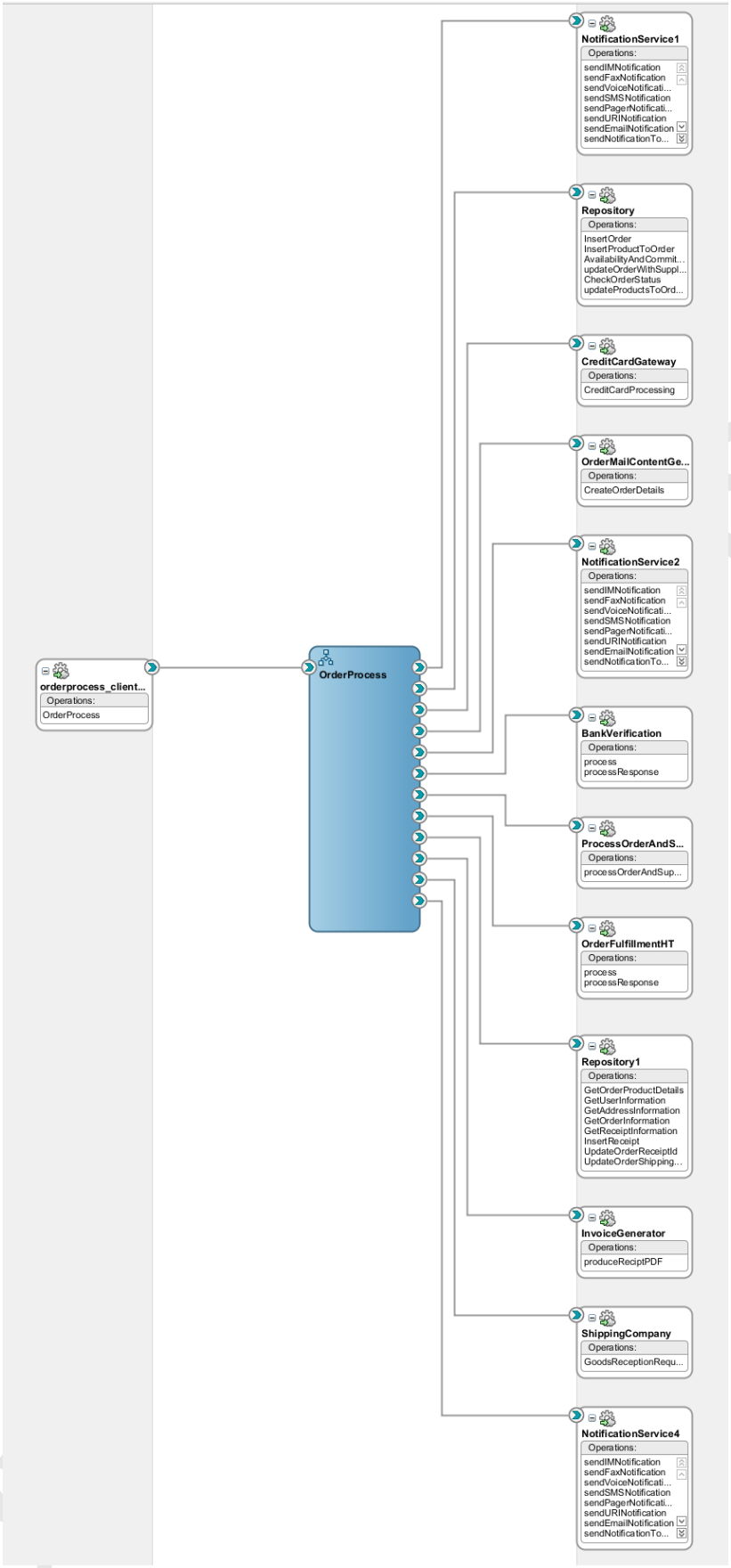
Σχήμα 33: Λίστα με τα Web Services που αναπτύχθηκαν

Τα Web Services που αναπτύχθηκαν μπορούν να φανούν και στο σχήμα 35 είναι τα εξής:

4.3.3 Σύνθετα Web Services (BPEL)

Τα σύνθετα Web Services είναι αυτά που συνδυάζουν τα απλά Web services του προηγούμενου επιπέδου και επιτρέπουν την μετατροπή των φυσικών επιχειρηματικών διαδικασιών σε ηλεκτρονικές επιχειρηματικές διαδικασίες με την χρήση της BPEL. Τέτοιου είδους Web services είναι αυτά που επιτρέπουν την ευθυγράμμιση της επιχείρησης με το IT της. Η κάθε διαδικασία BPEL δημοσιοποιείται σαν Web service και επιτρέπει στους πελάτες της να εκτελέσουν κάποια σύνθετη επιχειρηματική διαδικασία καλώντας τη. Η διαδικασία BPEL κατά την εκτέλεση των λογικών βημάτων της σύνθετης επιχειρηματικής διαδικασίας καλεί πολλά άλλα εξωτερικά components καθώς και Web services του προηγούμενου επιπέδου. Στο σχήμα 36 εμφανίζεται η διαδικασία παραγγελίας και τα Web services σύνθετα ή απλά που καλεί για την διεκπεραίωση της.

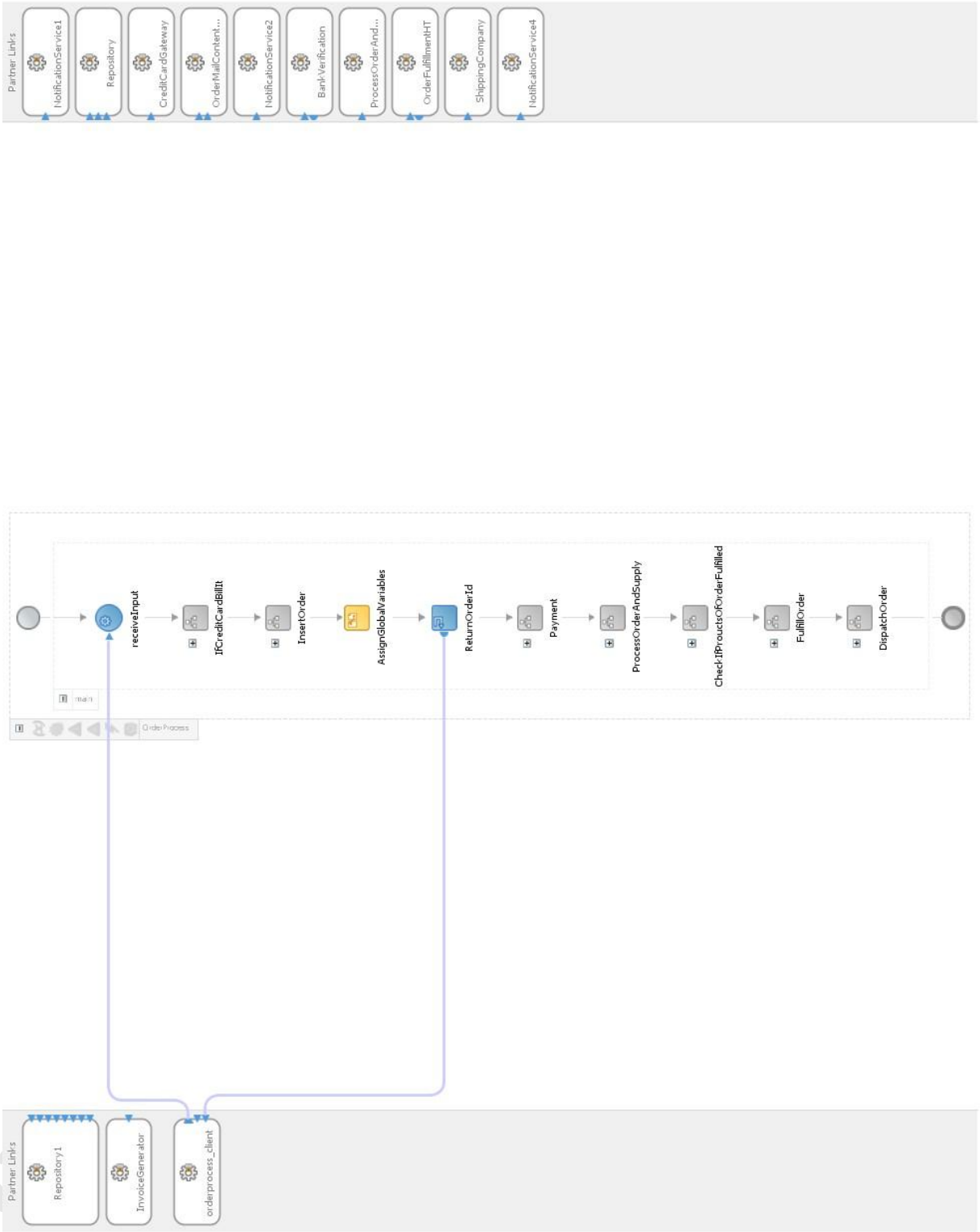
Πανεπιστήμιο Πειραιώς



Σχήμα 34: Το Composite.xml της διαδικασίας παραγγελιών με τα PartnerLinks και τα links με τα components της Oracle

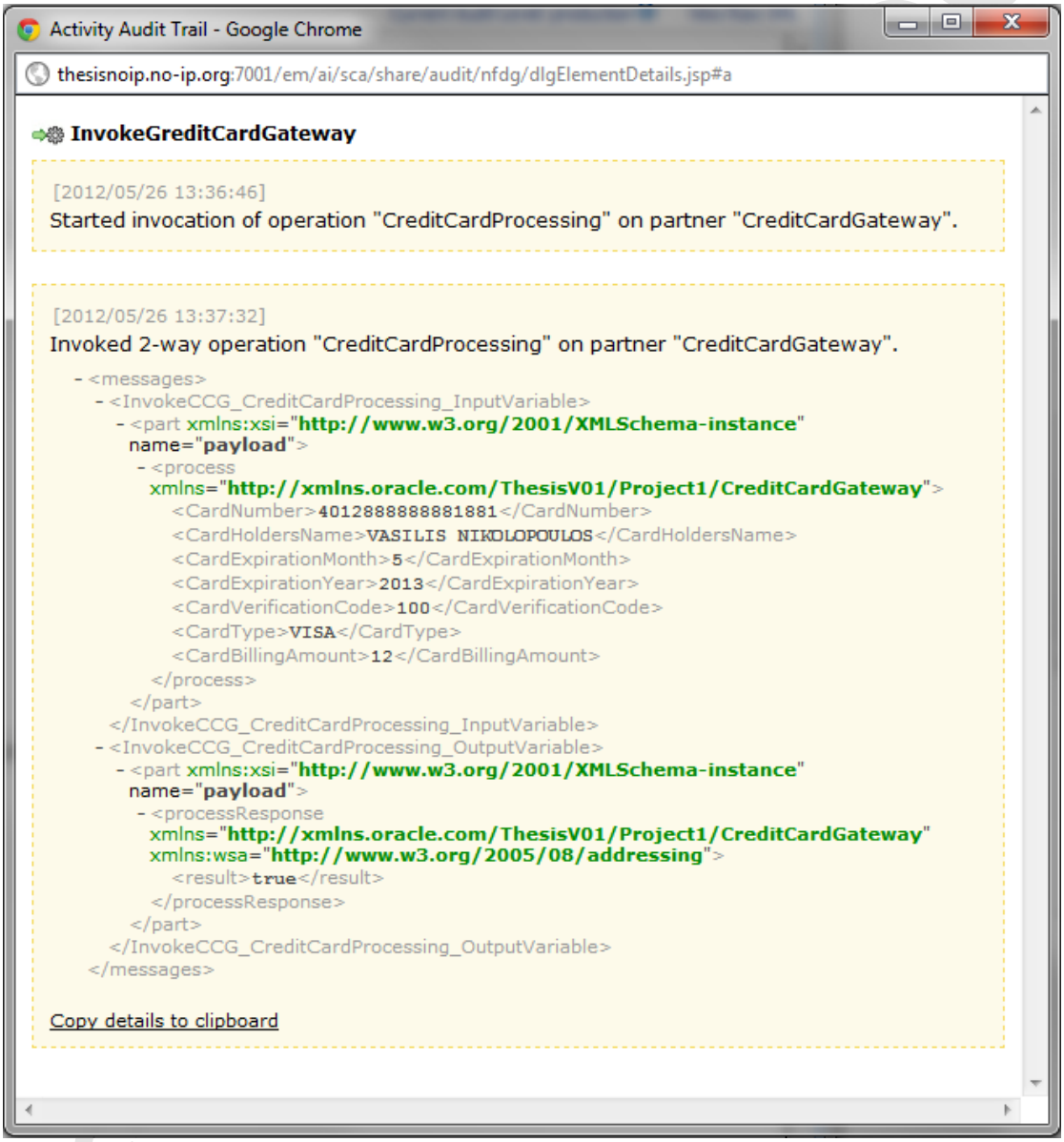
4.3.4 Ανάλυση διαδικασίας

Στο σχήμα 37 φαίνεται η διαδικασία συμπτυγμένη και ακολουθεί η ανάλυση της.



Σχήμα 35: Διαδικασία παραγγελίας BPEL συμπτυγμένη

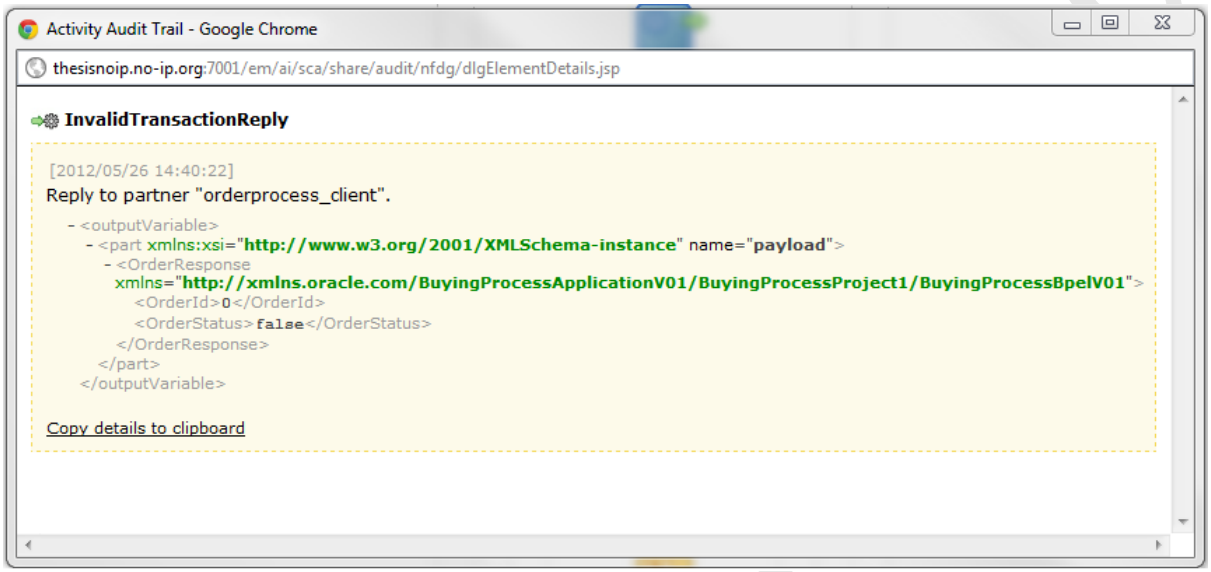
Η διαδικασία εκκινείται όταν ληφθεί το κατάλληλο μήνυμα SOAP. Το πρώτο βήμα τις διαδικασίας είναι να ελέγξει αν έχει επιλεγεί η πιστωτική κάρτα ως επιθυμητός τρόπος πληρωμής. Αν δεν έχει επιλεγεί ως επιθυμητός τρόπος πληρωμής η πιστωτική κάρτα η διαδικασία συνεχίζει με μια κενή δραστηριότητα <empty> NoCreditCardSelected σχήμα 41. Αν έχει επιλεγεί η πιστωτική κάρτα ως επιθυμητός τρόπος πληρωμής γίνεται αντιστοίχηση των τιμών των μεταβλητών εισόδου και καλείται μέσο της δραστηριότητας <invoke> το Web service (BPEL) CreditCardGateway σχήμα 38.



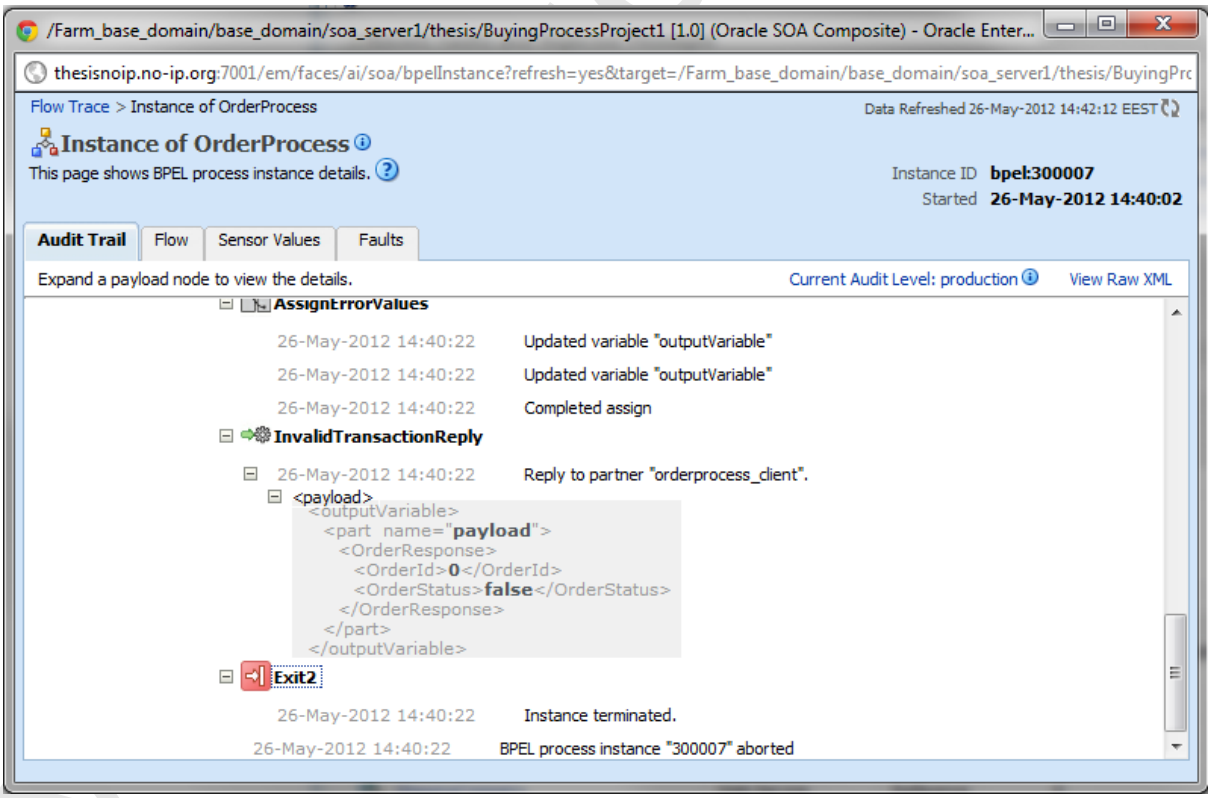
Σχήμα 36: BPEL Engine message Input/Output CreditCardGateway

Όταν επιστραφεί η boolean απάντηση του Web service αντιστοιχείται και ελέγχεται αν η τιμή της είναι <result>true</result> (βλ. σχήμα 38) σημαίνει ότι η διαδικασία εκτελέστηκε επιτυχώς και υπήρξε χρέωση της κάρτας σε αυτή την περίπτωση εκτελείται μια κενή δραστηριότητα DoNothing σχήμα 41 και η διαδικασία συνεχίζει στο επόμενο βήμα. Αν όμως η επιστρεφόμενη τιμή της είναι false γίνεται αντιστοίχηση των τιμών λάθους και επιστρέφονται ως μήνυμα απάντησης στον

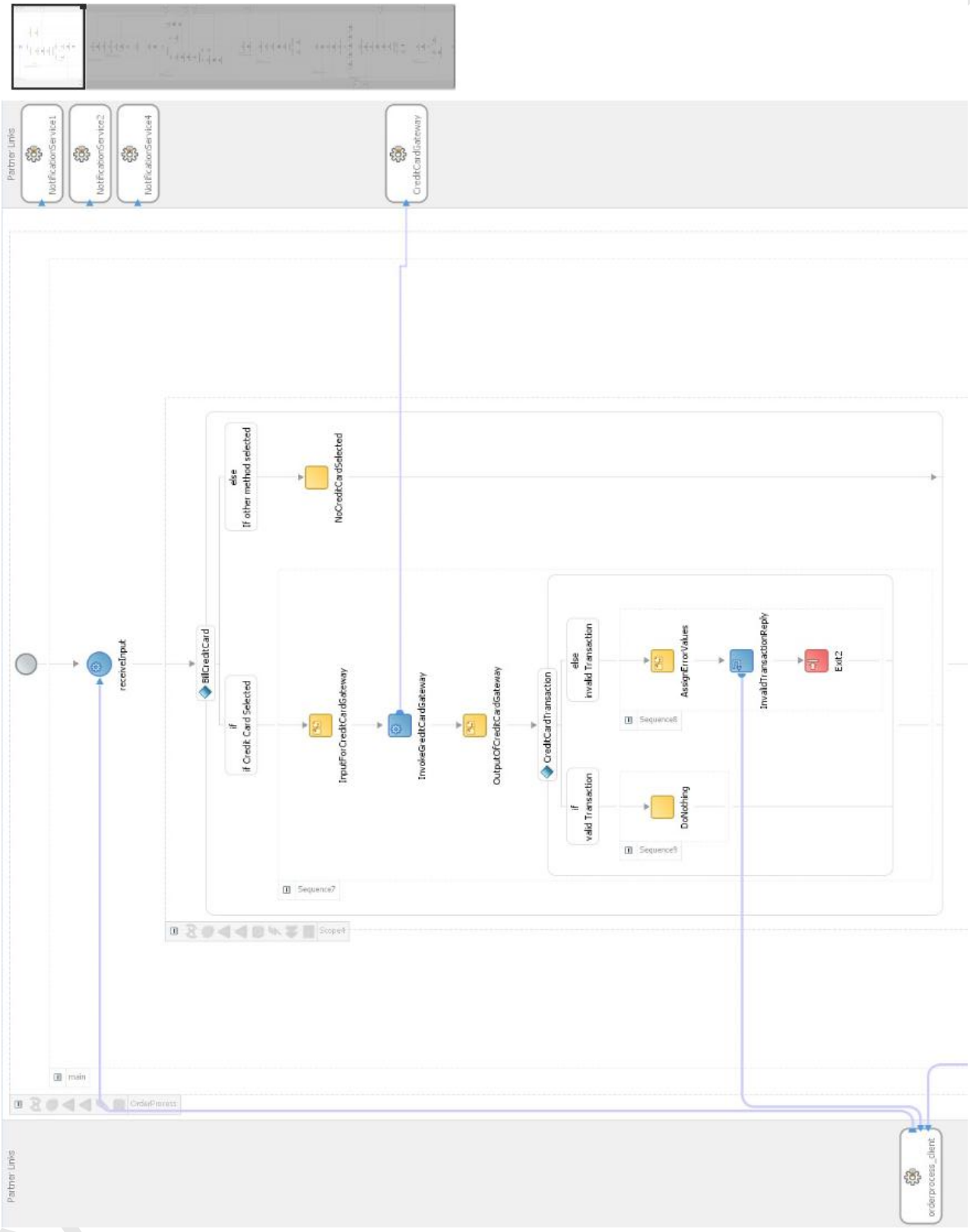
πελάτη της παραγγελίας σχήμα 39 μέσο της δραστηριότητας <reply> που φαίνεται στο σχήμα 41 με την ονομασία InvalidTransactionReply . Το στιγμιότυπο της διαδικασίας τερματίζεται σε αυτό το σημείο όπως φαίνεται και στο σχήμα 40.



Σχήμα 37: BPEL Engine message Output InvalidTransactionReply

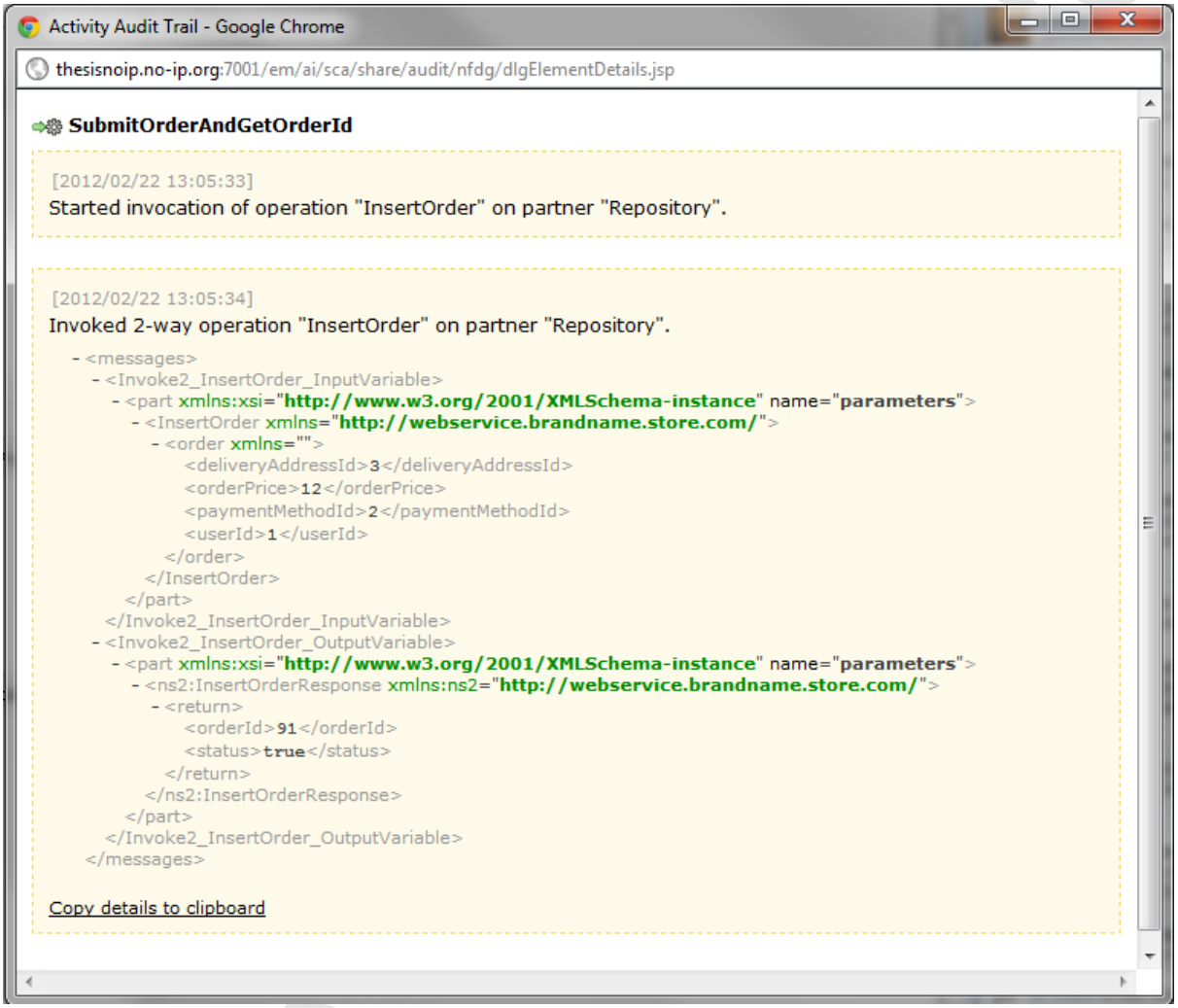


Σχήμα 38: BPEL engine Process Instance termination



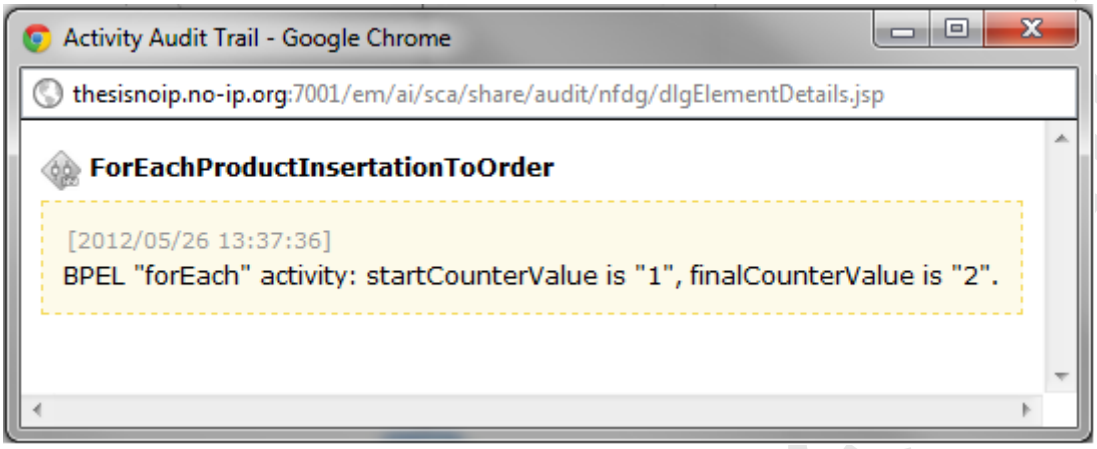
Σχήμα 39: Ανεπτυγμένη διαδικασία παραγγελίας (1/6)

Επόμενο βήμα της διαδικασίας είναι η καταχώρηση της παραγγελίας στη βάση δεδομένων και η επιστροφή του αναγνωριστικού της, για αυτό το σκοπό καλείται η μέθοδος InsertOrder του Web service του Repository, σχήμα 42, η οποία φαίνεται στο σχήμα 46 με την ονομασία SubmitOrderAndGetOrderId.

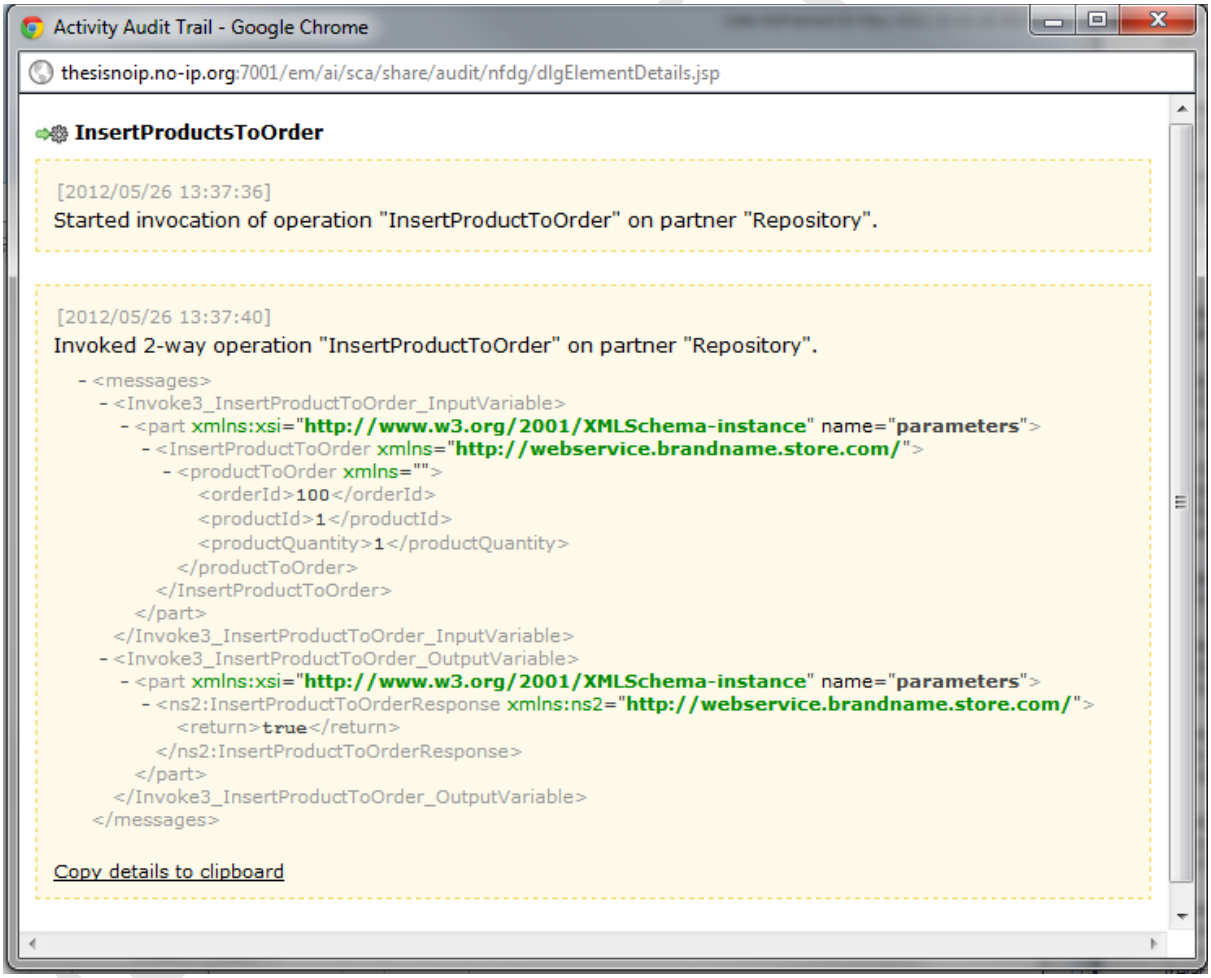


Σχήμα 40:BPEL Engine message Input/Output SubmitOrderAndGetOrderId

Όταν επιστραφεί το αναγνωριστικό της παραγγελίας εκτελείται μια δραστηριότητα <forEach> η οποία φαίνεται με την ονομασία ForEachProductInsertationToOrder σχήμα 46 για να γίνει manipulation στα δεδομένα XML της παραγγελίας στο σχήμα 43 φαίνεται ότι η παραγγελία είχε δύο προϊόντα καθώς η τελική τιμή του finalCounterValue είναι το 2. Αναλυτικότερα με το manipulation των δεδομένων διαχωρίζονται τα προϊόντα της παραγγελίας και καταχωρούνται ένα-ένα στην βάση δεδομένων στο κατάλληλο πίνακα καλώντας την μέθοδο InsertProductToOrder του Web service του Repository σχήμα 44, η οποία φαίνεται και στο σχήμα 46 με την ονομασία InsertProductsToOrder.

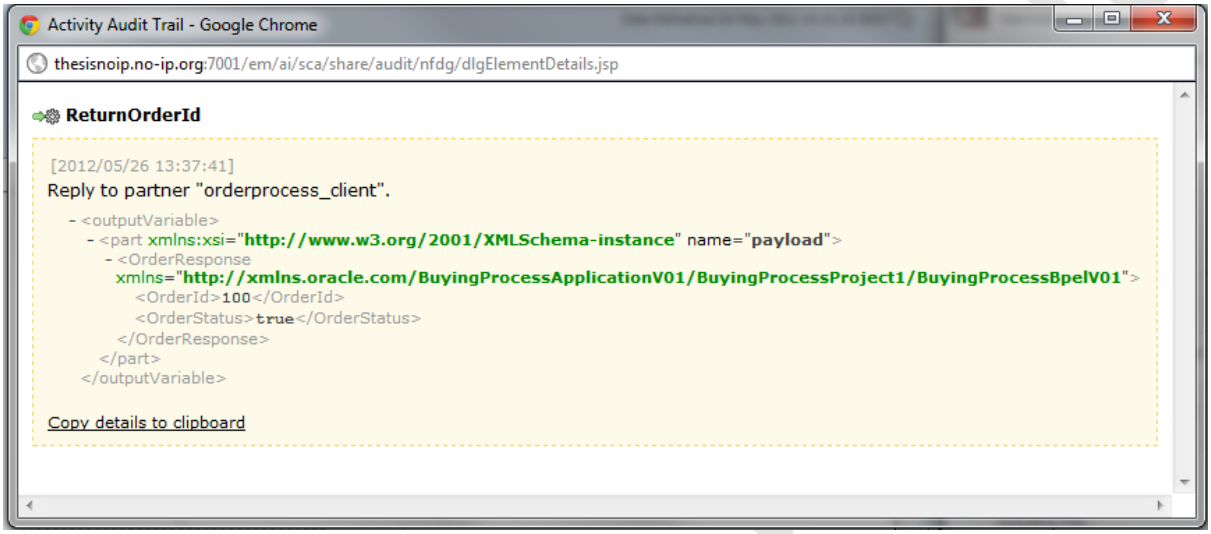


Σχήμα 41:BPEL Engine message μετρητής των προϊόντων παραγγελίας.



Σχήμα 42: BPEL Engine message Inpout/Outpout InsertProductToOrder

Εφόσον η παραγγελία καταχωρηθεί με επιτυχία επιστρέφεται στον πελάτη της διαδικασίας ένα μήνυμα SOAP με το αναγνωριστικό της παραγγελίας και ότι εκτελέσθηκε επιτυχώς <OrderStatus>true</OrderStatus> σχήμα 45, η δραστηριότητα <reply> φαίνεται στο σχήμα 46 με την ονομασία ReturnOrderId.



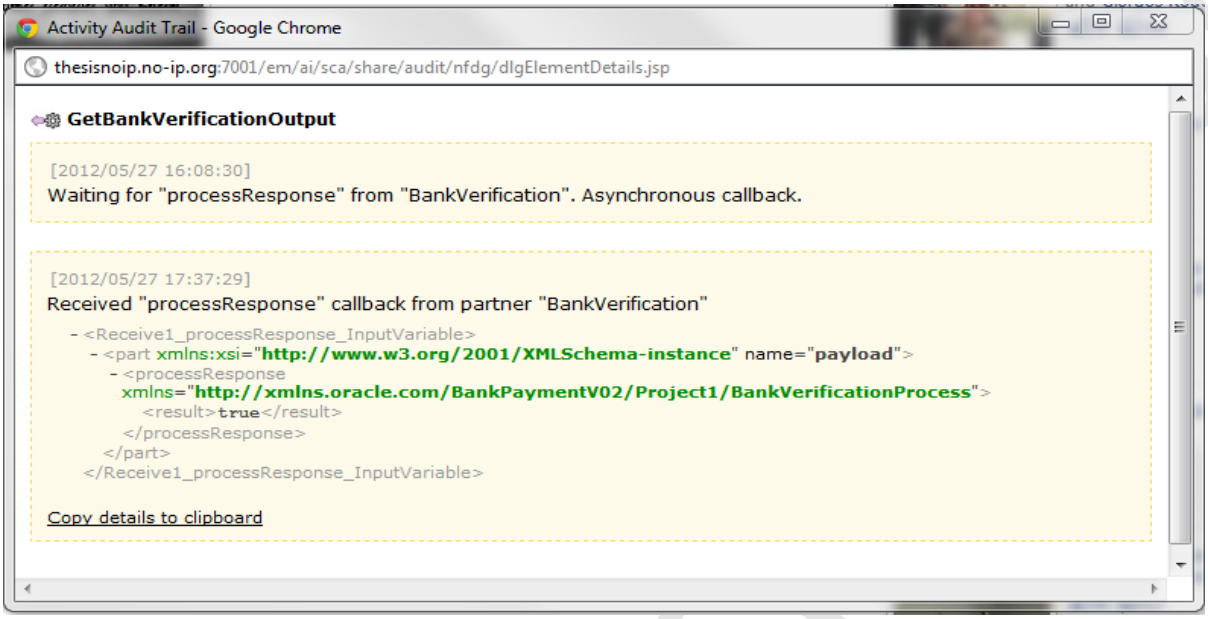
Σχήμα 43: BPEL Engine message Output ReturnOrderId

Σχήμα 44: Ανεπτυγμένη διαδικασία παραγγελίας (2/6)

Η διαδικασία έπειτα μπαίνει σε παράλληλη ροή, για την εκτέλεση της παράλληλης ροής χρησιμοποιείται η δραστηριότητα <flow> η οποία συμβολίζεται με ένα γκρι ρόμβο ο οποίος περιέχει ένα σταυρό (σχήμα 53), στο δεξί μέρος της παράλληλης ροής μια αποστέλλεται ενημερωτικό email στον πελάτη της παραγγελίας με το αναγνωριστικό και τα στοιχεία της, και στην άλλη ροή εξετάζεται αν ο επιθυμητός τρόπος πληρωμής είναι τα μετρητά ή τραπεζική κατάθεση και η ροή της διαδικασίας οδηγείται στην σωστή διαδρομή.

Αναλυτικότερα το αριστερό μέρος του σχήματος 53 είναι αυτό που εξετάζει τον τρόπο πληρωμής, αν έχει επιλεγεί τραπεζική κατάθεση η διαδικασία καλεί μια ασύγχρονη Human Task για την επιβεβαίωση της τραπεζικής κατάθεσης, στο σχήμα 47 φαίνεται το μήνυμα που αποστέλλεται για την εκκίνηση του Human Task της επιβεβαίωσης της τραπεζικής διαδικασίας , και στο σχήμα 48 φαίνεται η απάντηση του Human Task. Οι δραστηριότητες μπορεί να φανούν στο σχήμα 53 με τις ονομασίες SendDataForBankVerification και GetBankVerificationOutput αντίστοιχα.

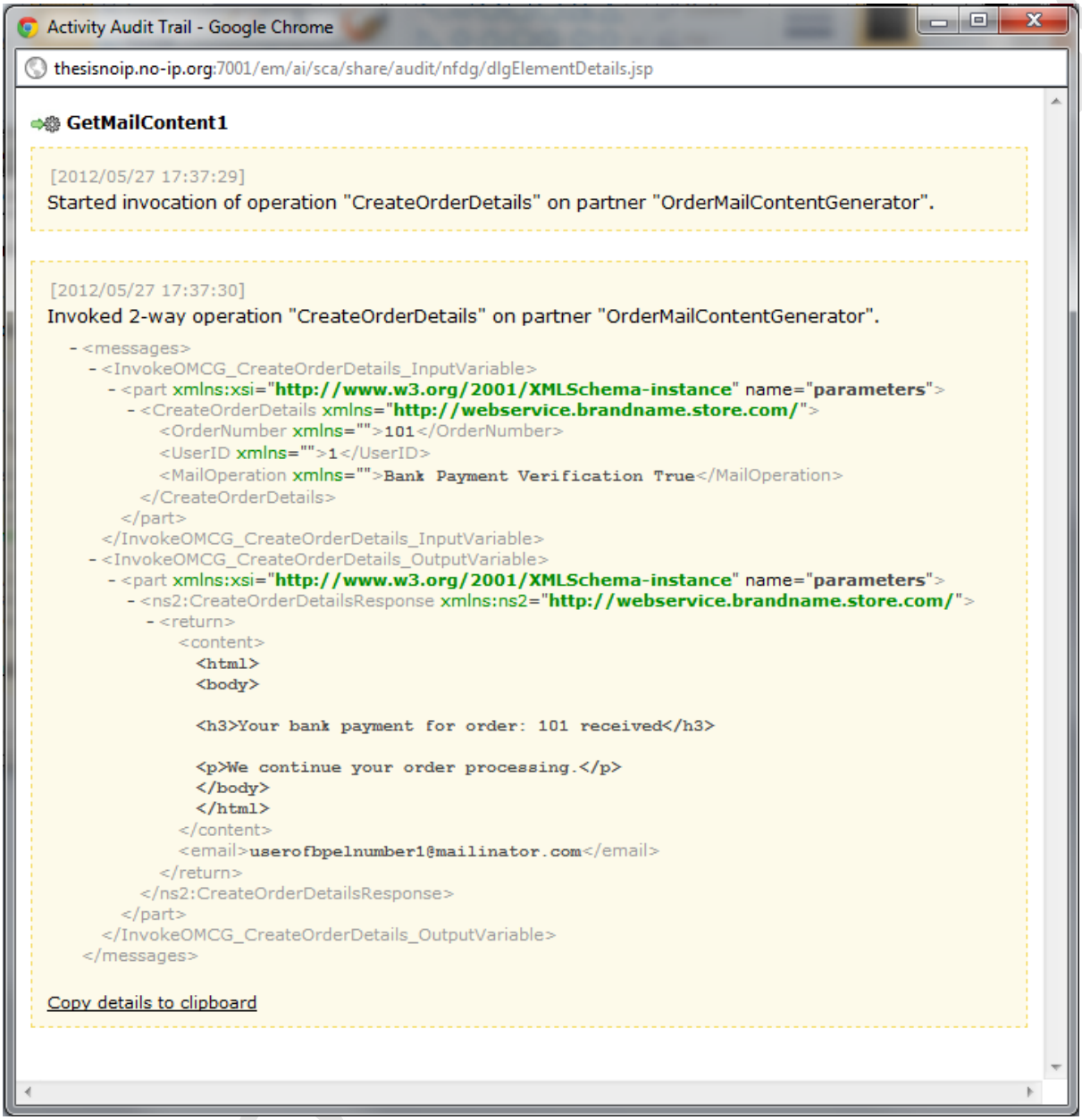
Σχήμα 45: BPEL Engine message Input SendDataForBankVerification



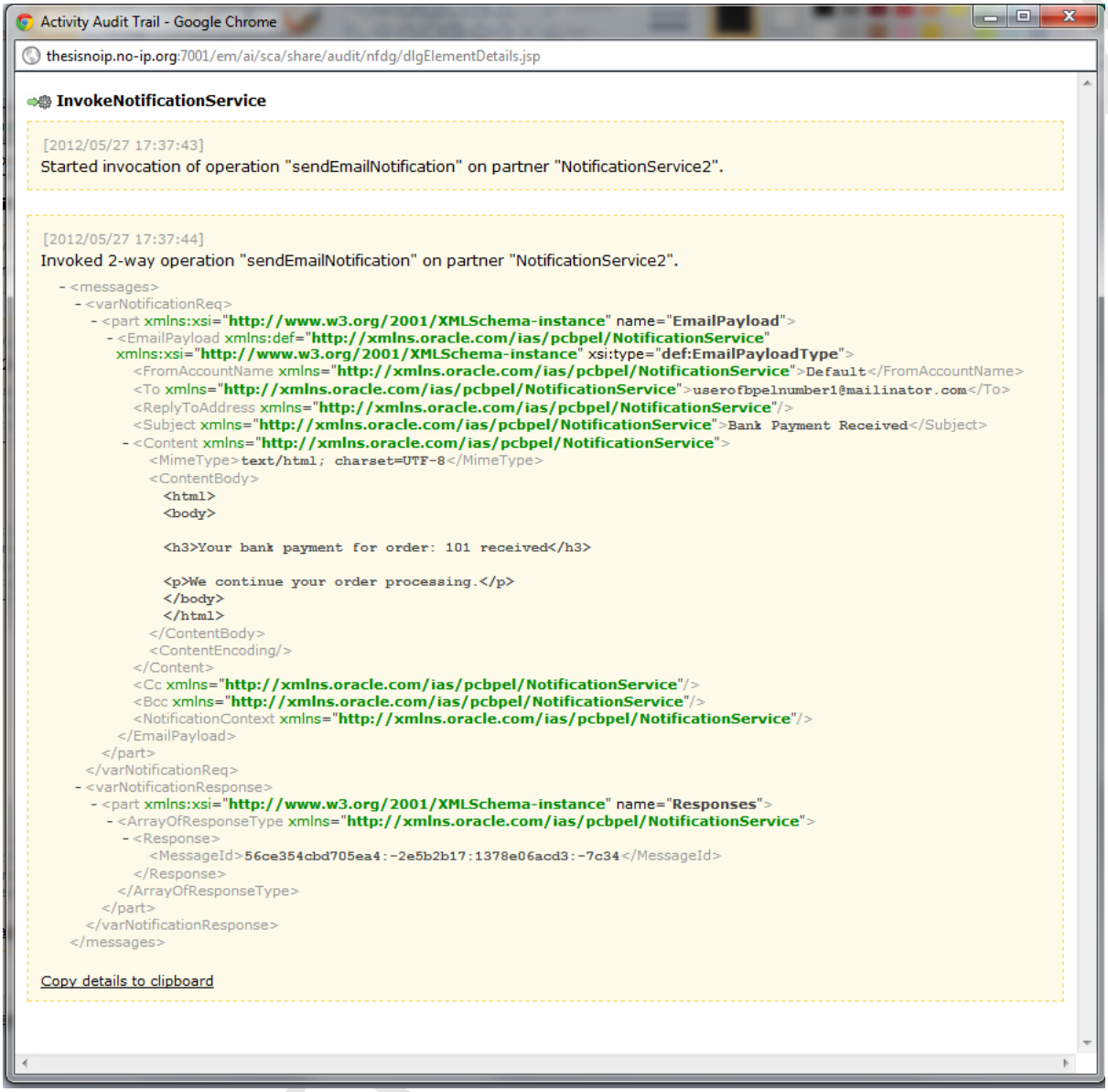
Σχήμα 46: BPEL Engine message Output GetBankVerificationOutput

Έπειτα ελέγχεται αν η επιβεβαίωση της τραπεζικής κατάθεσης. Αν ήταν επιτυχής αποστέλλεται ενημερωτικό e-mail στον πελάτη της παραγγελίας. Για την αποστολή του ενημερωτικού αυτού e-mail γίνεται κλήση της μεθόδου CreateOrderDetails του Web service του OrderMailContentGenerator σχήμα 49 που επίσης φαίνεται και στο σχήμα 53 με την ονομασία GetMailContent1. Έπειτα με η έξοδος του προηγουμένου Web service χρησιμοποιείται ως είσοδος κατά την κλήση του component της Oracle για την αποστολή e-mail μηνυμάτων InvokeNotificationService σχήμα 50, το οποίο φαίνεται και στο σχήμα 53 με την ονομασία BankPaymentVerification.

Πανεπιστήμιο Πειραιώς



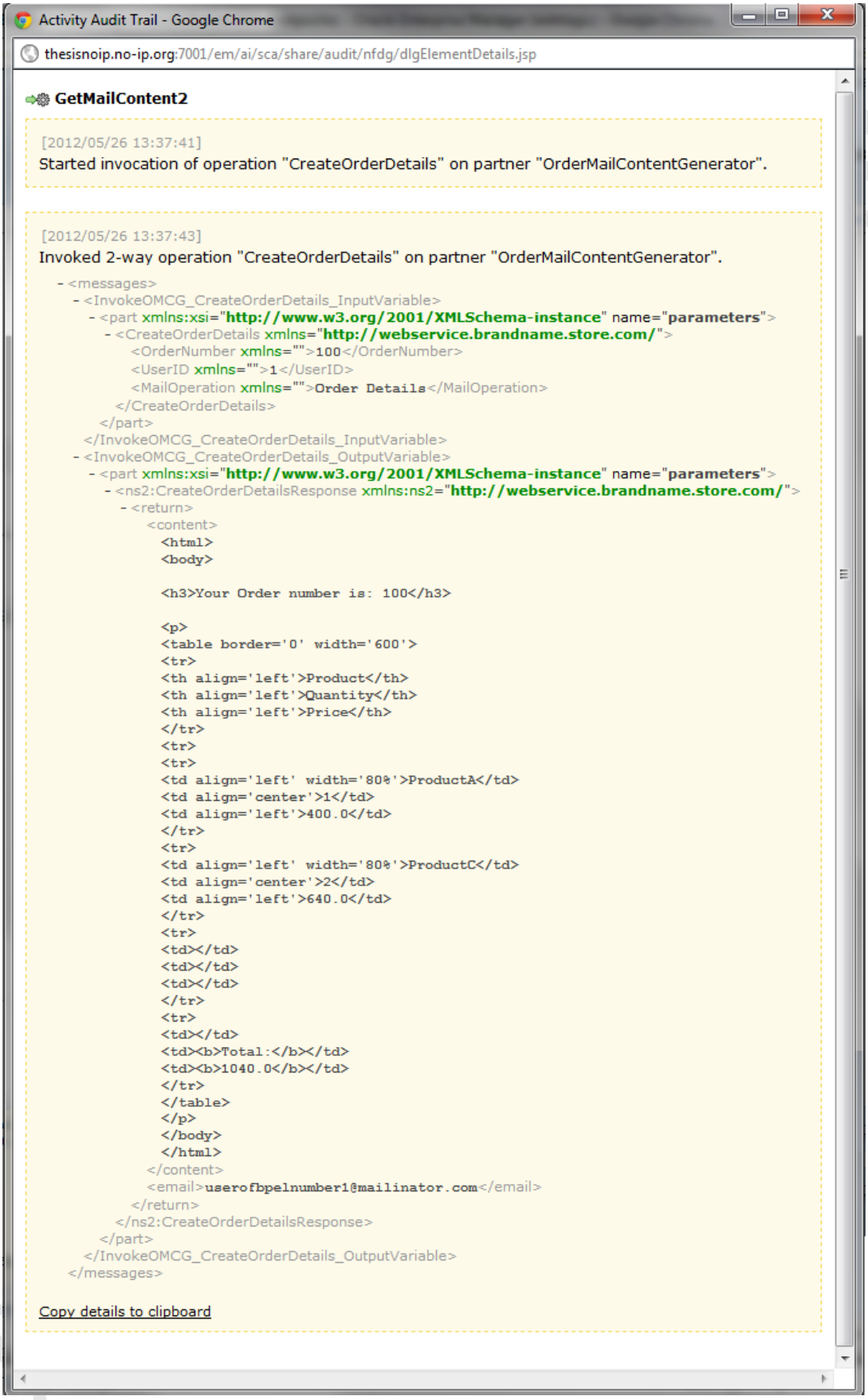
Σχήμα 47: BPEL Engine message Input/Output GetMailContent1



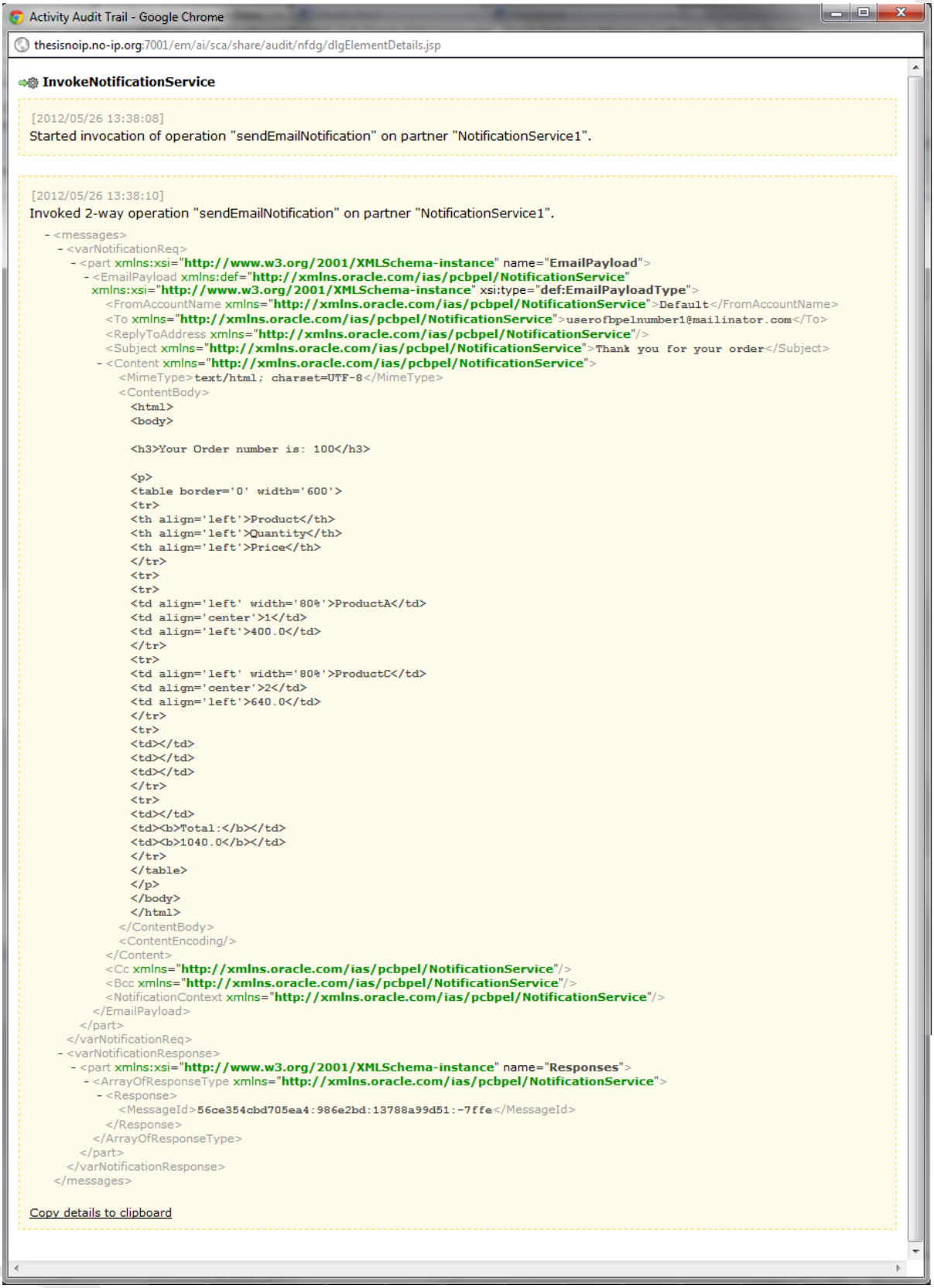
Σχήμα 48: BPEL Engine message Input/Output InvokeNotificationService

Αν έχει επιλεγεί ως μέθοδος πληρωμής τα μετρητά εκτελείται μια κενή δραστηριότητα την DoNothing2 όπως μπορεί να φανεί και στο σχήμα 53, και η διαδικασία συνεχίζει στο επόμενο της βήμα/δραστηριότητα.

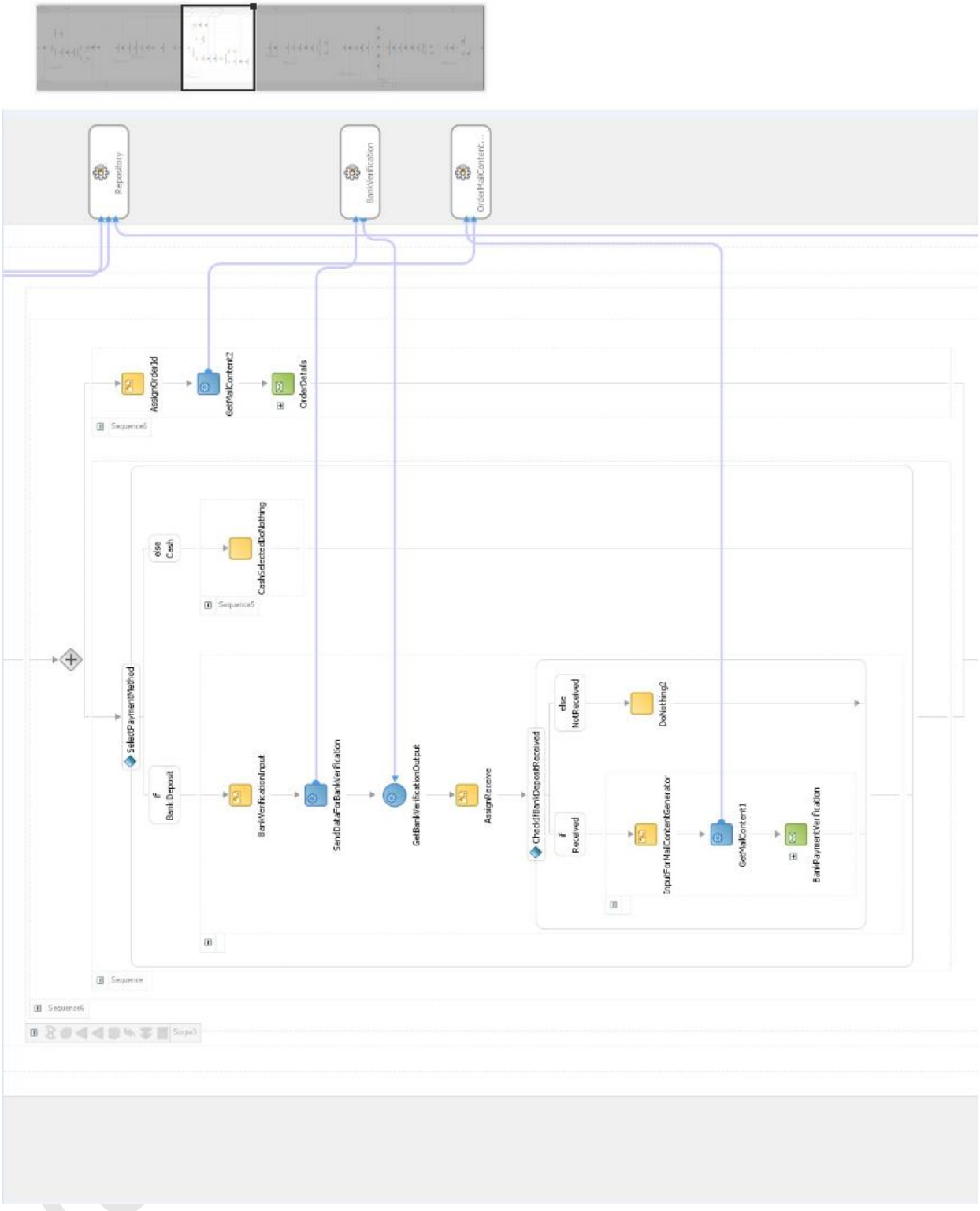
Στο δεξί μέρος του σχήματος της διαδικασίας BPEL εκτελείτε, όπως αναφέρθηκε σε προγενέστερη παράγραφο, η αποστολή ενημερωτικού e-mail στον πελάτη με το αναγνωριστικό της παραγγελίας και τα στοιχεία της. Για να αποσταλεί το ενημερωτικό αυτό e-mail γίνεται κλήση της μεθόδου CreateOrderDetails του Web service του OrderMailContentGenerator σχήμα 51, το οποίο μπορεί να φανεί και στο σχήμα 53 με την ονομασία GetMailContent. Έπειτα με η έξοδος του προηγουμένου Web service χρησιμοποιείται ως είσοδος κατά την κλήση του component της Oracle για την αποστολή e-mail μηνυμάτων InvokeNotificationService σχήμα 52, το οποίο φαίνεται στο σχήμα 53 με την ονομασία OrderDetails.



Σχήμα 49: BPEL Engine message Input/Output GetmailContent2

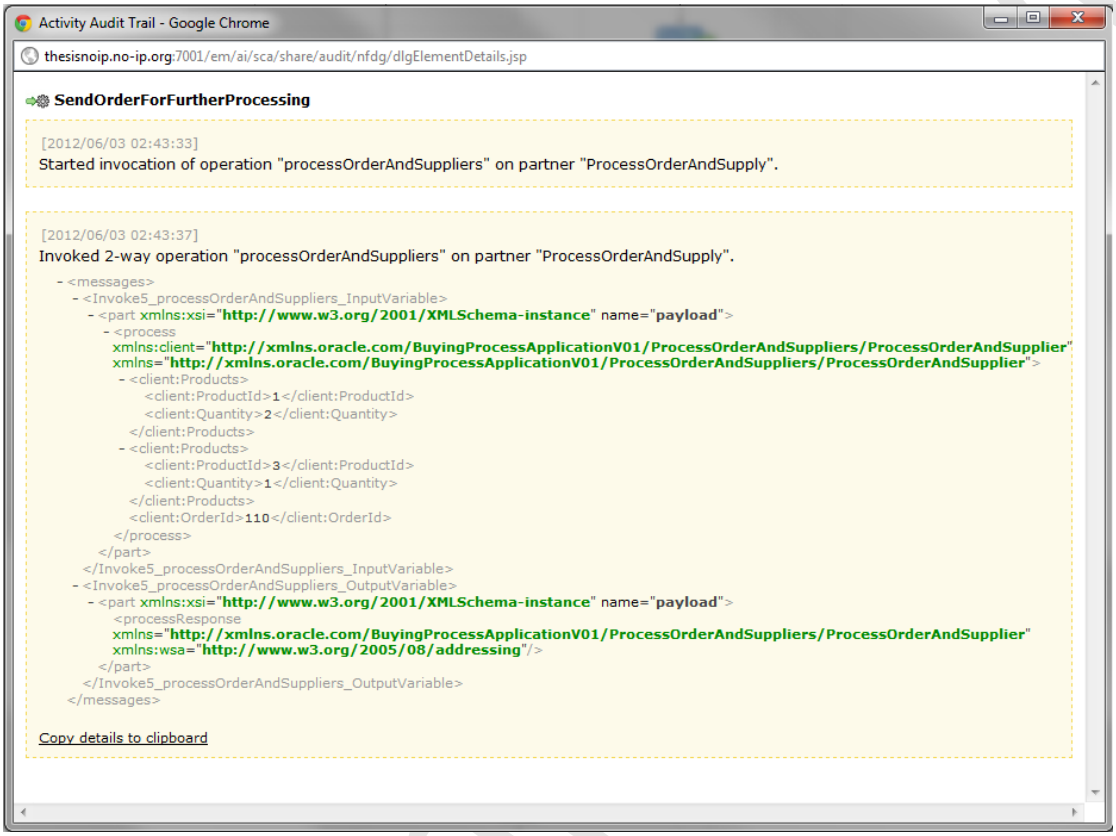


Σχήμα 50: BPEL Engine message Input/Output InvokeNotificationService



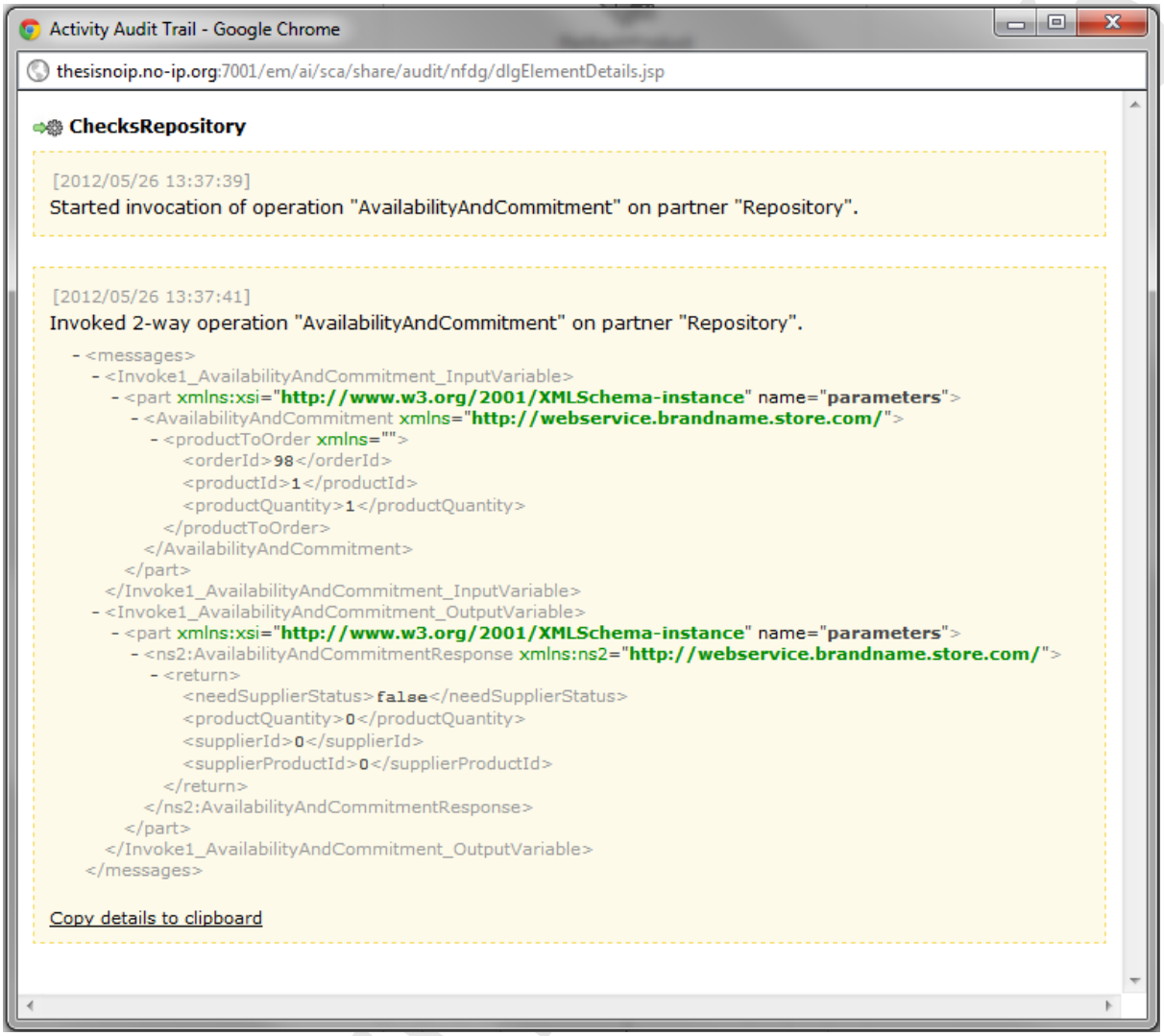
Σχήμα 51: Ανεπτυγμένη διαδικασία παραγγελίας (3/6)

Επόμενο βήμα της διαδικασίας παραγγελίας είναι η κλήση μια άλλης διαδικασίας BPEL για την περεταίρω επεξεργασία της. . Αυτό γίνεται με την κλήση της Web Service μεθόδου ProcessOrderAndSuppliers του ProcessOrderAndSupply σχήμα 54, η οποία μπορεί να φανεί με την ονομασία SendOrderForFurtherProcessing στο σχήμα 61.



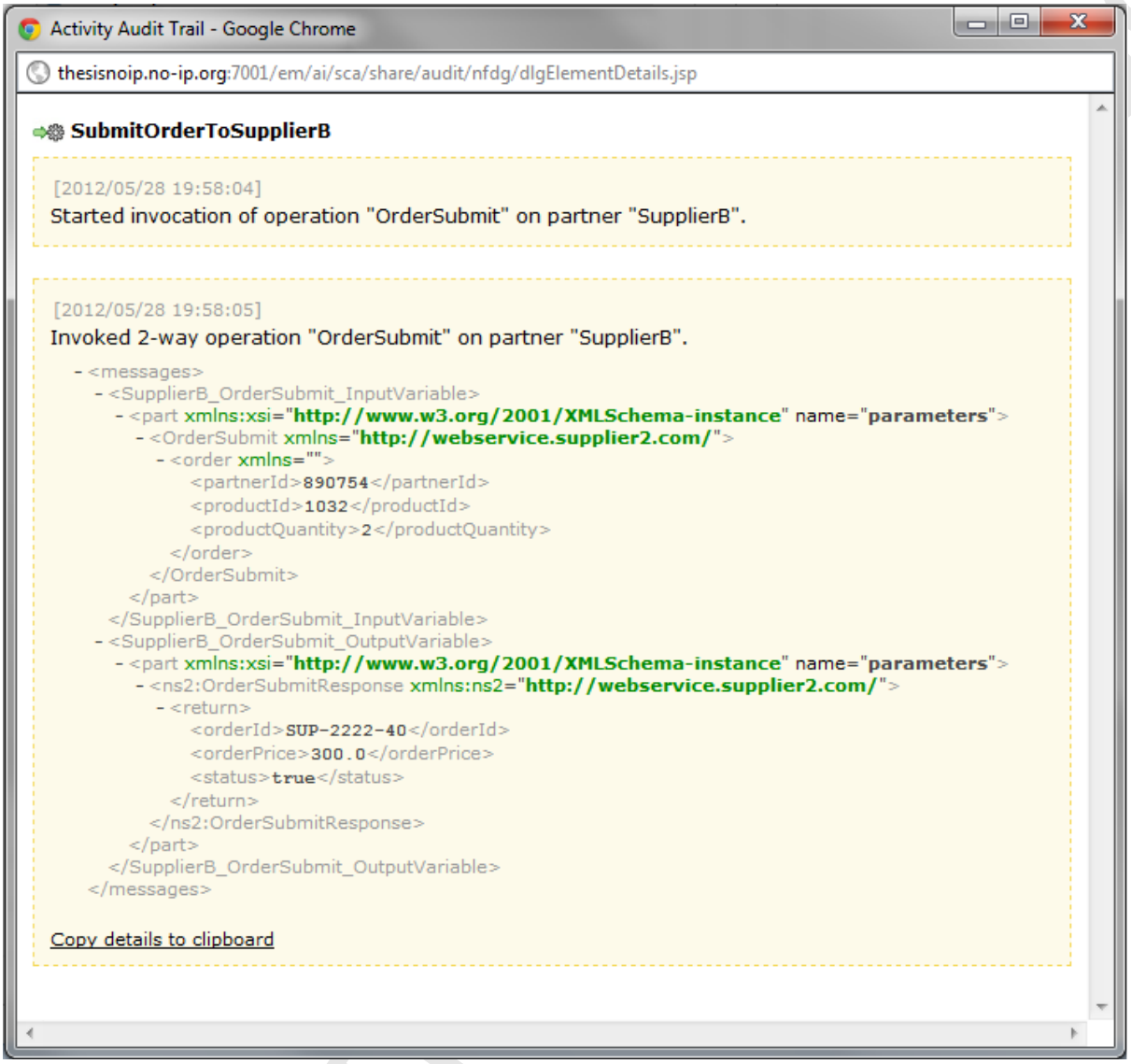
Σχήμα 52:BPEL Engine message Input/Output SendOrderForFurtherProcessing

Στο επόμενο βήμα η διαδικασία BPEL που κλήθηκε και μπορεί να φανεί στο σχήμα 62 ελέγχει την διαθεσιμότητα κάθε προϊόντος της παραγγελίας στην αποθήκη και εάν υπάρχει επαρκές υπόλοιπο το δεσμεύει, αν δεν υπάρχει επιστρέφει το αναγνωριστικό του προϊόντος καθώς και την ποσότητα που υπολείπεται όπως φαίνεται και στο σχήμα 55, αυτό γίνεται με την κλήση της μεθόδου AvailabilityAndCommitment του Web service του Repository που μπορεί να φανεί και στο σχήμα 61 με την ονομασία CheckRepository.



Σχήμα 53: BPEL Engine message Input/Output ChecksRepository

Το επόμενο βήμα είναι ο έλεγχος της εξόδου του προηγούμενου βήματος σχήμα 55. Αν το <needSupplierStatus> επιστραφεί με τιμή false τότε εκτελείται μια κενή δραστηριότητα η οποία έχει την ονομασία DoNothing στο σχήμα 62. Αν το <needSupplierStatus> επιστραφεί με τιμή true τότε σε επόμενο έλεγχο εξακριβώνεται σε ποιόν προμηθευτή ανήκει το προϊόν και η διαδικασία δρομολογείται αντίστοιχα. Από σχήμα φαίνεται πως η διαδικασία ανήκει στον supplier με αναγνωριστικό το 2 οποίο στον κώδικα της BPEL έχει οριστεί ως SUPPLIER B, άρα η διαδικασία δρομολογείται στον SUPPLIER B. Η διαδικασία για να καταχωρήσει μια παραγγελία προϊόντων καλεί την μέθοδο OrderSubmit του SupplierB σχήμα 56 και αποστέλλει στο μήνυμα το αναγνωριστικό του πελάτη της παραγγελίας τον κωδικό προϊόντος και την ποσότητα, η δραστηριότητα αυτή μπορεί να φανεί στο σχήμα 62 με την ονομασία SubmitOrderToSupplierB. Το Web service επιστρέφει το αναγνωριστικό της παραγγελίας, την τιμολόγηση της καθώς και αν εκτελέσθηκε επιτυχώς.

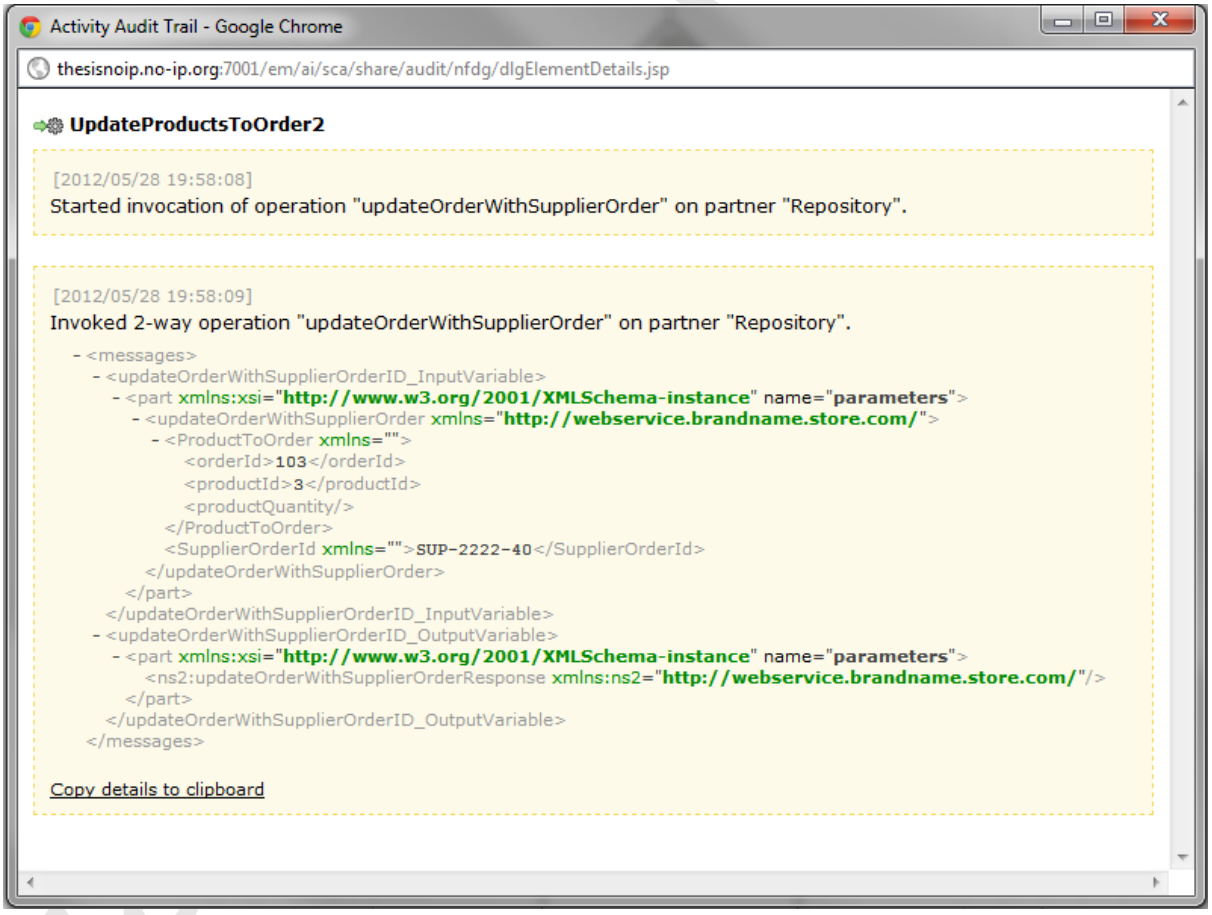


Σχήμα 54: BPEL Engine message Input/Output SubmitOrderToSupplierB

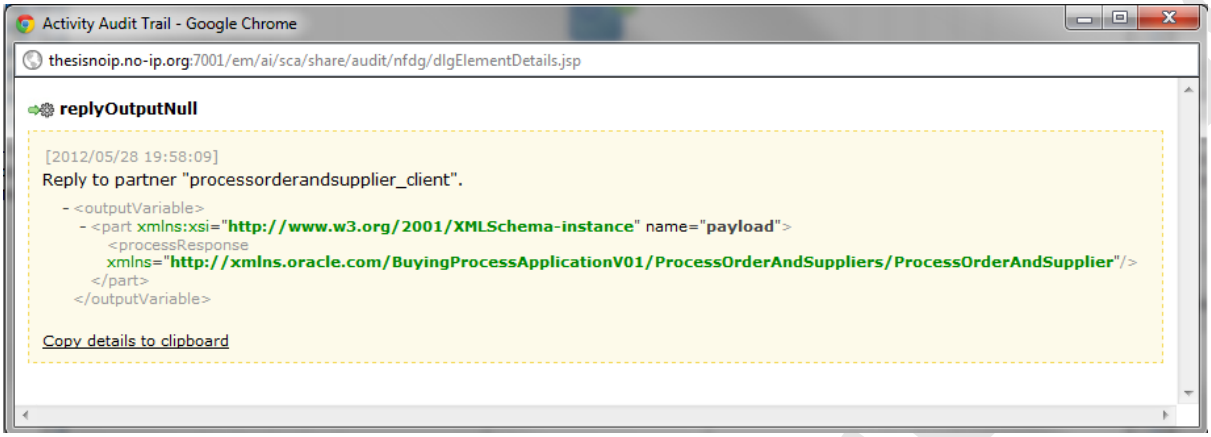
Έπειτα η διαδικασία εισέρχεται σε μια συγχρονισμένη ροή στο αριστερό μέρος όπως φαίνεται και στο σχήμα 62 η δραστηριότητα VerifyPhysicalReceivedOrderFromSupplier1 καλεί την Web Service μέθοδο SendSupplierOrder του SupplierOrderSubmitHumanTask και αποστέλλει ένα μήνυμα με τα δεδομένα που χρειάζονται στο Human Task της επιβεβαίωσης λήψης παραγγελίας προϊόντων από προμηθευτή όπως μπορεί να φανεί στο σχήμα 57. Στο δεξί μέρος της ροής με την δραστηριότητα UpdateProductsToOrder1 του σχήματος 62 ενημερώνεται ο πίνακας PRODUCTS_TO_ORDER της βάσης δεδομένων με το αναγνωριστικό της παραγγελίας του προμηθευτή, αυτό συμβαίνει με την κλήση της μεθόδου UpdateOrderWithSupplierOrder του Web service του Repository σχήμα 58.

Το επόμενο βήμα της διαδικασίας είναι να επιστραφεί μια κενή δραστηριότητα <reply> η οποία έχει την ονομασία replyOutputNull και φαίνεται στο σχήμα 59 για να σηματοδοτήσει το τέλος της διαδικασίας BPEL που φαίνεται στο σχήμα 62.

Σχήμα 55: BPEL Engine message Input VerifyPhysicalReceivedOrderFromSupplier1

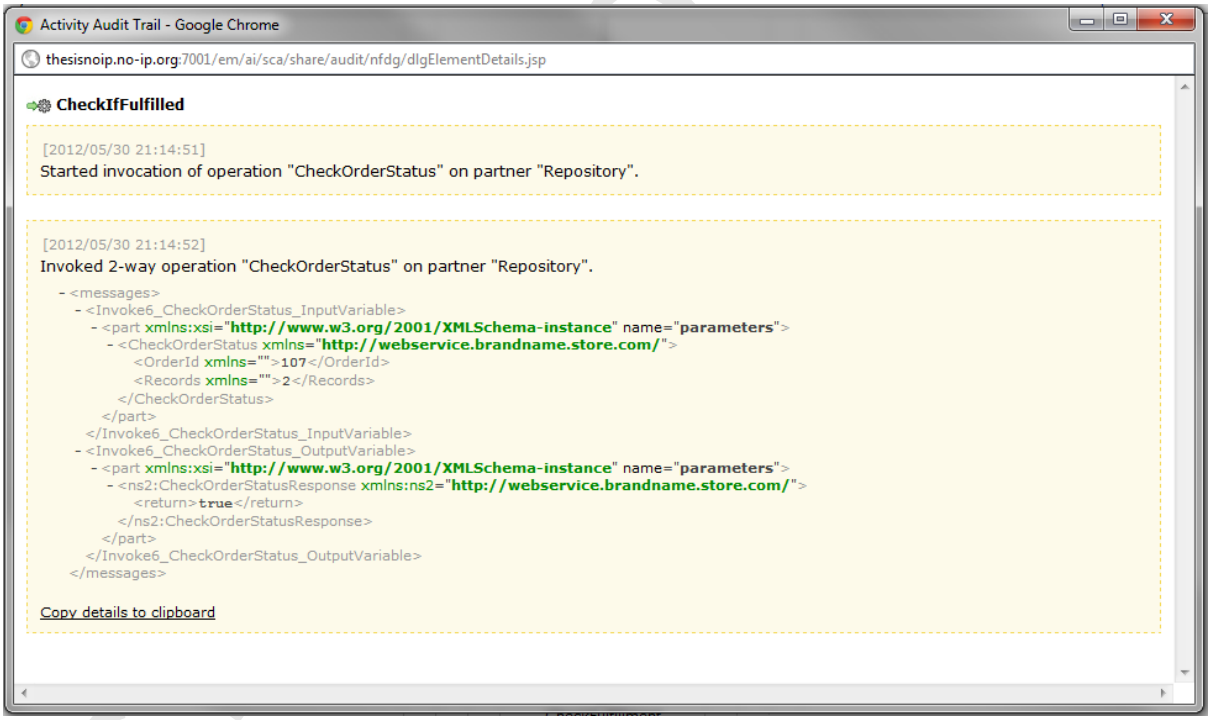


Σχήμα 56: BPEL Engine message Input/Output UpdateProductsToOrder2



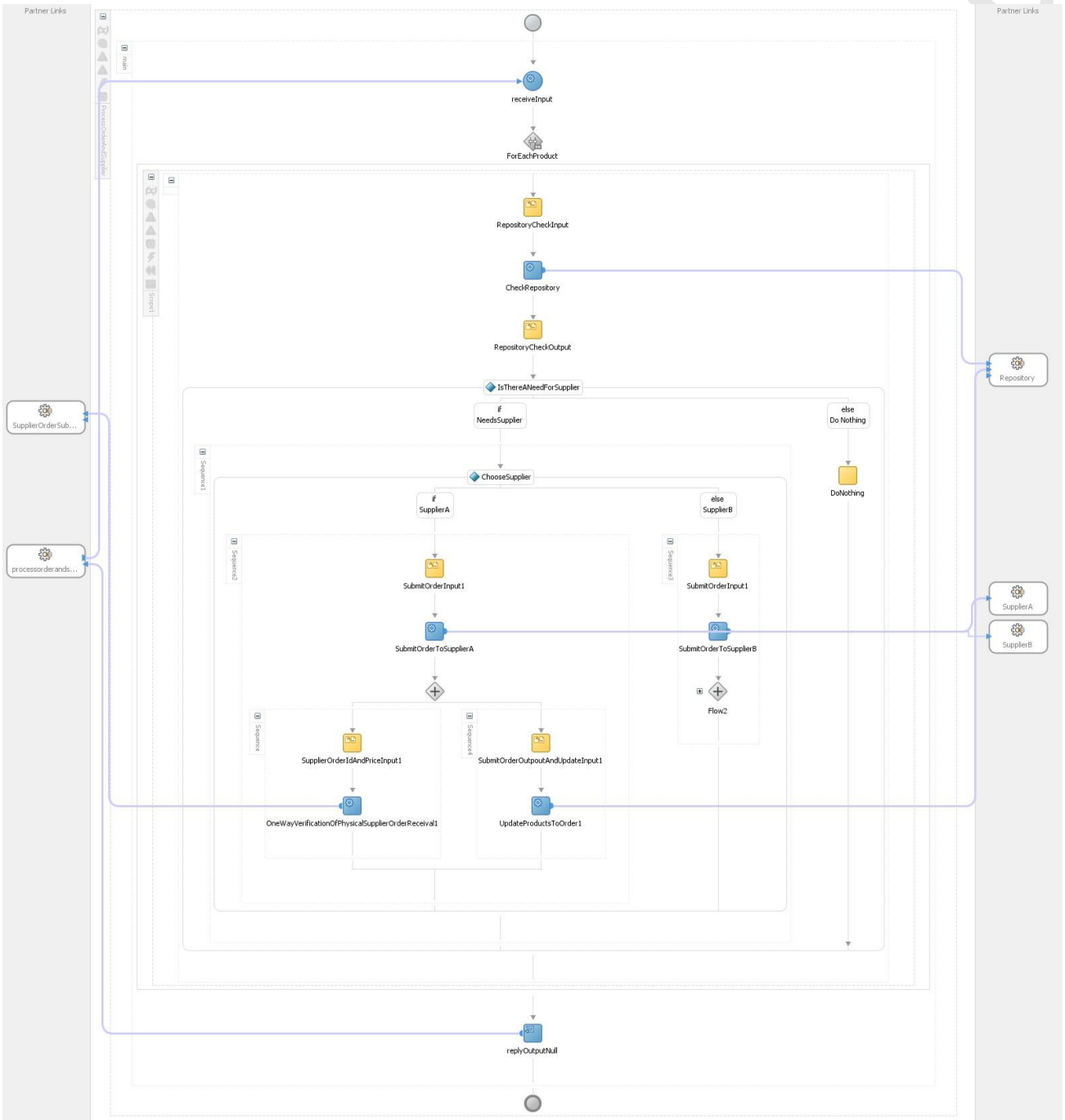
Σχήμα 57: BPEL Engine message Output replyOutputNull

Επόμενο βήμα είναι ο έλεγχος της διαθεσιμότητας των προϊόντων στο πίνακα της βάσης δεδομένων που αφορά την αποθήκη ανά προκαθορισμένα διαστήματα μέσο της δραστηριότητας <wait> WaitAndCheckAgain σχήμα 61, αυτό συμβαίνει με την κλήση της CheckOrderStatus του Web service του repository σχήμα 60, μπορεί να φανεί με την ονομασία CheckIfFulfilled στο σχήμα 61, όταν η απάντηση επιστραφεί και έχει την τιμή true η διαδικασία συνεχίζει στην επόμενη δραστηριότητα.



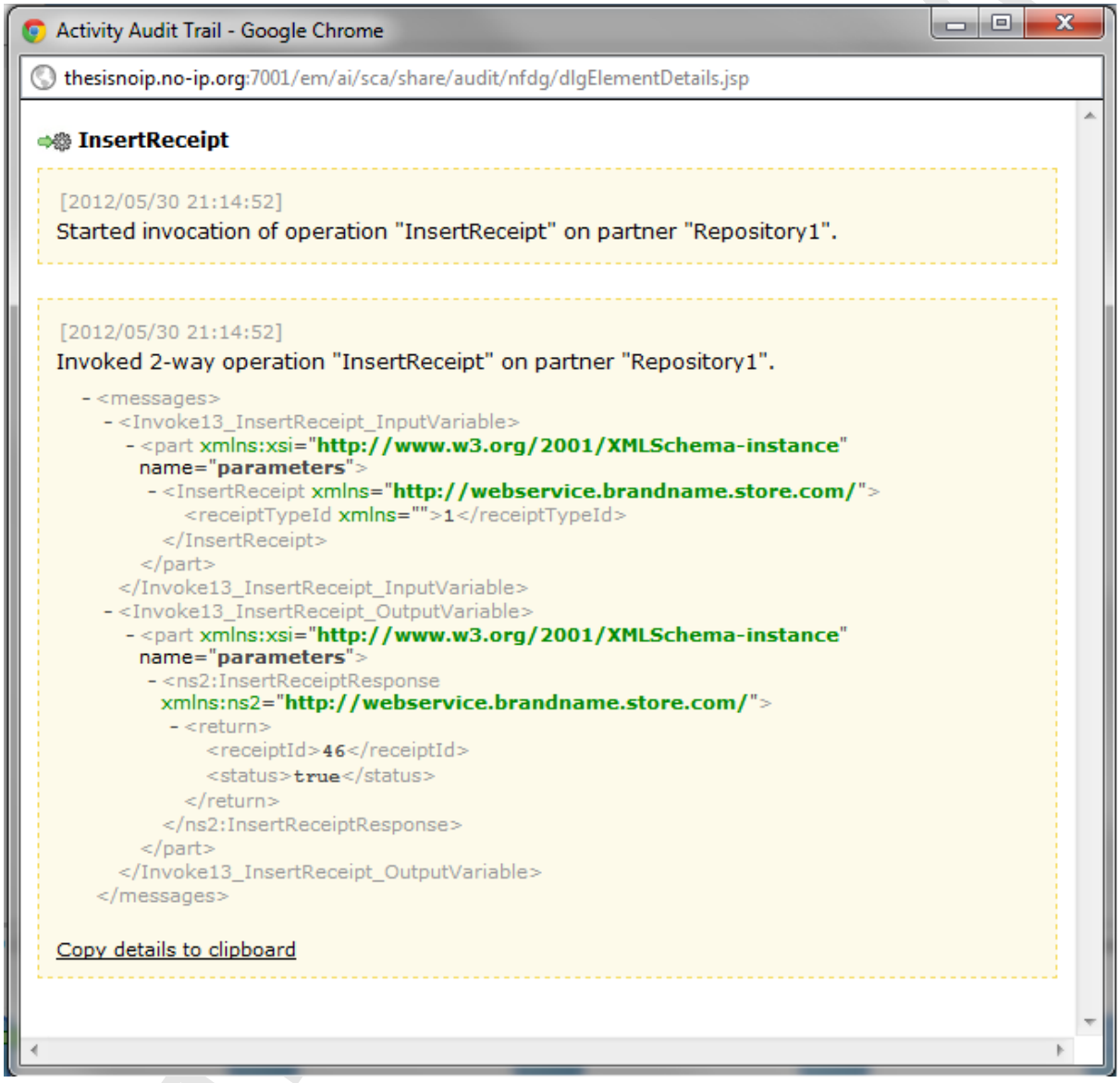
Σχήμα 58:BPEL Engine message Input/Output CheckIfFulfilled

Σχήμα 59: Ανεπτυγμένη διαδικασία παραγγελίας (4/6)



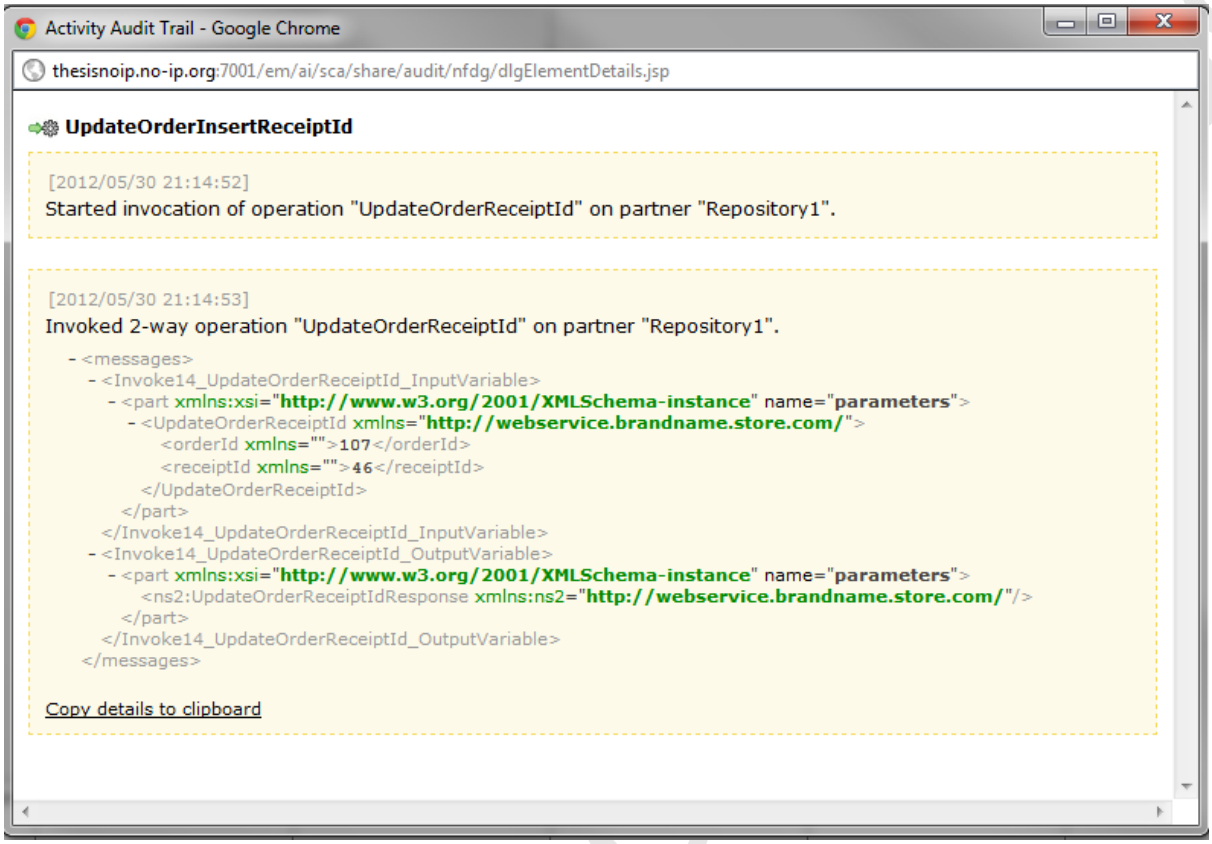
Σχήμα 60: Διαδικασία ελέγχου αποθήκης και παραγγελίας προϊόντων από προμηθευτές

Επόμενο βήμα της διαδικασίας είναι η έκδοση παραστατικού/απόδειξης αυτό συμβαίνει με την κλήση της μεθόδου InsertReceipt του Repository1 σχήμα 63 και σχήμα 74, η οποία παίρνει ως εισαγωγή τον τύπο της απόδειξης και επιστρέφει ένα αναγνωριστικό για αυτή <receiptId> και αν η διαδικασία εκτελέσθηκε επιτυχώς <status>.



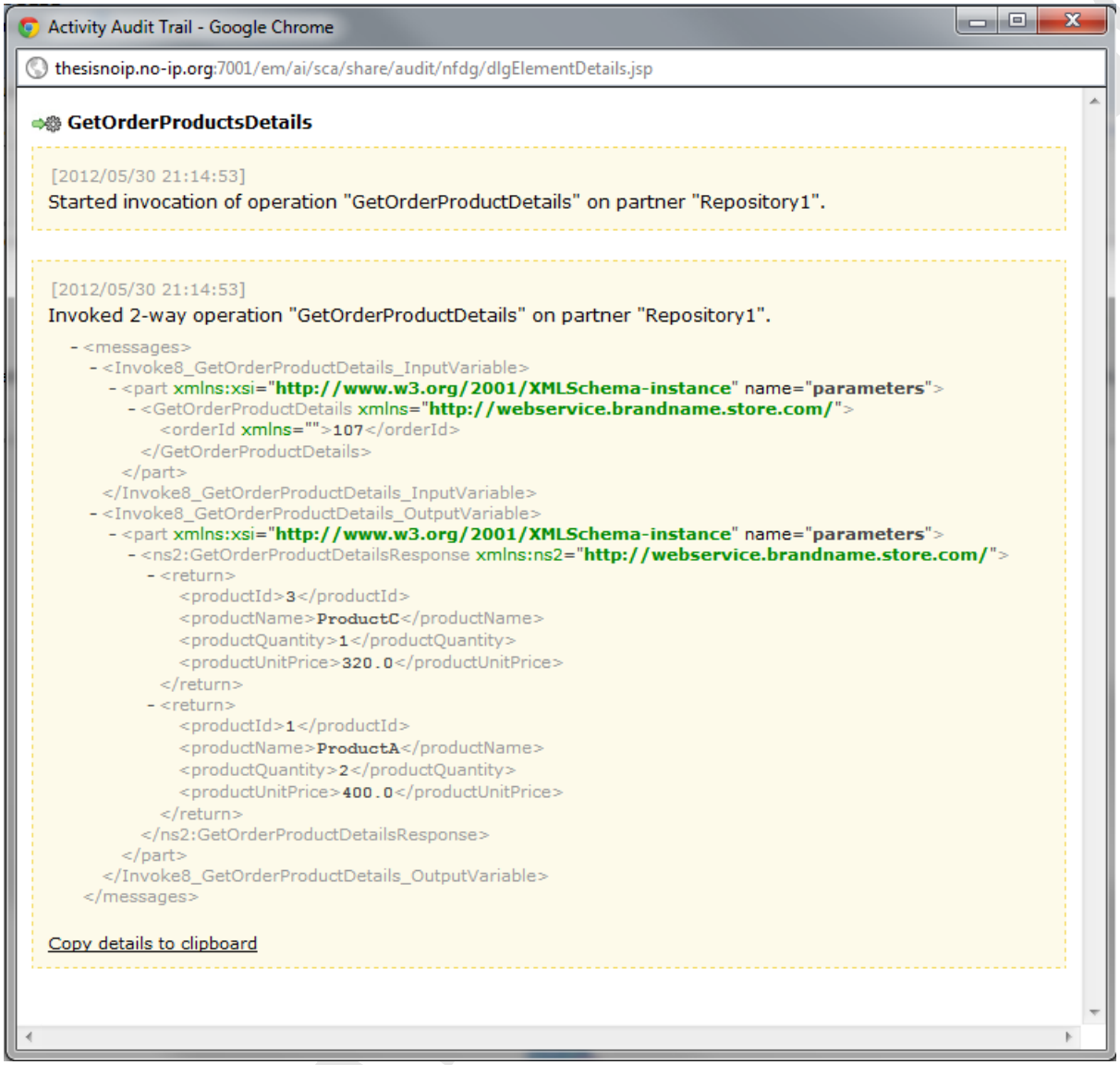
Σχήμα 61: BPEL Engine message Input/Output InsertReceipt

Επόμενο βήμα είναι η ενημέρωση της παραγγελίας και η αντιστοίχηση σε αυτή του αναγνωριστικού που επιστρέφεται από το προηγούμενο βήμα της δραστηριότητας. Αυτό ολοκληρώνεται με την κλήση της μεθόδου UpdateOrderReceiptId του Web service του Repository1 όπως φαίνεται και στο σχήμα 64 και μεν την ονομασία UpdateOrderInsertReceiptId στο σχήμα 74.

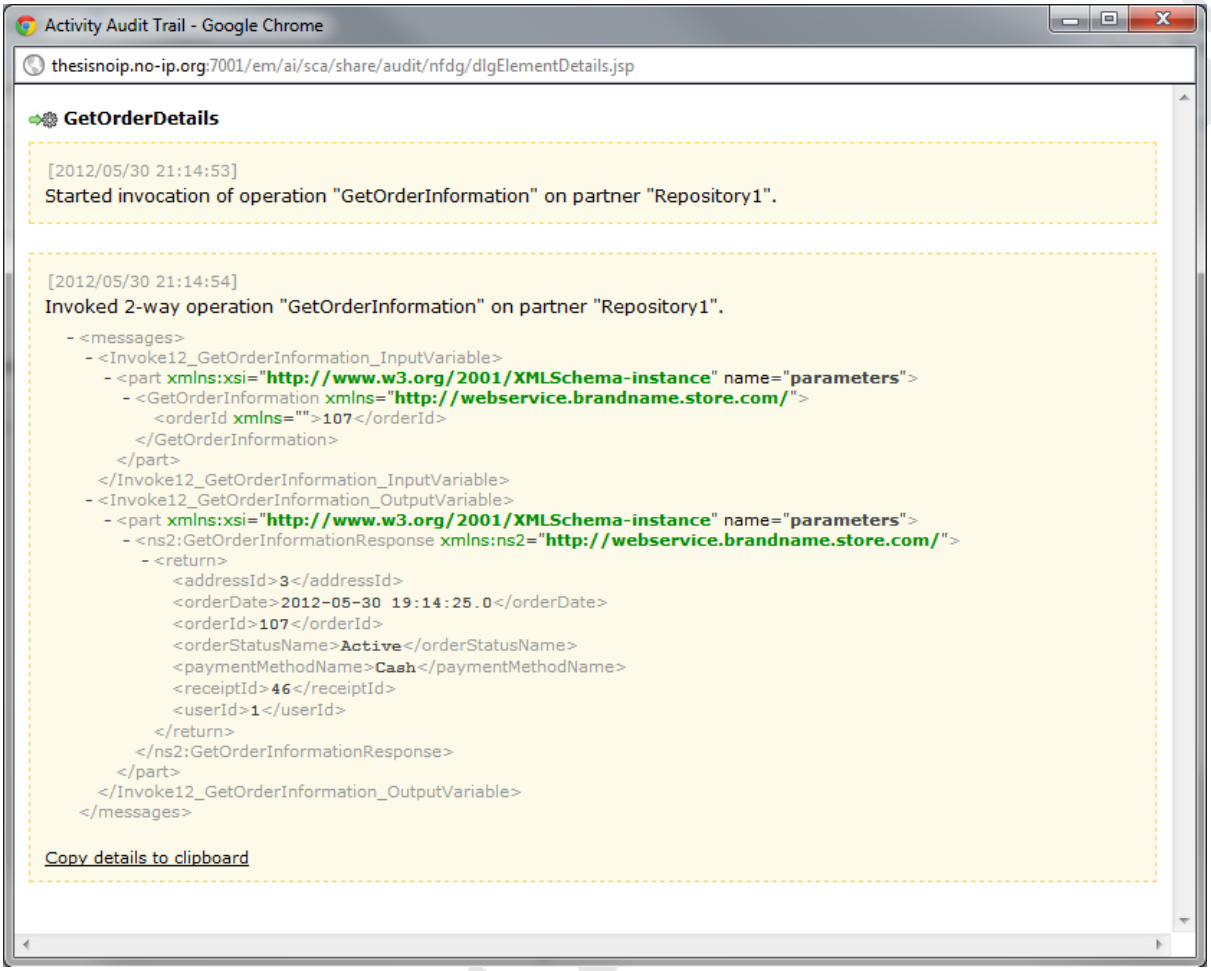


Σχήμα 62: BPEL Engine message Input/Output UpdateOrderInsertReceiptId

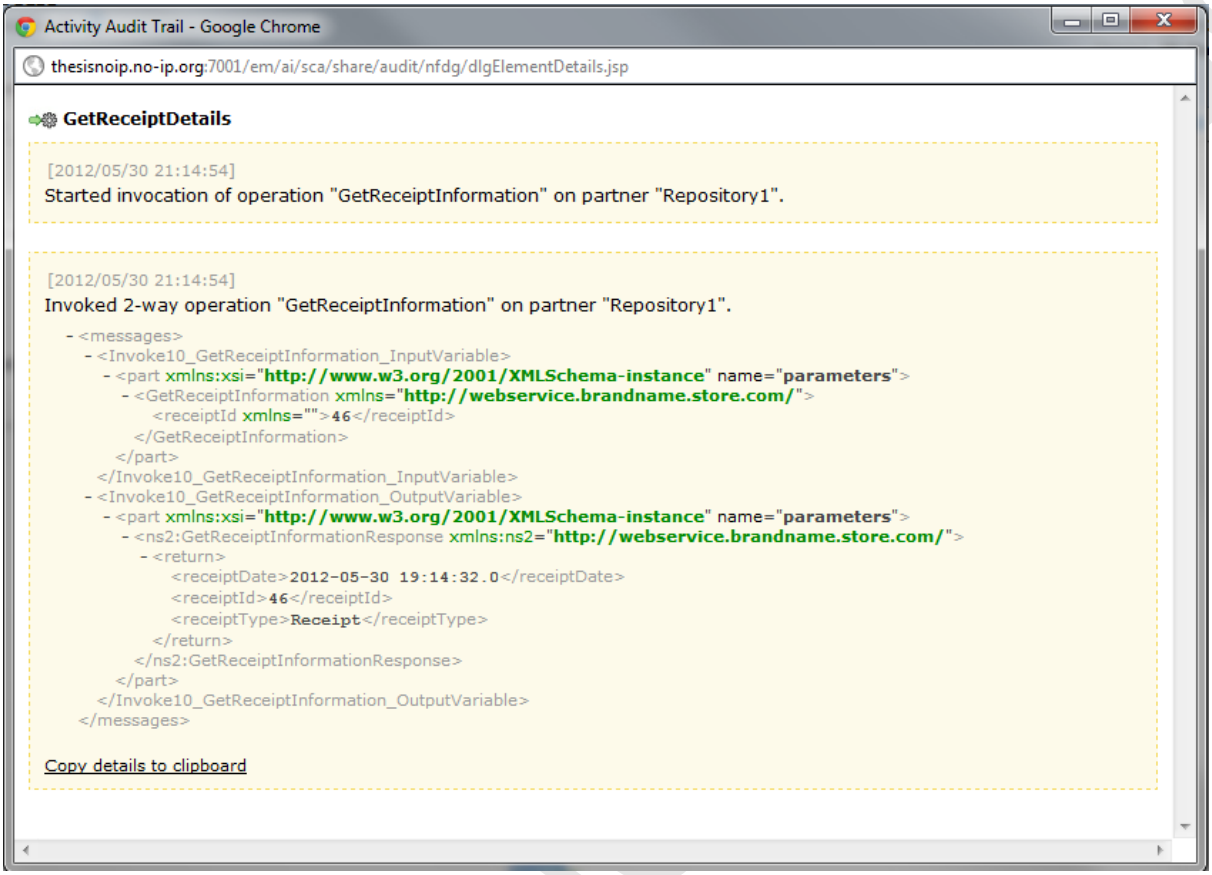
Μετά η διαδικασία μπαίνει σε μια παράλληλη ροή κατά την οποία εκτελούνται κλήσεις των Web service μεθόδων GetOrderProductDetails σχήμα 65, GetOrderInformation σχήμα 66, GetReceiptInformation σχήμα 67, GetUserInformation σχήμα 68, και GetAddressInformation σχήμα 69, του Repository1 οι δραστηριότητες <invoke> μπορούν να εντοπιστούν στο σχήμα 74 με τις ονομασίες GetOrderProductDetails, GetOrderDetails, GetReceiptDetails, GetUserDetails, GetAddressDetails.



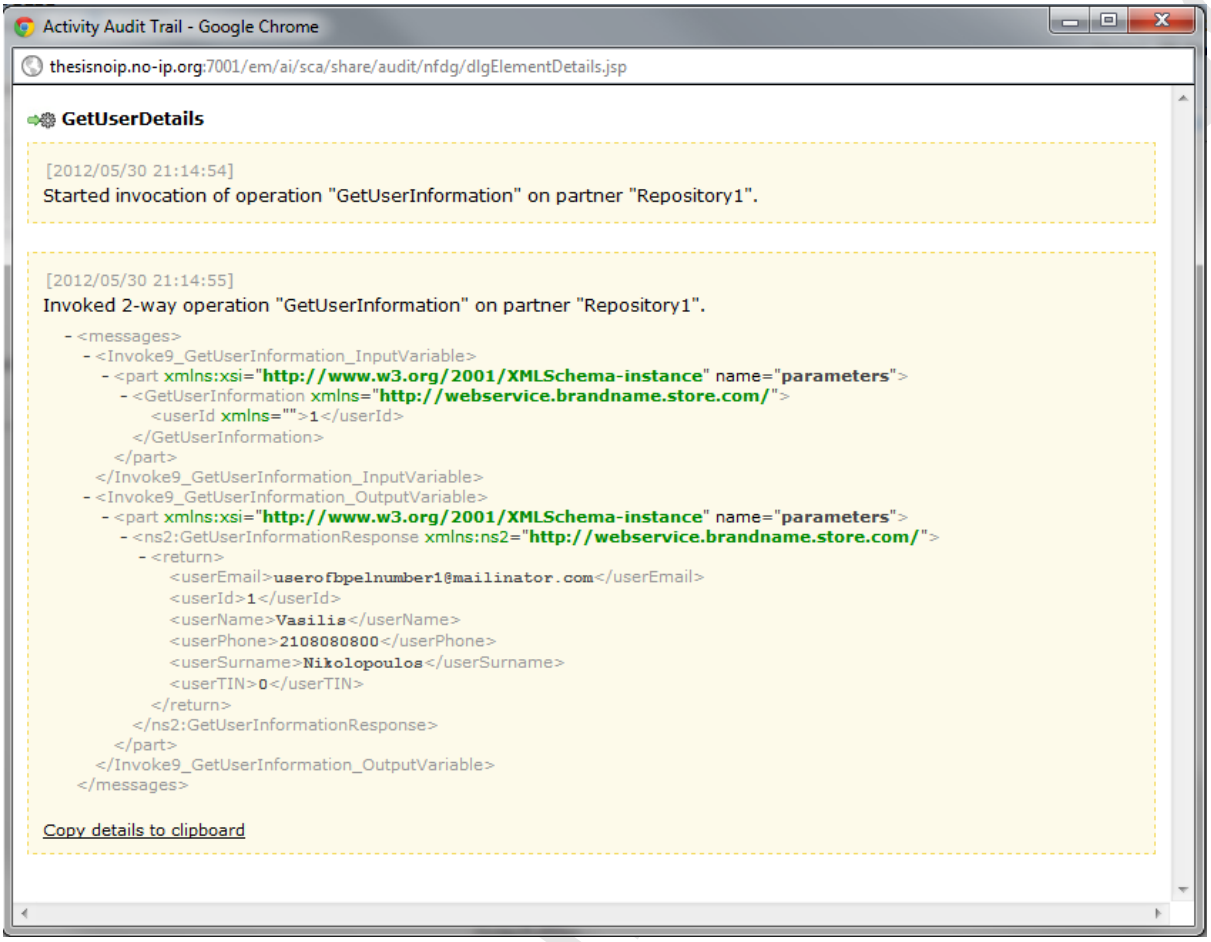
Σχήμα 63: BPEL Engine message Input/Output GetOrderProductDetails



Σχήμα 64: BPEL Engine message Input/Output GetOrderDetails

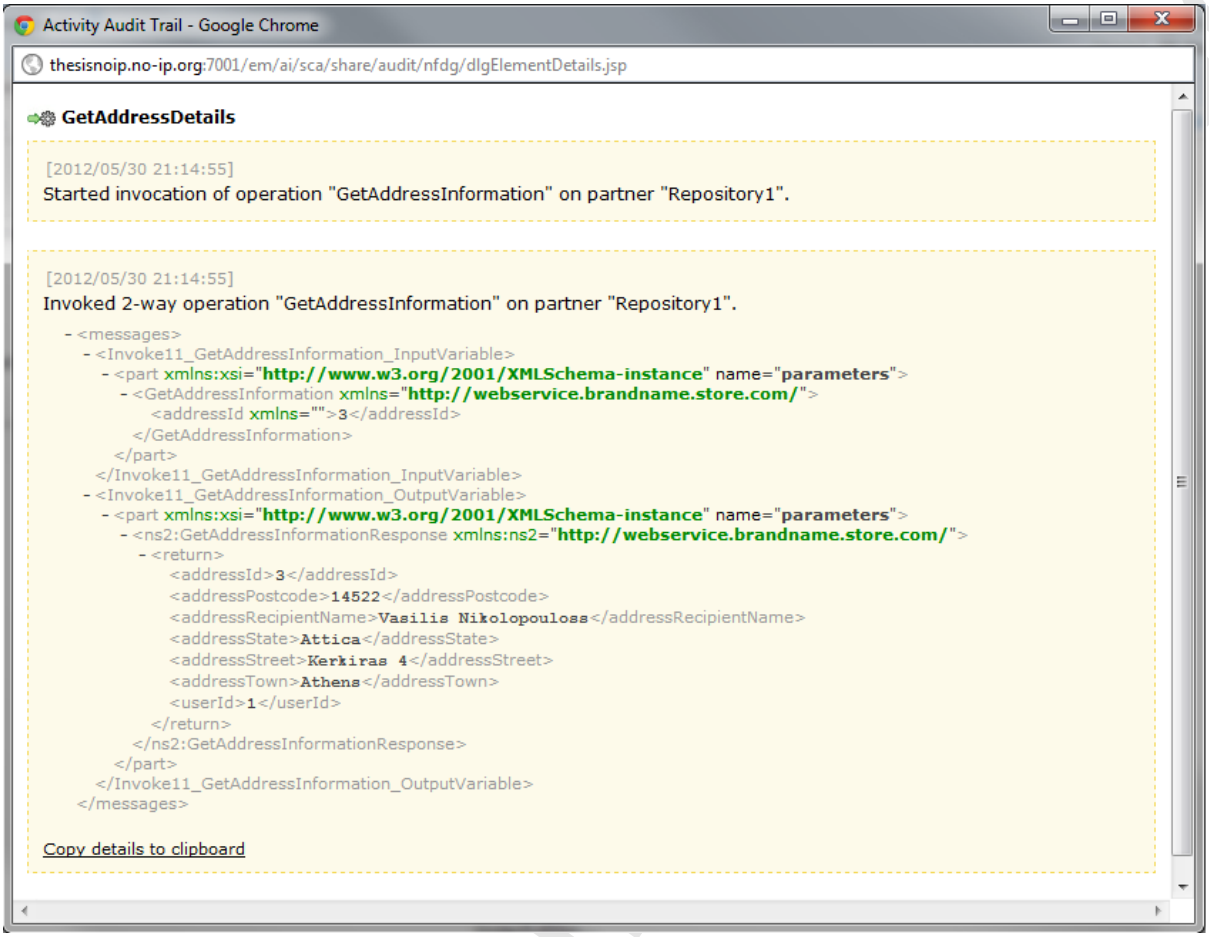


Σχήμα 65: BPEL Engine message Input/Output GetReceiptDetails



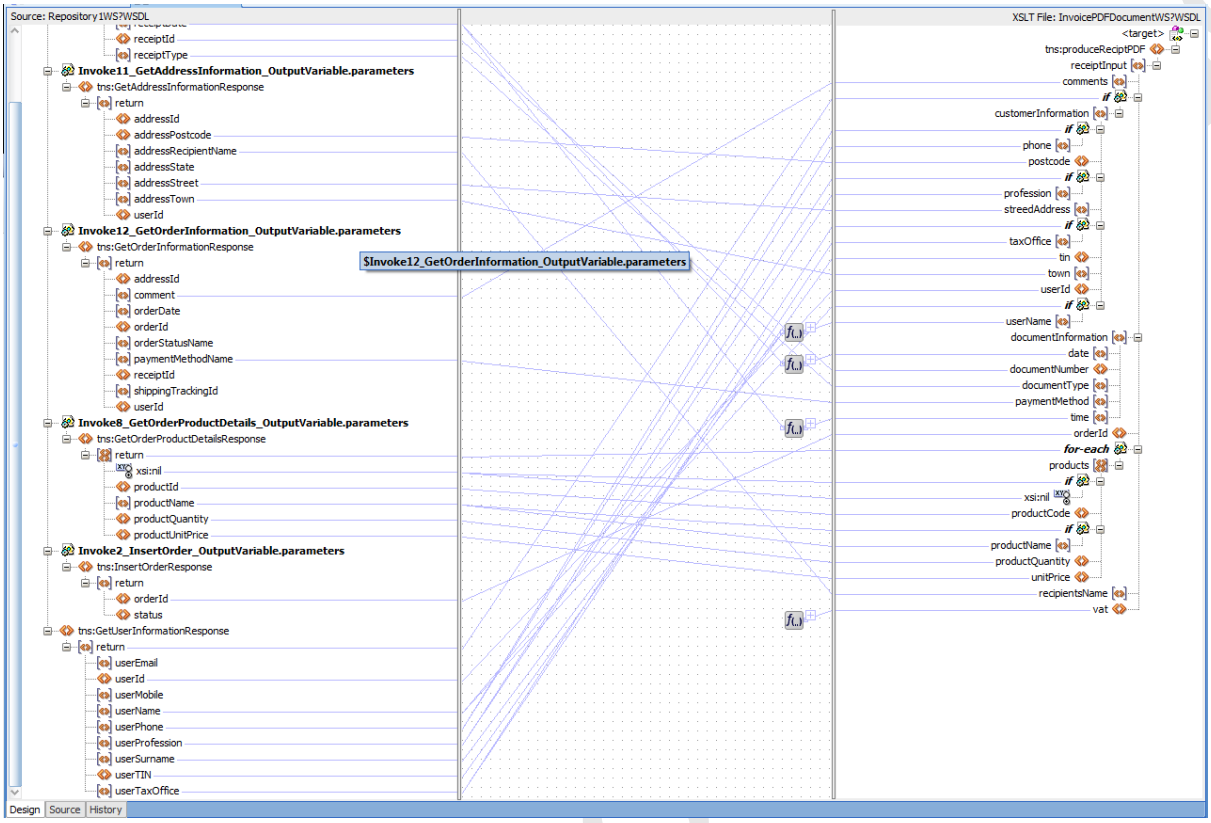
Σχήμα 66: BPEL Engine message Input/Output GetUserDetails

Πανεπιστήμιο Πειραιώς

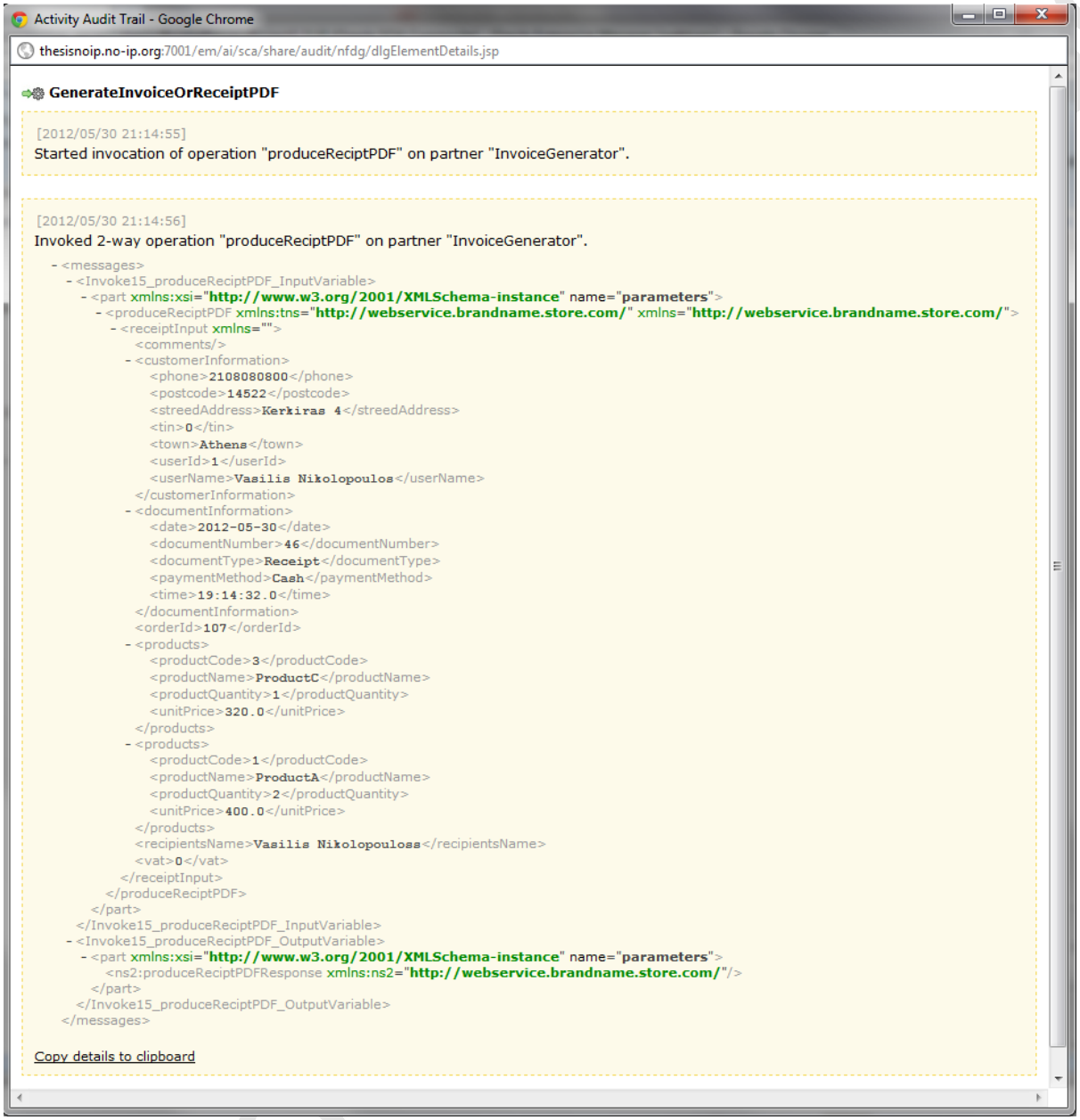


Σχήμα 67: BPEL Engine message Input/Output GetAdderssDetails

Οι έξοδοι των δραστηριοτήτων invoke συνδυάζονται και γίνεται manipulation τους μέσω ενός xsl αρχείου (σχήμα 72) για να διαμορφώσουν την κατάλληλη είσοδο για την κλήση της μεθόδου produceReceiptPDF του invoiceGenerator σχήμα 71, η δραστηριότητα μπορεί να βρεθεί στο σχήμα 74 με την ονομασία GenerateInvoiceOrReceiptPDF και δημιουργεί το PDF αρχείο του παραστατικού που φαίνεται στο σχήμα 97.

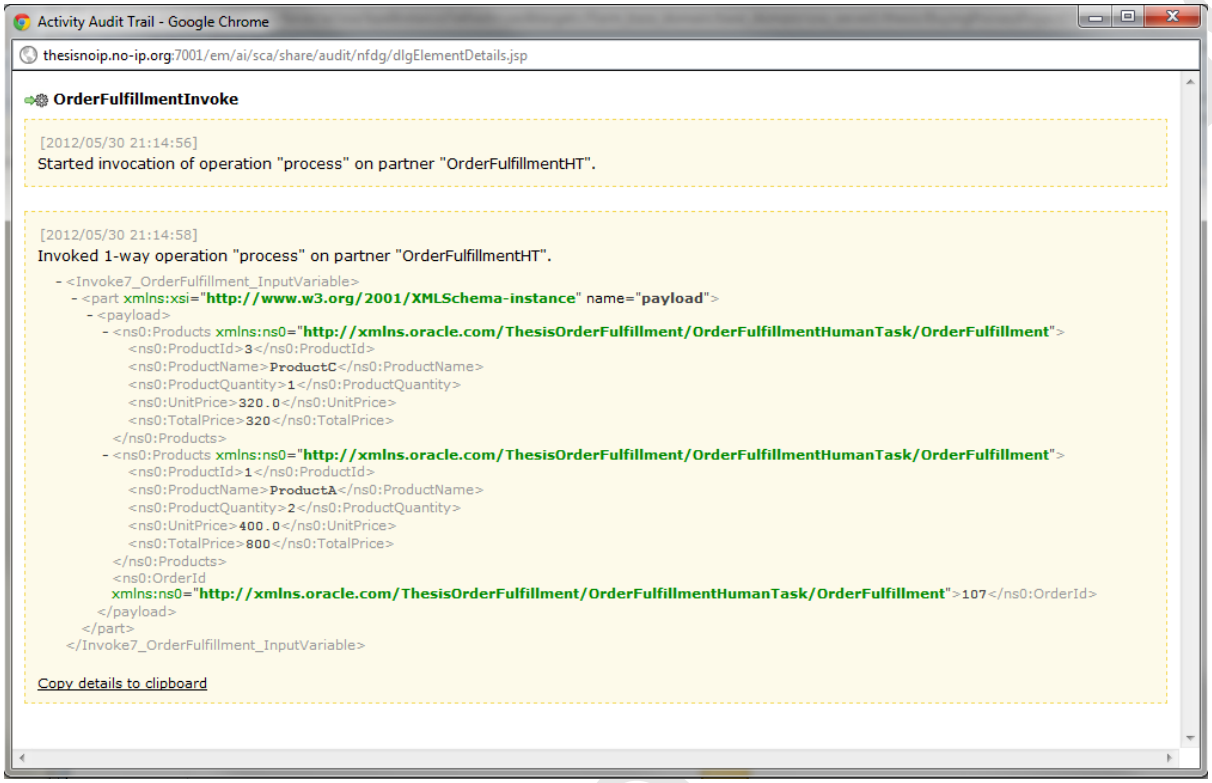


Σχήμα 68: Transformation XSL sheet για την απόδειξη



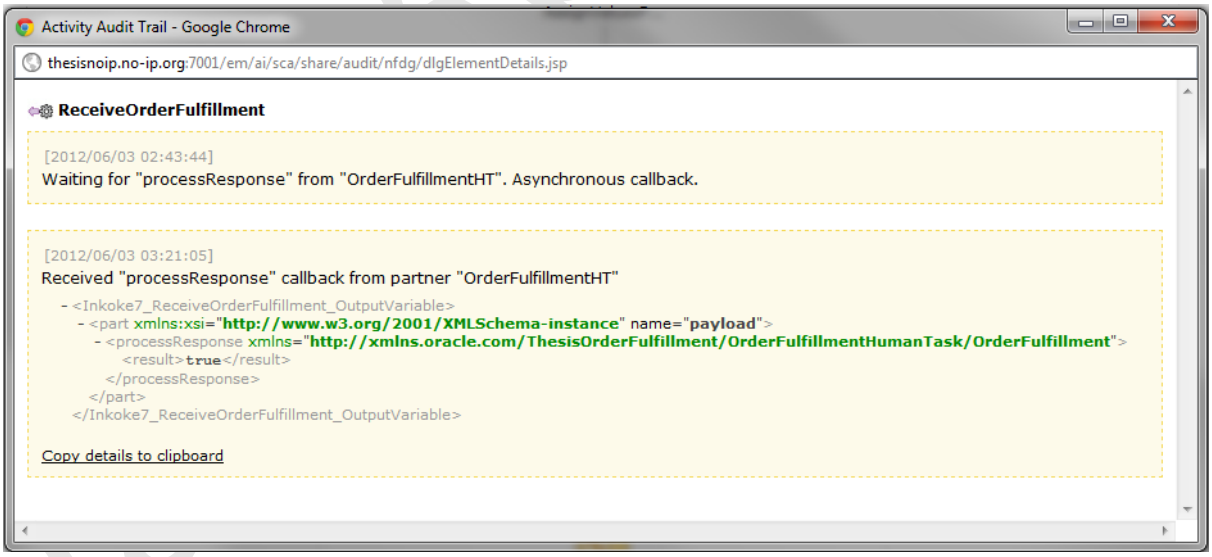
Σχήμα 69: BPEL Engine message Input/Output GenerateInvoiceOrReceiptPDF

Η διαδικασία αποστέλλει την παραγγελία στο Human Task που αποτελείται από δύο βήματα αυτό του ελέγχου και της επισύναψης της τιμολόγησης, και αυτό της διεκπεραίωσης της παραγγελίας, το Human Task φαίνεται στο σχήμα 83. Αυτό συμβαίνει με την ασύγχρονη κλήση της Web service μεθόδου process του OrderFulfillmentHT σχήμα 72 η οποία περιέχει τα στοιχεία της παραγγελίας που χρειάζονται για αυτή την διαδικασία, η δραστηριότητα αυτή εμφανίζεται στο σχήμα 74 με την ονομασία OrderFulfillmentInvoke.

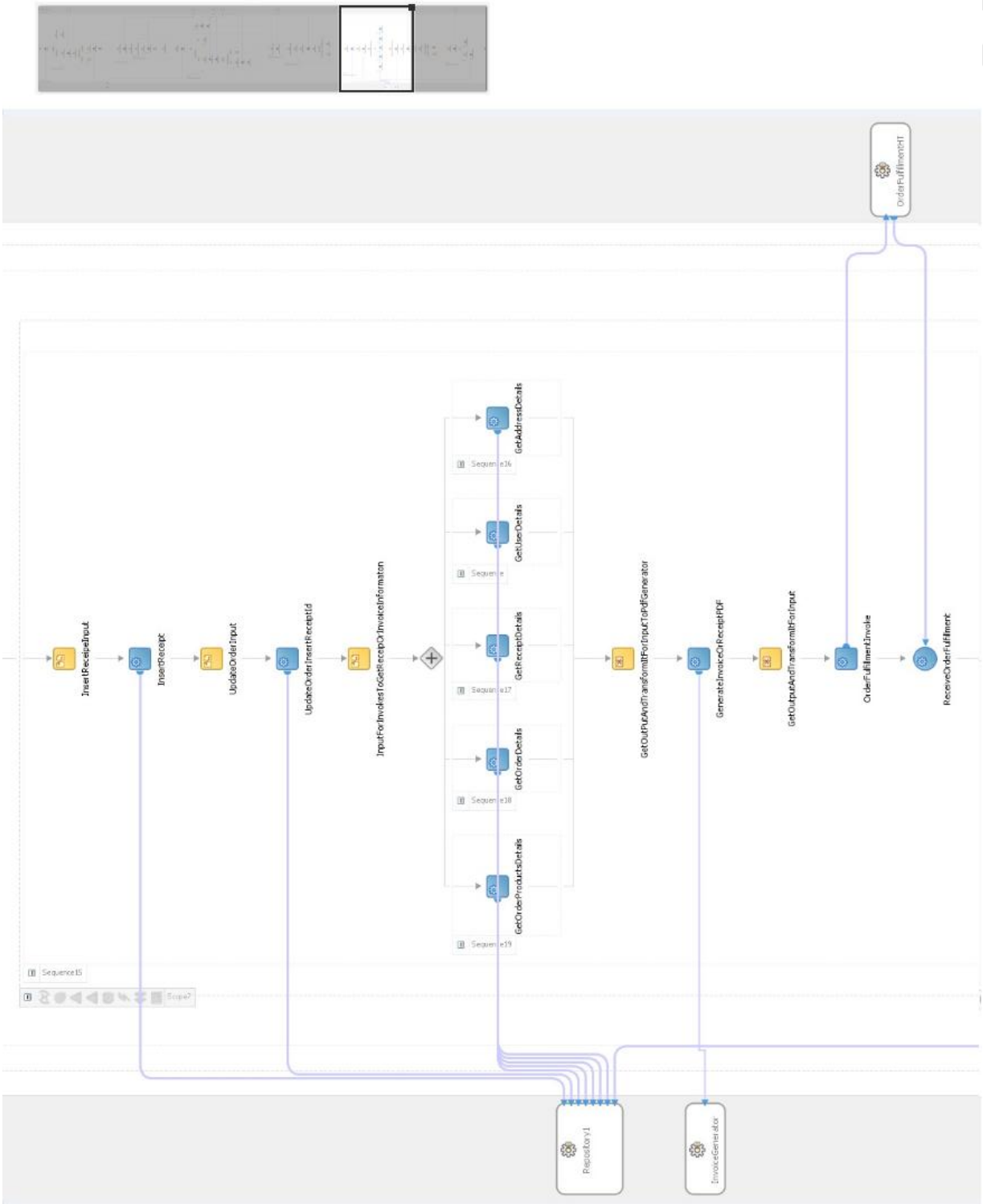


Σχήμα 70: BPEL Engine message Input OrderFulfillmentInvoke

Η απάντηση της ασύγχρονης επικοινωνίας επιστρέφεται στην δραστηριότητα <receive> ReceiveOrderFulfillment σχήμα 74.

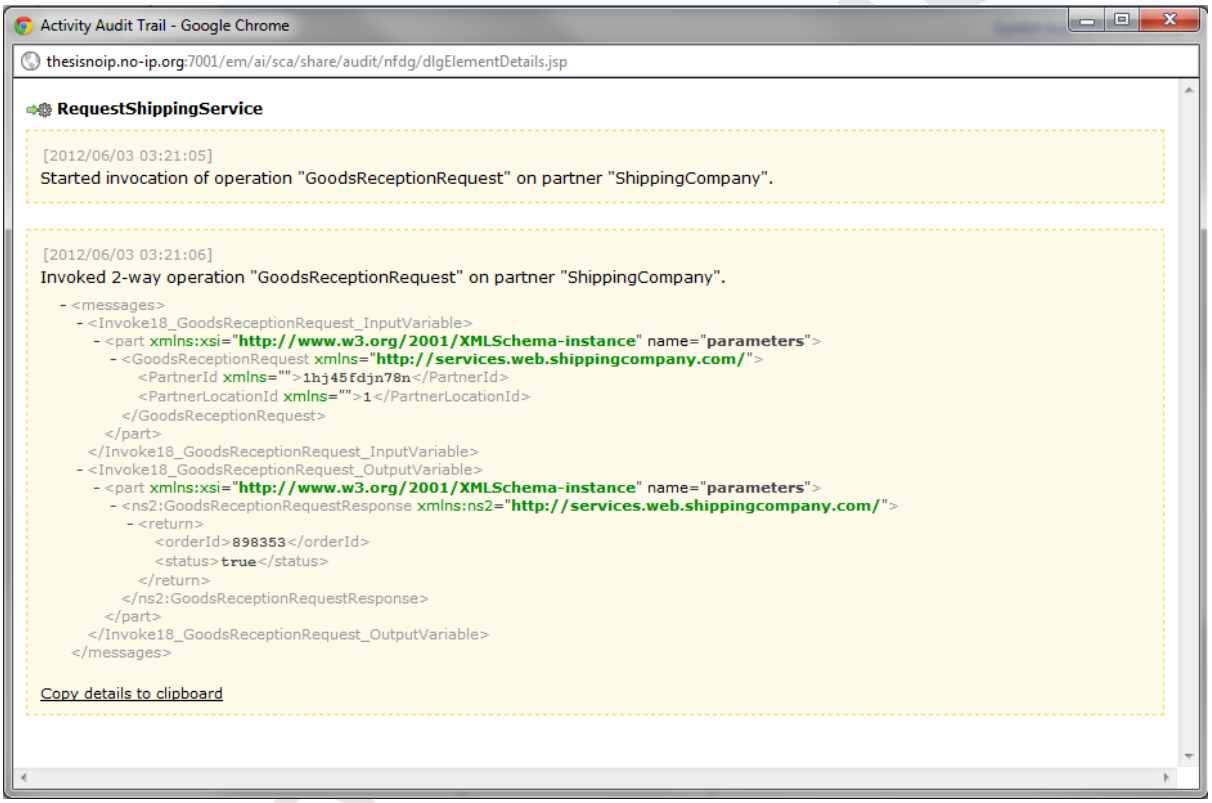


Σχήμα 71: BPEL Engine message Output ReceiveOrderFulfillment



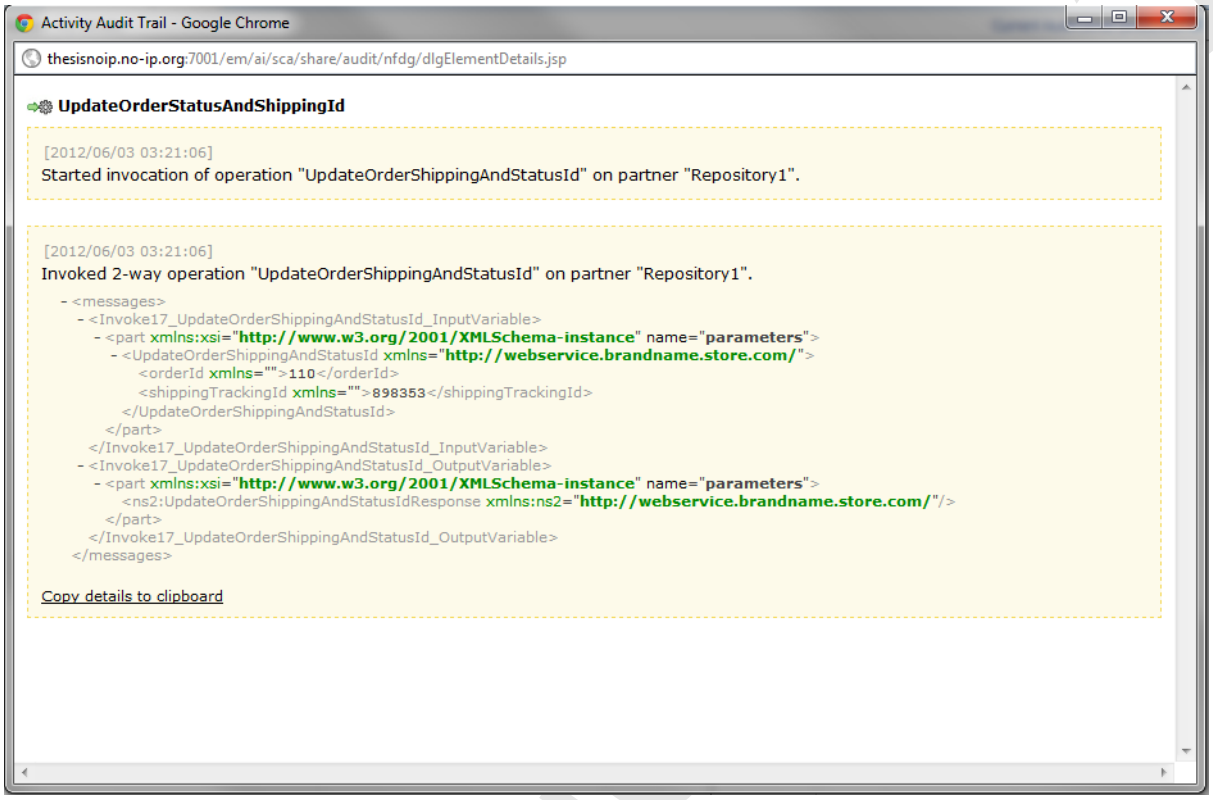
Σχήμα 72: Ανεπτυγμένη διαδικασία παραγγελίας (5/6)

Η διαδικασία συνεχίζει στο επόμενο βήμα που είναι ο έλεγχος του αν το Human Task ολοκληρώθηκε με επιτυχία, αν η ολοκλήρωση ήταν επιτυχής αν ήταν επιτυχής <result> true σχήμα 73, τότε η επόμενη δραστηριότητα που εκτελείται είναι μια κενή δραστηριότητα DoNothing3 σχήμα 78, ειδάλλως αν η εκτέλεση ήταν ανεπιτυχής <result> false, τότε η διαδικασία τερματίζεται και εκτελείται η δραστηριότητα <terminate> exit3 σχήμα 78. Το επόμενο βήμα της διαδικασίας είναι να αιτηθεί η μεταφορική εταιρία για την παραλαβή της παραγγελίας, αυτό θα συμβεί με την κλήση της Web service μεθοδου GoodsReceptionRequest του ShippingCompany σχήμα 75, η δραστηριότητα εμφανίζεται στο σχήμα 78 με την ονομασία RequestShippingService. Το Web service επιστρέφει αν εκτελέσθηκε επιτυχώς καθώς και το αναγνωριστικό της πράξης.



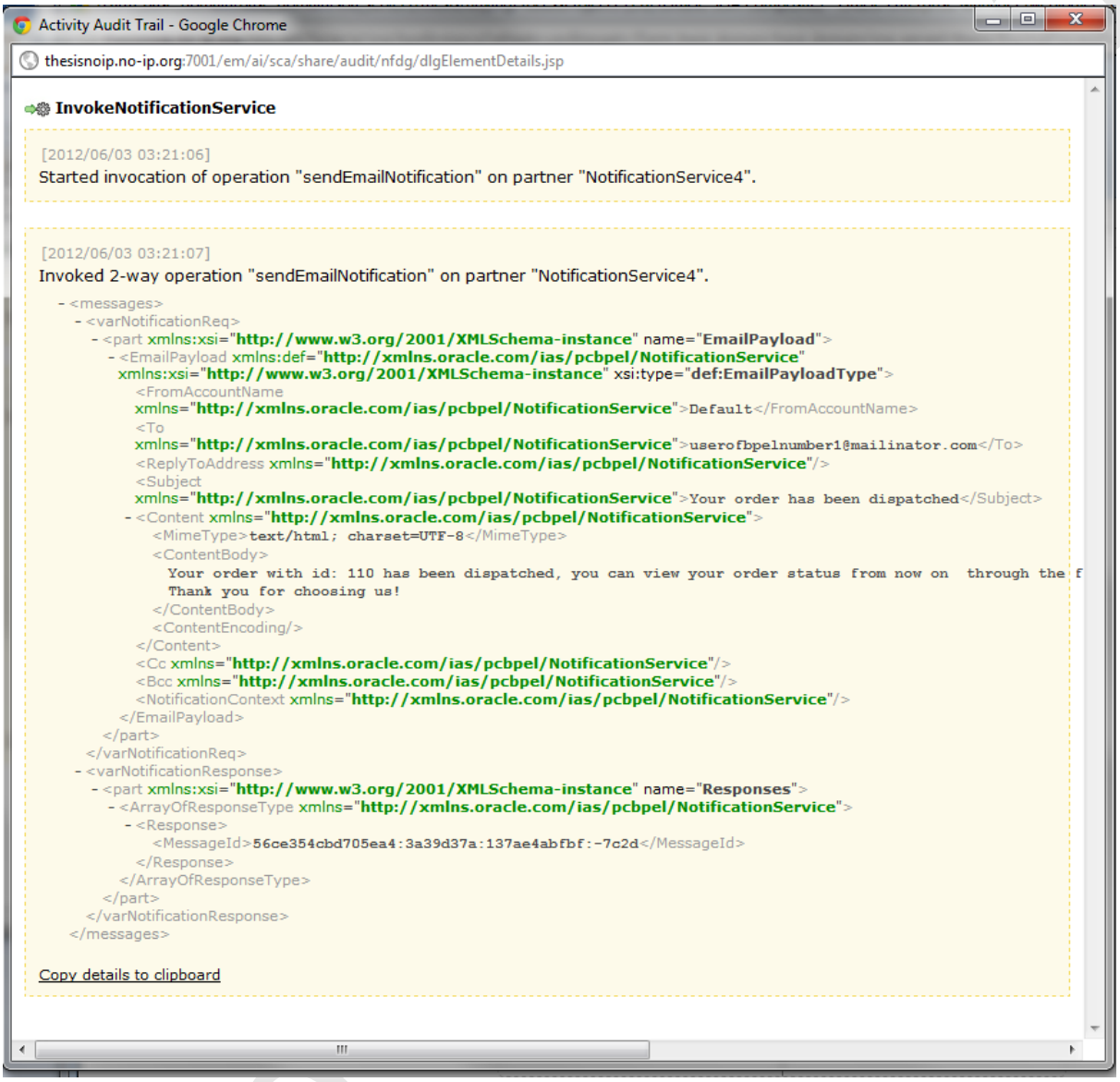
Σχήμα 73: BPEL Engine message RequestShippingService

Η διαδικασία εισέρχεται στην τελική φάση της και σε μια δραστηριότητας παράλληλης ροής <flow>. Στο αριστερό κομμάτι εκτελείται η ενημέρωση της βάσης δεδομένων για την κατάσταση της παραγγελίας και αντιστοιχίζεται στην παραγγελία το παραληφθέν αναγνωριστικό της προηγούμενης <invoke> δραστηριότητας από την μεταφορική εταιρία. Η ενημέρωση της βάσης επιτυγχάνεται μέσω της κλήσης της Web service μεθόδου UpdateOrderShippingAndStatusId του Repository1 σχήμα 76 και σχήμα 78.

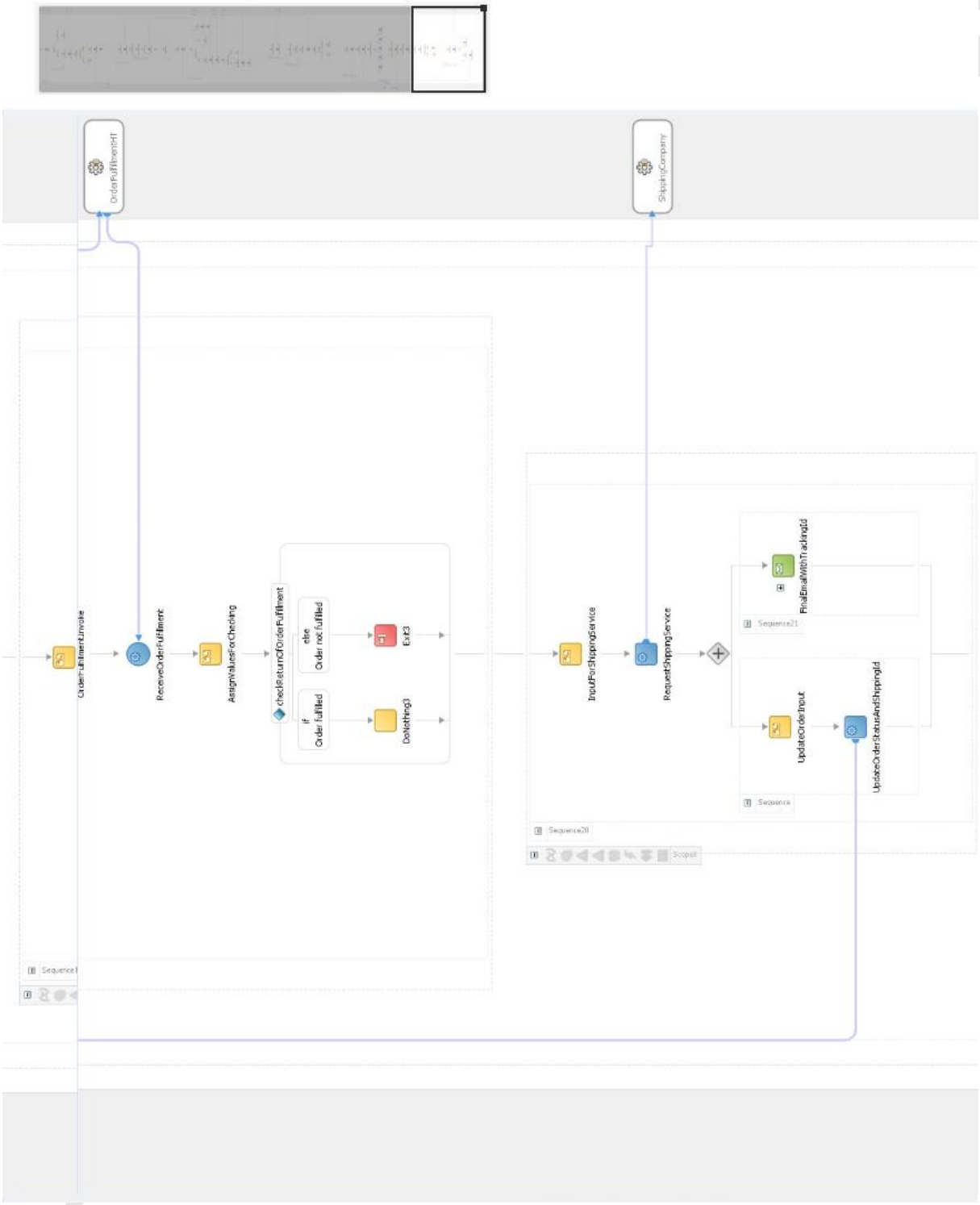


Σχήμα 74: BPEL Engine message Input/Output UpdateOrderStatusAndShippingId

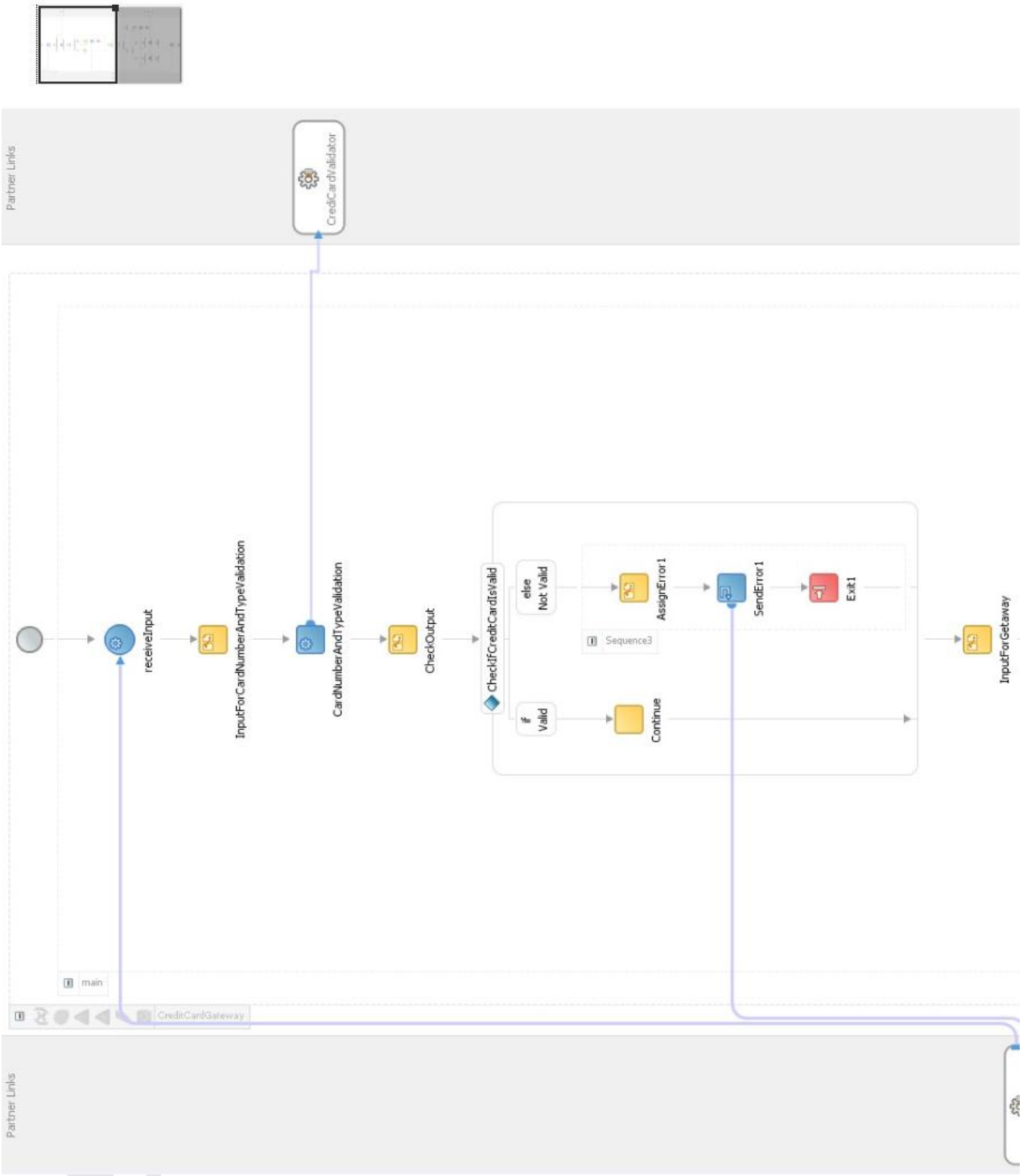
Στο δεξιό μέρος της παράλληλης ροής αποστέλλεται ενημερωτικό e-mail στον πελάτη της παραγγελίας μέσω component της Oracle Bpel Engine που αναφέρει ότι η παραγγελία έχει αποσταλεί και περιέχει ένα σύνδεσμο της μεταφορικής εταιρίας μέσω του οποίου μπορεί να παρακολουθηθεί η διαδικασία παράδοσης της παραγγελίας FinalEmailWithTrackingId σχήμα 78, το μήνυμα που αποστέλλεται στο mail component φαίνεται στο σχήμα 77, και παράδειγμα του email φαίνεται στο σχήμα 102.



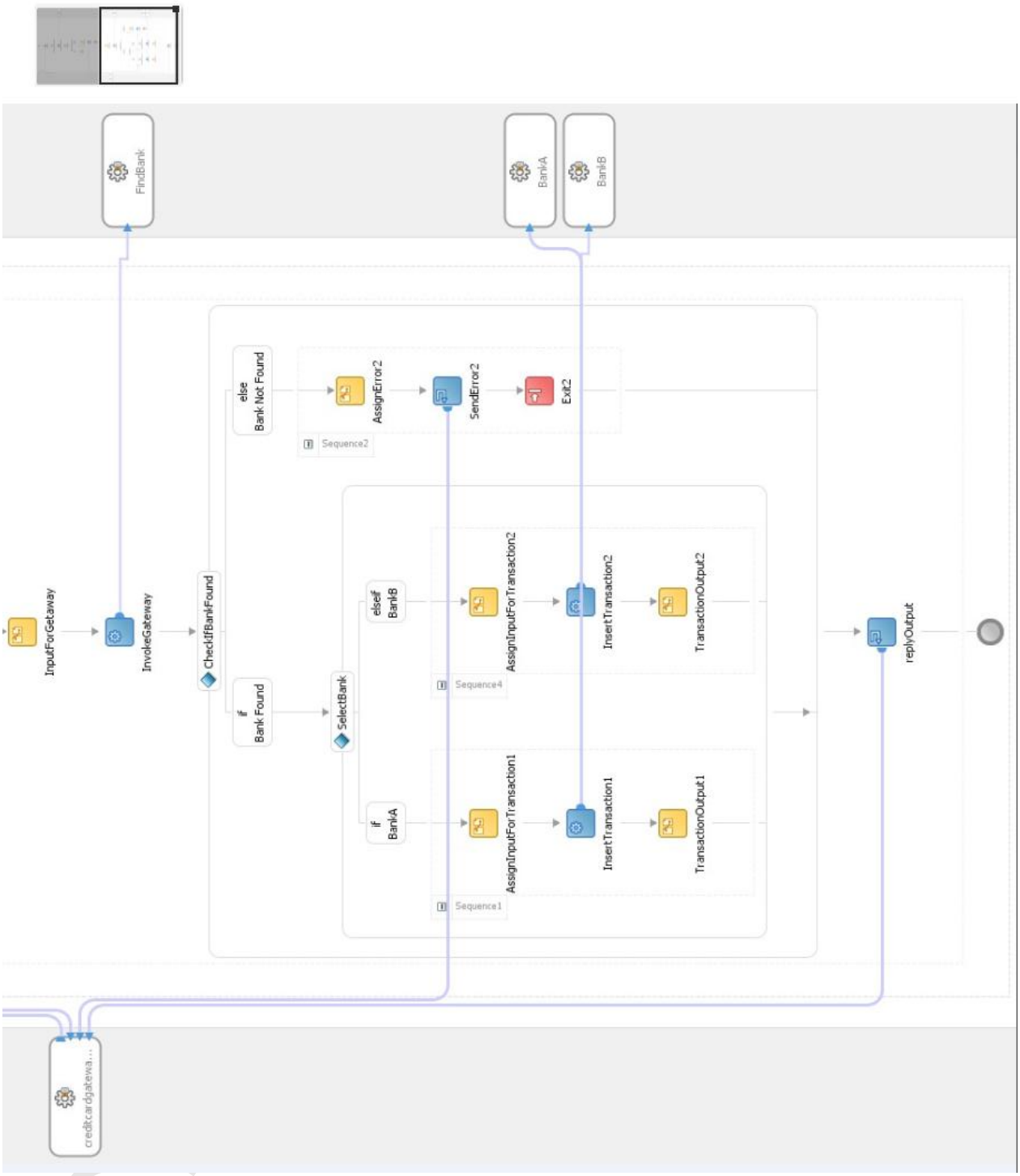
Σχήμα 75: BPEL Engine message Input/Output InvokeNotificationService



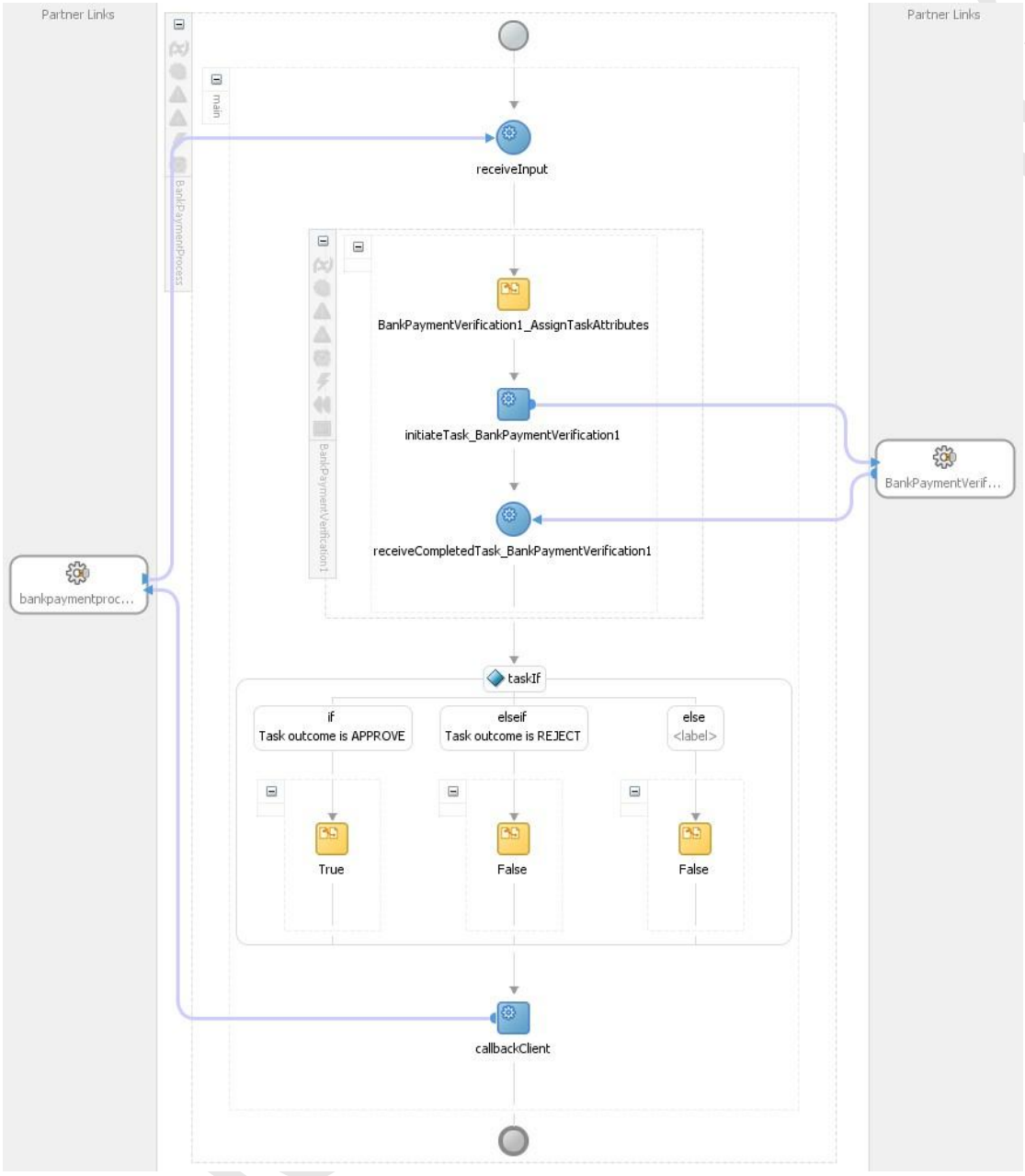
Σχήμα 76: Ανεπτυγμένη διαδικασία παραγγελίας (6/6)



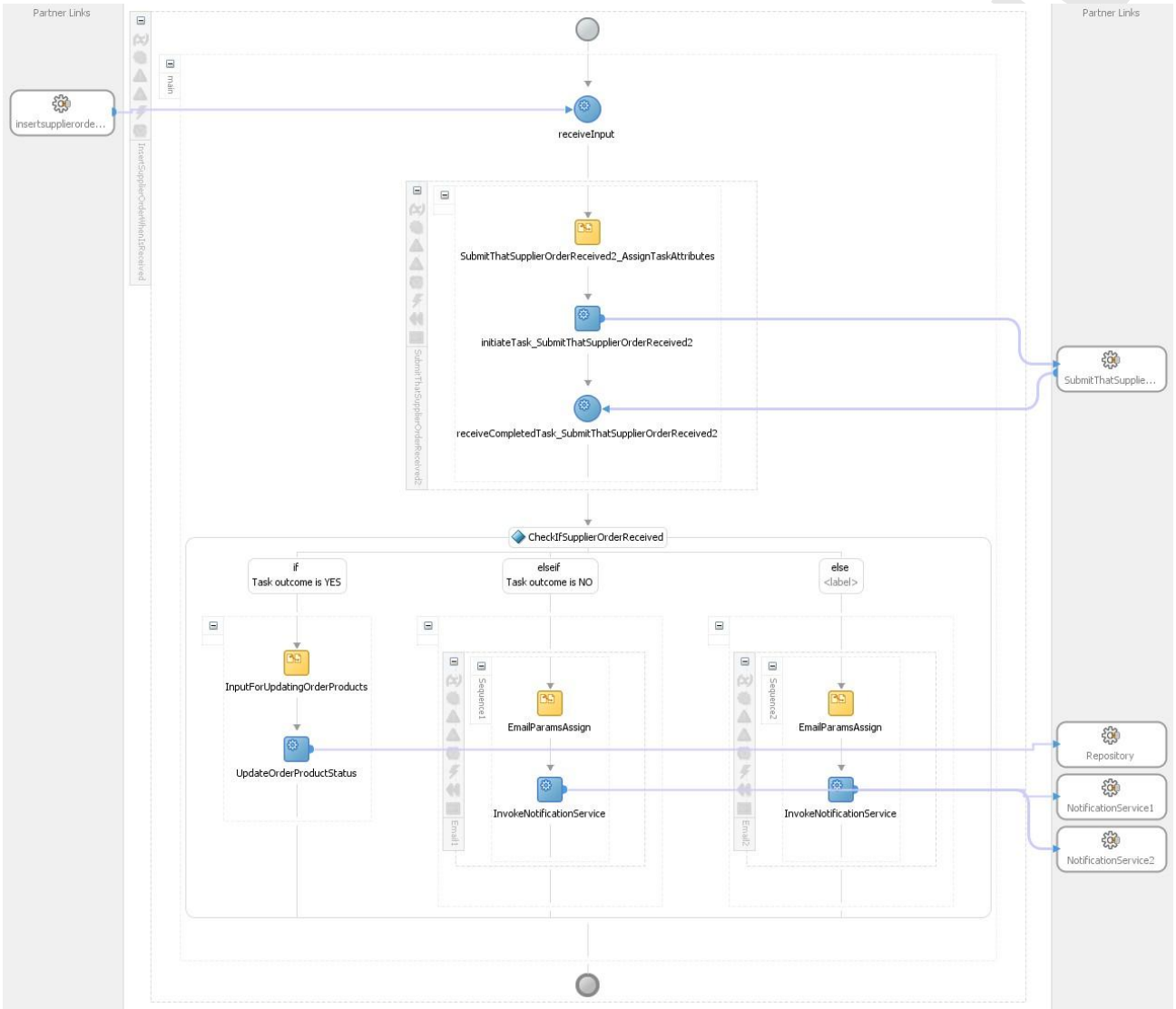
Σχήμα 77: Διαδικασία προσομοίωσης διεκπεραίωσης πληρωμής με πιστωτική(1/2)



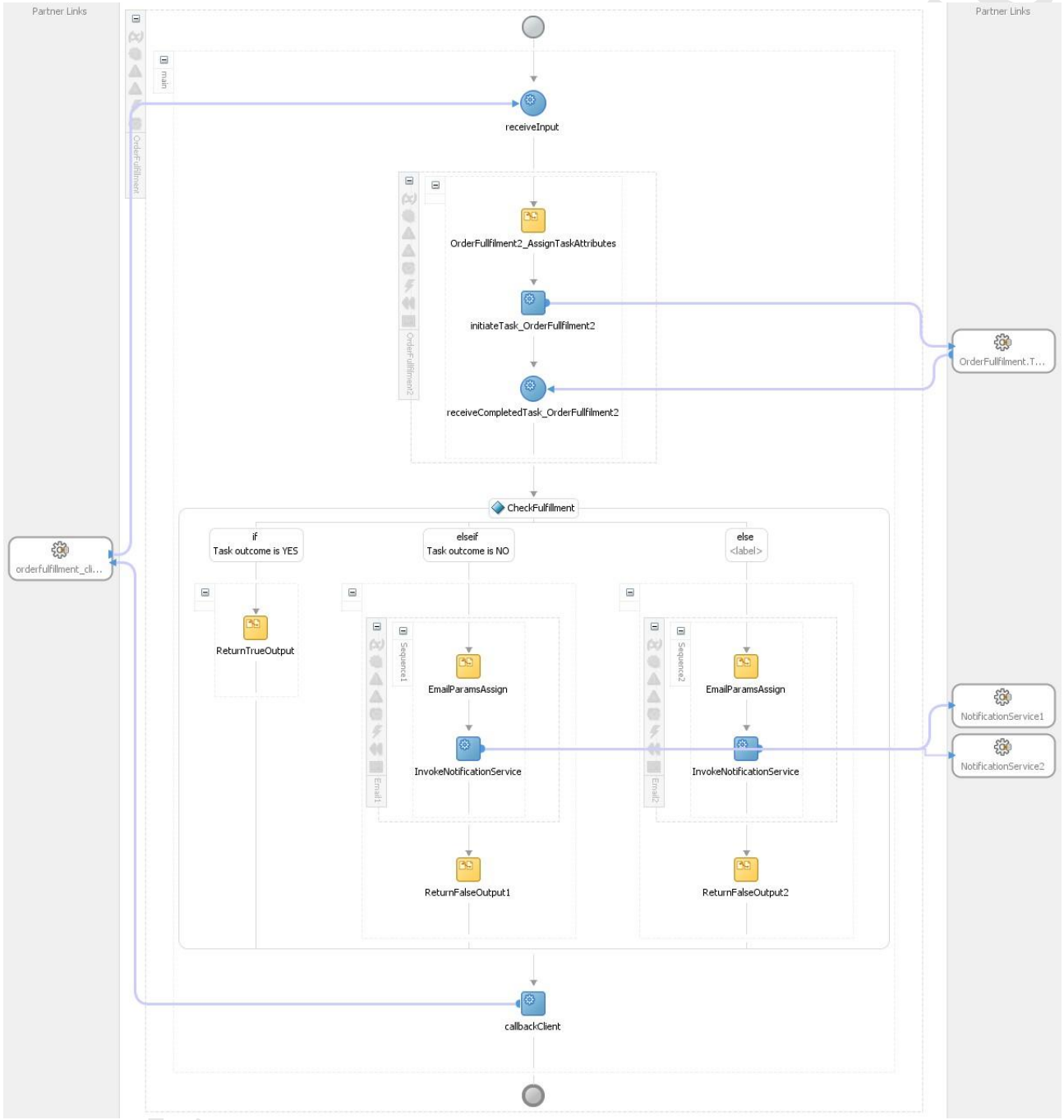
Σχήμα 78: Διαδικασία προσομοίωσης διεκπεραίωσης πληρωμής με πιστωτική(1/2)



Σχήμα 79: Human Task επιβεβαίωσης τραπεζικής πληρωμής



Σχήμα 80: Human Task παραλαβής προϊόντων από προμηθευτές



Σχήμα 81: Human Task έλεγχου τιμολόγησης και διεκπεραίωσης παραγγελίας

4.3.5 User Interface

Ένα στιγμιότυπο της διαδικασίας BPEL εκκινείτε όταν αποσταλεί ένα κατάλληλα μορφοποιημένο μήνυμα SOAP (σχήμα 84) στην μηχανή εκτέλεσης BPEL. Ένα παράδειγμα ενός μηνύματος SOAP που μπορεί να αποσταλεί για την εκκίνηση ενός στιγμιότυπου της εφαρμογής που υλοποιήθηκε για την παρούσα διπλωματική μπορεί να φανεί στον πίνακα 4 αποτελείται από τα εξής στοιχεία:

- Τις πληροφορίες του πελάτη (UserInformation), πιο αναλυτικά για τον καθένα πρέπει να περιλαμβάνει
    - Τον κωδικό πελάτη (UserId)
    - Και τον κωδικό της διεύθυνσης αποστολής (DeliveryAddressId)
- Την μέθοδο πληρωμής (PaymentMethodId)
- Τον τύπο παραστατικού/απόδειξης που θα εκδοθεί (OrderReceiptTypeId)
- Την τιμολόγηση της παραγγελίας (orderPrice)
- Τα προϊόντα της παραγγελίας (products), για το καθένα πρέπει να περιλαμβάνει:
    - Το κωδικό προϊόντος (productId)
    - Και την ποσότητα του (productQuantity)
- Και προαιρετικά, αν η επιλεγμένη μέθοδος πληρωμής είναι η πιστωτική κάρτα (creditCard) θα πρέπει να περιλαμβάνει τα στοιχεία της που είναι τα εξής:
    - Το μήνα λήξης της (cardExpirationMonth)
    - Το έτος λήξης της (cardExpirationYear)
    - Τον αριθμό της (cardNumber)
    - Τον τύπο της (cardType)
    - Το όνομα του κατόχου (cardUserName)
    - Και τον αριθμό επικύρωσης της (cardVerificationCode)

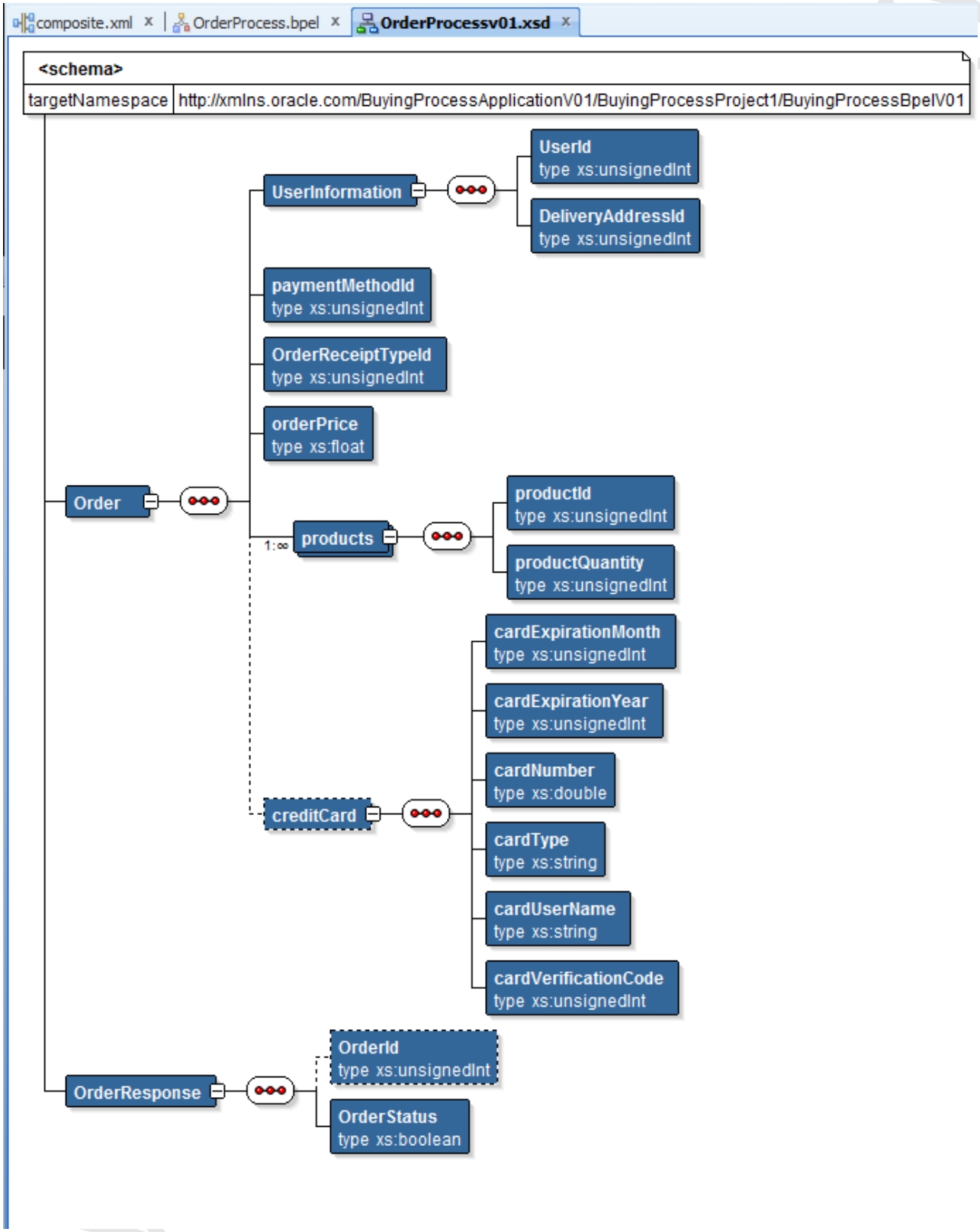
Η απάντηση της μηχανής BPEL που αποτελεί και της έξοδο της σύγχρονης διαδικασίας BPEL στο προαναφερθέν SOAP μήνυμα εκκίνησης μπορεί να φανεί στο πίνακα 5 και η δομή του προδιαγράφεται στο xsd του σχήματος 85 αποτελείται από τα εξής στοιχεία :

- Το αναγνωριστικό της παραγγελίας(OrderId)
- Και ένα Boolean αν καταχωρήθηκε σωστά η παραγγελία στο σύστημα (OrderStatus)

Πανεπιστήμιο Πειραιώς

Στην συγκεκριμένη περίπτωση η δεν υπήρξε σφάλμα (<OrderStatus>true</OrderStatus>) κατά την καταχώρηση της παραγγελίας και επιστράφηκε το αναγνωριστικό της (<OrderId>96</OrderId>).

Πανεπιστήμιο Πειραιώς



Σχήμα 82: OrderDetails.xsd

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:buy="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessPr
oject1/BuyingProcessBpelV01">
  <soapenv:Header/>
  <soapenv:Body>
    <buy:Order>
      <buy:UserInformation>
        <buy:UserId>1</buy:UserId>
        <buy:DeliveryAddressId>3</buy:DeliveryAddressId>
      </buy:UserInformation>
      <buy:paymentMethodId>3</buy:paymentMethodId>
      <buy:OrderReceiptTypeId>1</buy:OrderReceiptTypeId>
      <buy:orderPrice>1040</buy:orderPrice>
      <!--1 or more repetitions:-->
      <buy:products>
        <buy:productId>1</buy:productId>
        <buy:productQuantity>1</buy:productQuantity>
      </buy:products>
      <buy:products>
        <buy:productId>3</buy:productId>
        <buy:productQuantity>2</buy:productQuantity>
      </buy:products>
      <!--Optional:-->
      <buy:creditCard>
        <buy:cardExpirationMonth>?</buy:cardExpirationMonth>
        <buy:cardExpirationYear>?</buy:cardExpirationYear>
        <buy:cardNumber>?</buy:cardNumber>
        <buy:cardType>?</buy:cardType>
        <buy:cardUserName>?</buy:cardUserName>
        <buy:cardVerificationCode>?</buy:cardVerificationCode>
      </buy:creditCard>
    </buy:Order>
  </soapenv:Body>
</soapenv:Envelope>
```

Πίνακας 4: SOAP μήνυμα εκκίνησης διαδικασίας BPEL

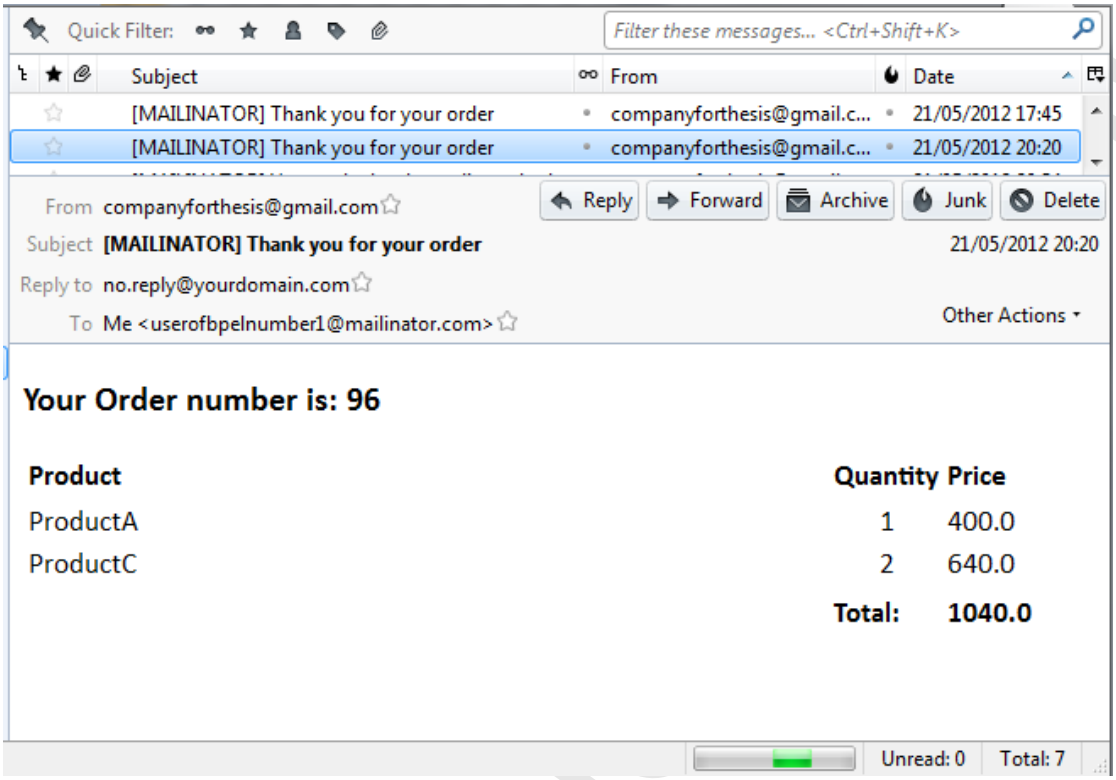
Πανεπιστήμιο Πειραιώς

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <env:Header>

<wsa:MessageID>urn:C4A7B440A2B911E1AF6931F104D8E7CE</wsa:MessageID
>
    <wsa:ReplyTo>

<wsa:Address>http://www.w3.org/2005/08/addressing/anonymous</wsa:Address>
    </wsa:ReplyTo>
  </env:Header>
  <env:Body>
    <OrderResponse
xmlns="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessProjec
t1/BuyingProcessBpelV01">
      <OrderId>96</OrderId>
      <OrderStatus>true</OrderStatus>
    </OrderResponse>
  </env:Body>
</env:Envelope>
```

Πίνακας 5: Απαντητικό SOAP μήνυμα διαδικασίας BPEL

Πανεπιστήμιο Πειραιώς

Στον πελάτη της παραγγελίας αποστέλλεται e-mail με τις λεπτομέρειες της, **σχήμα**.



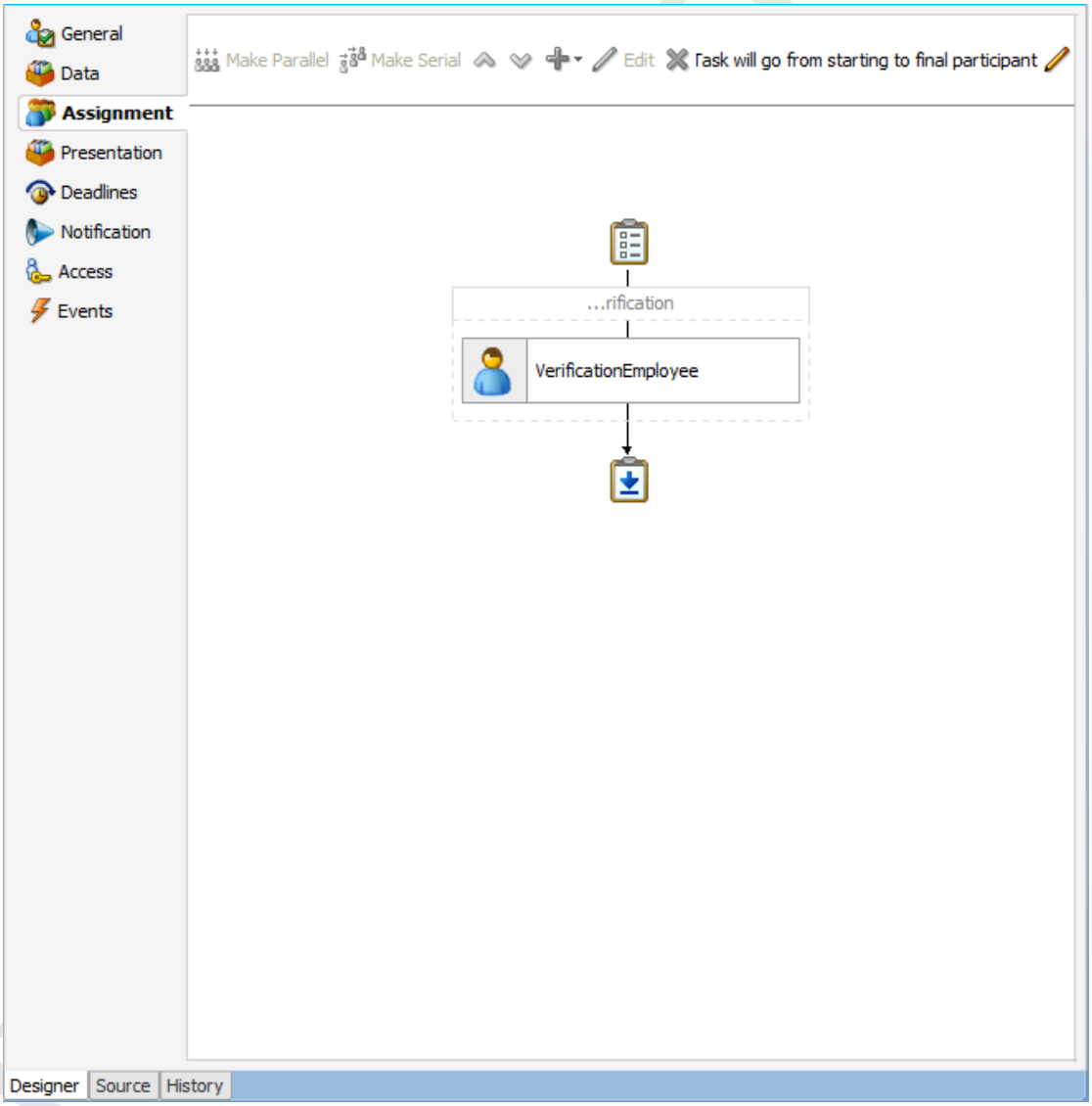
Σχήμα 83: Ενημερωτικό email με τις λεπτομέρειες της παραγγελίας.

4.3.5.1 Human Tasks

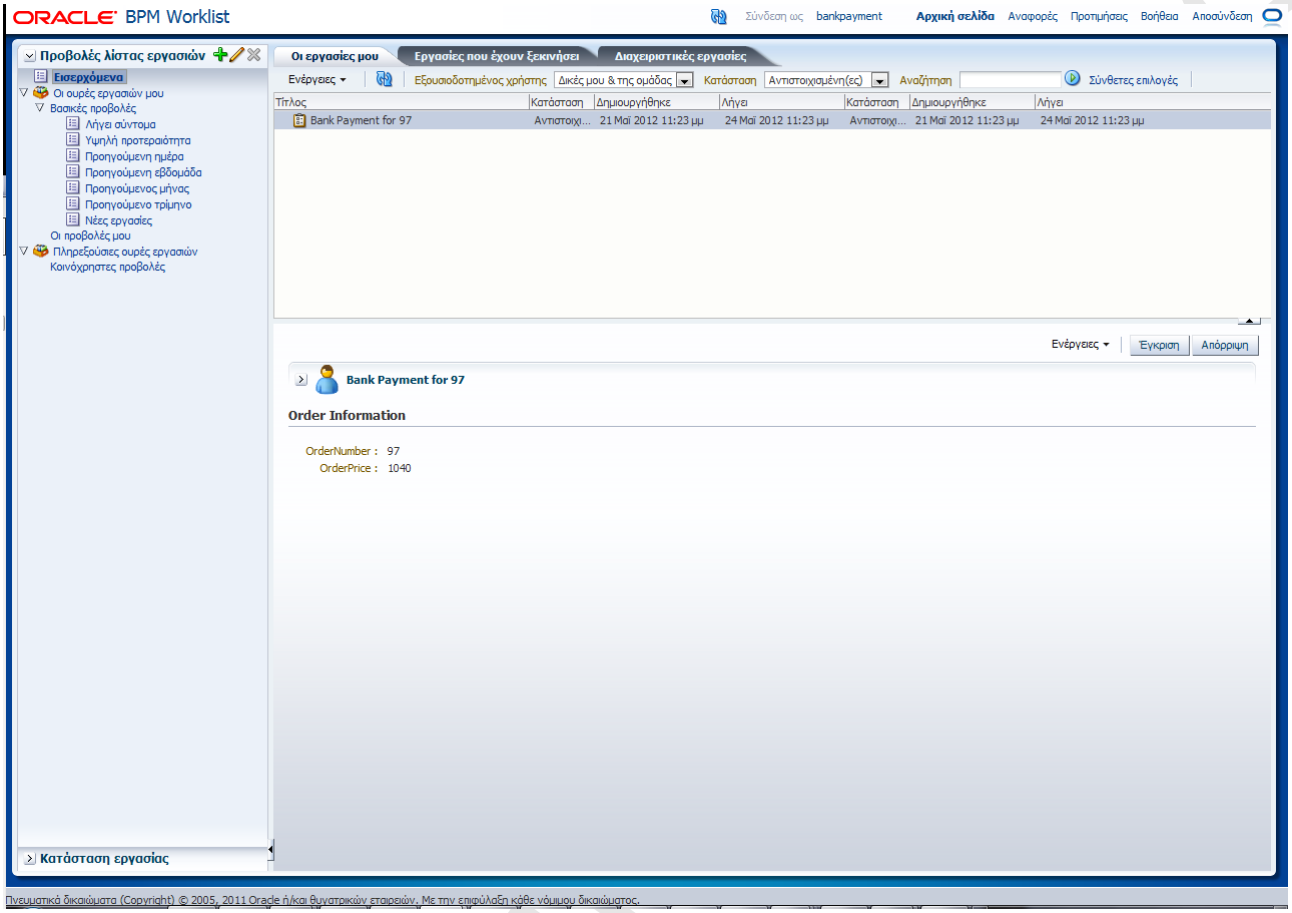
Ακολουθεί ανάλυση των ανθρώπινων αλληλεπιδράσεων με την διαδικασία BPEL.

4.3.4.1.1 Επιβεβαίωση Τραπεζική κατάθεσης

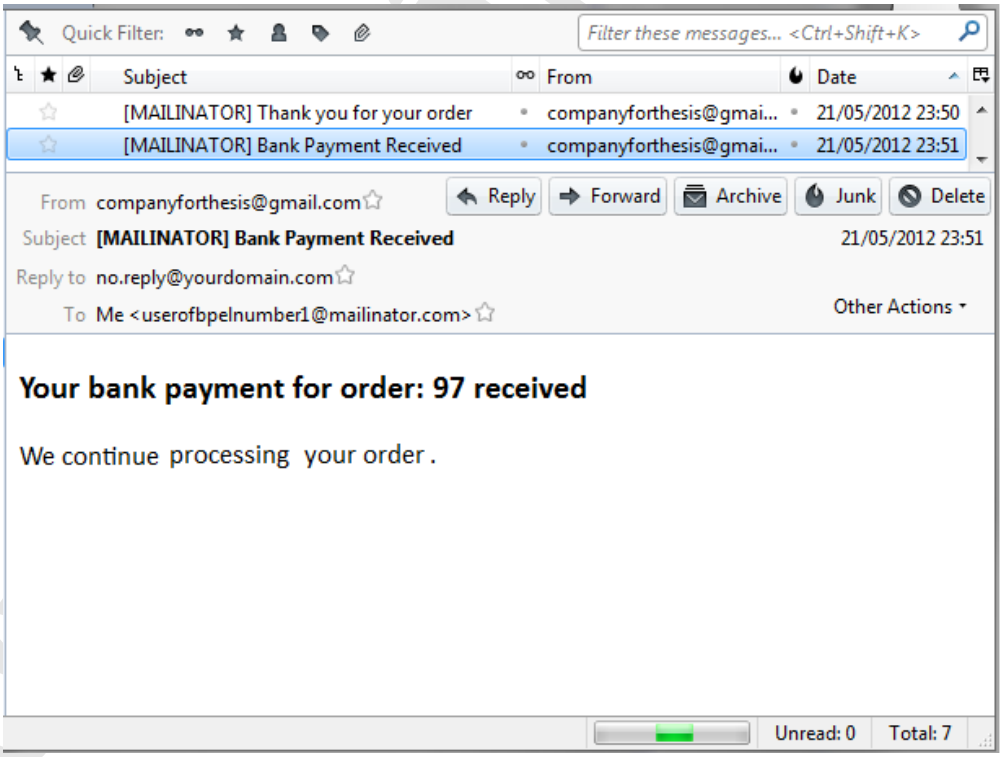
Το Human Task του σχήματος 86 αποτελείται από ένα στάδιο, το οποίο είναι η επιβεβαίωση της τραπεζικής κατάθεσης. Ένα υπάλληλος του λογιστηρίου σχήμα 87 μπορεί να εισέλθει στο σύστημα (σύνδεση ως bankpayment) και να επιβεβαιώσει (επιλέγοντας το "Έγκριση" από τις δύο Ενέργειες) ότι έγινε η κατάθεση του χρηματικού ποσού που αντιστοιχεί σε μια παραγγελία, η διαδικασία μετά από αυτό μπορεί να προχωρήσει στα επόμενα βήματα που είναι η αποστολή ενημερωτικού email στον πελάτη της παραγγελίας σχήμα 88 και ο έλεγχος της διαθεσιμότητας των προϊόντων αυτής.



Σχήμα 84: Human Task επιβεβαίωσης τραπεζικής κατάθεσης



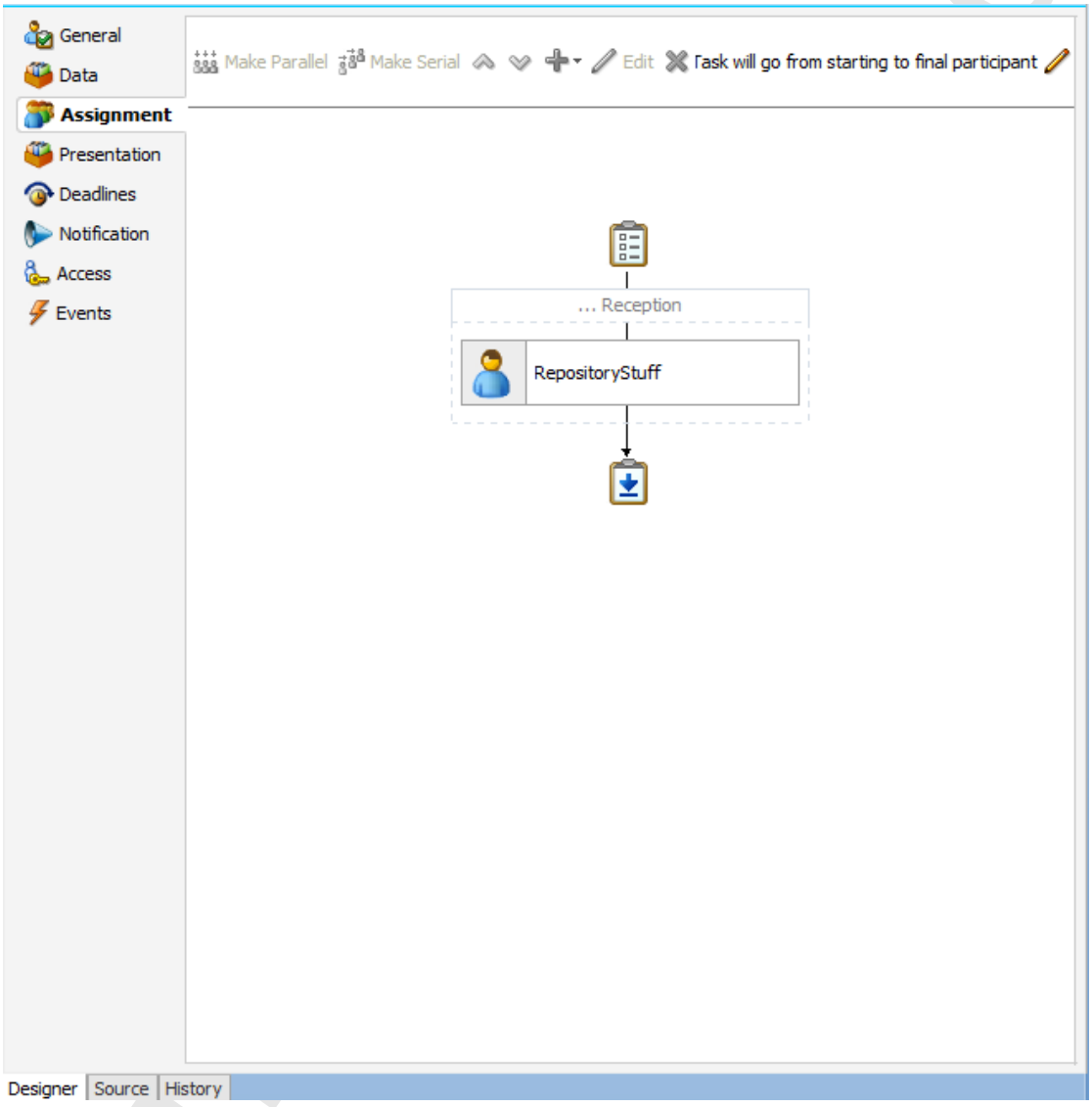
Σχήμα 85: Γραφικό περιβάλλον επιβεβαίωσης τραπεζικής κατάθεσης



Σχήμα 86: Ενημερωτικό email λήψης τραπεζικής κατάθεσης

4.3.4.1.2 Επιβεβαίωση παραλαβής προϊόντων από προμηθευτές

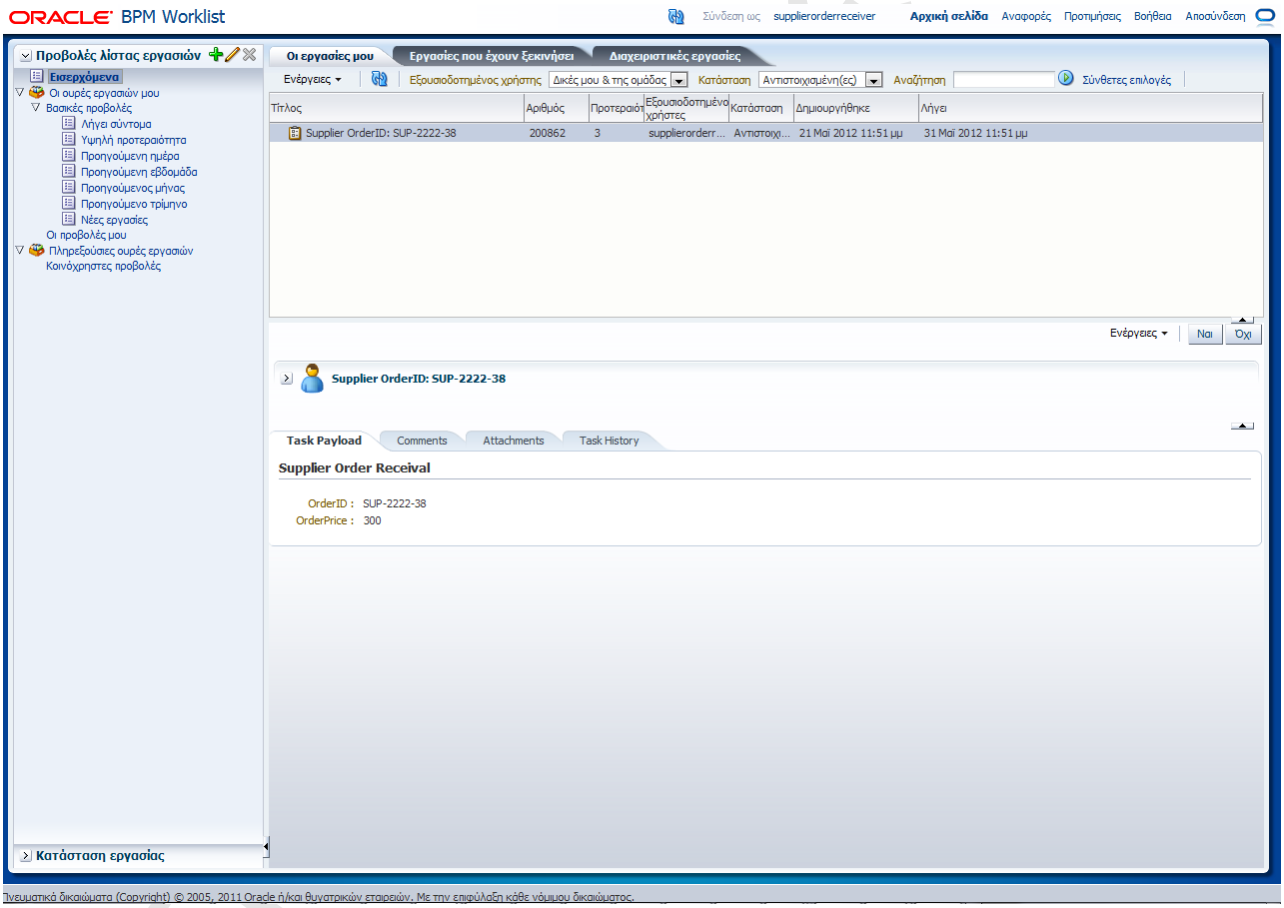
Το Human Task που φαίνεται στο σχήμα 89 όπως και το προηγούμενο αποτελείται από ένα στάδιο, το οποίο είναι η επιβεβαίωση της λήψης ενός προϊόντος από κάποιον προμηθευτή.



Σχήμα 87: Human Task επιβεβαίωσης παραλαβής προϊόντων από προμηθευτές

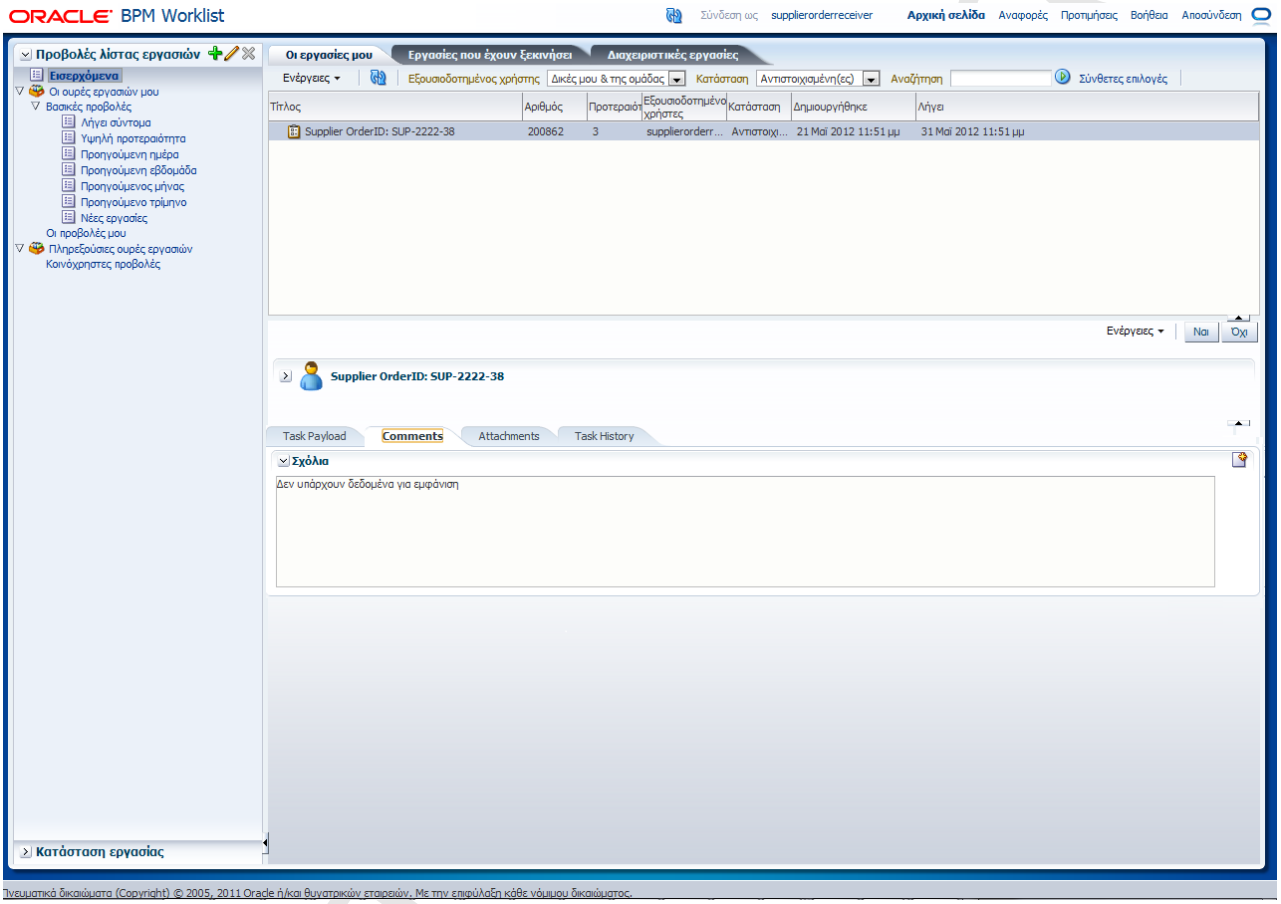
Ένα υπάλληλος της αποθήκης σχήμα 90 **(**Σύνδεση ως supplierorderreceiver**)** μπορεί:

- Να εισέλθει στο σύστημα και να επιβεβαιώσει την λήψη μιας παραγγελίας προϊόντων από κάποιον προμηθευτή που συνεργάζεται η επιχείρηση επιλέγοντας το "Ναι" από τις δύο Ενέργειες



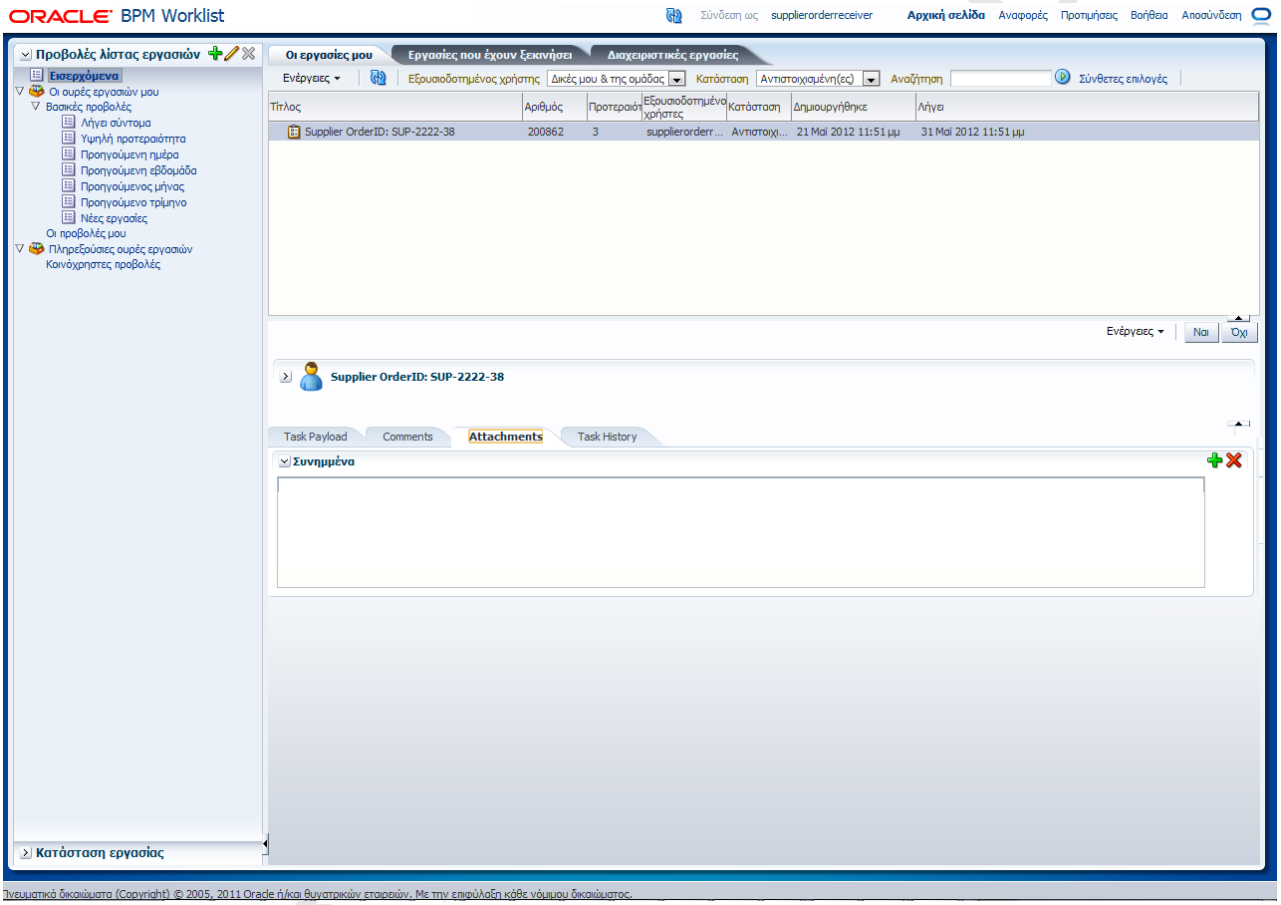
Σχήμα 88: Γραφικό περιβάλλον επιβεβαίωσης παραλαβής προϊόντων από τους προμηθευτές(1/3)

- Να προσθέσει κάποιο σχόλιο , όπως φαίνεται σχήμα 91



Σχήμα 89: Γραφικό περιβάλλον επιβεβαίωσης παραλαβής προϊόντων από τους προμηθευτές (2/3)

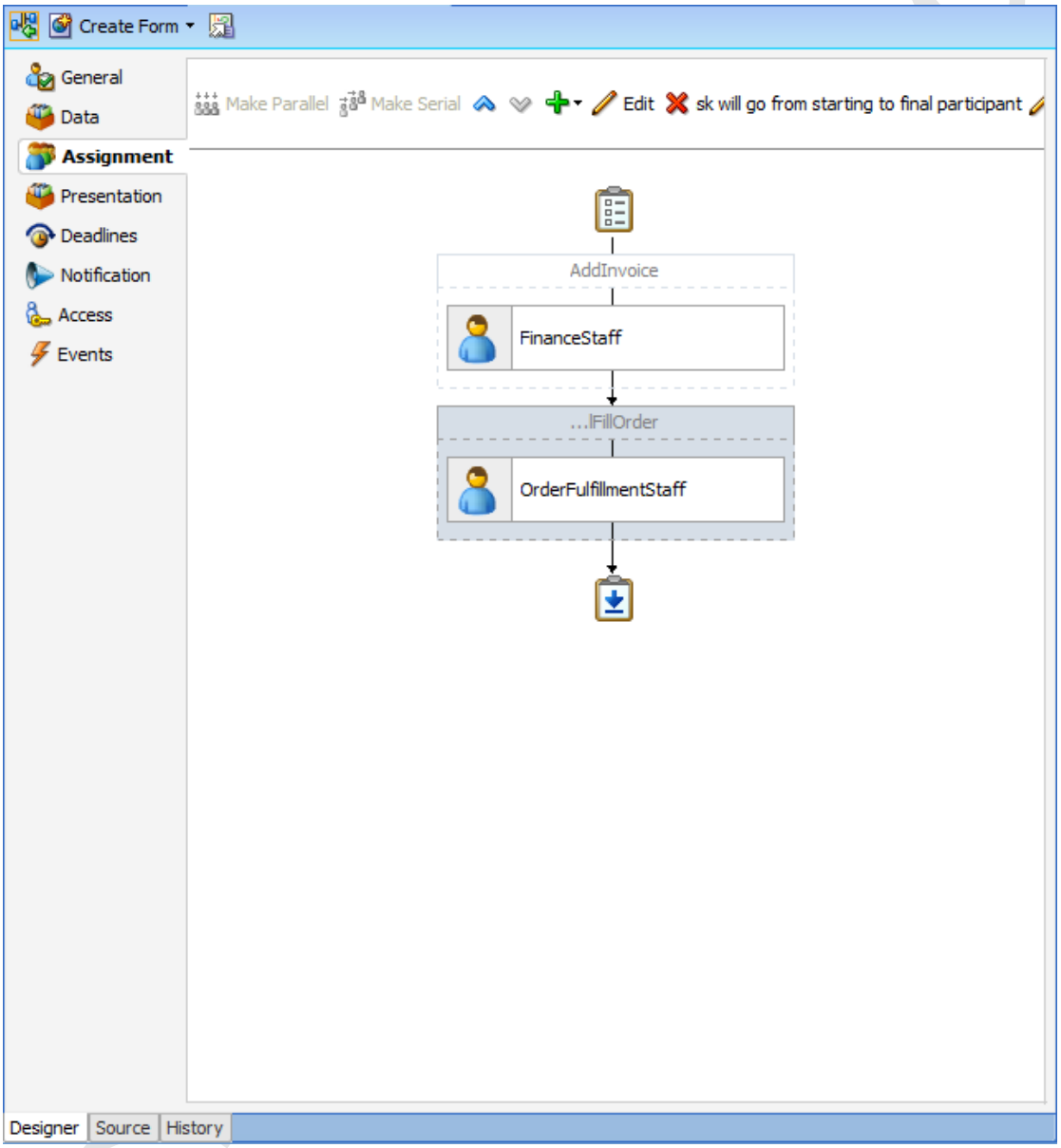
- Και αν επιθυμεί να επισυνάψει κάποιο έγγραφο σε ηλεκτρονική μορφή , όπως φαίνεται σχήμα 92



Σχήμα 90 Γραφικό περιβάλλον επιβεβαίωσης παραλαβής προϊόντων από τους προμηθευτές(3/3)

4.3.4.1.3 Τιμολόγηση και διεκπεραίωση παραγγελίας.

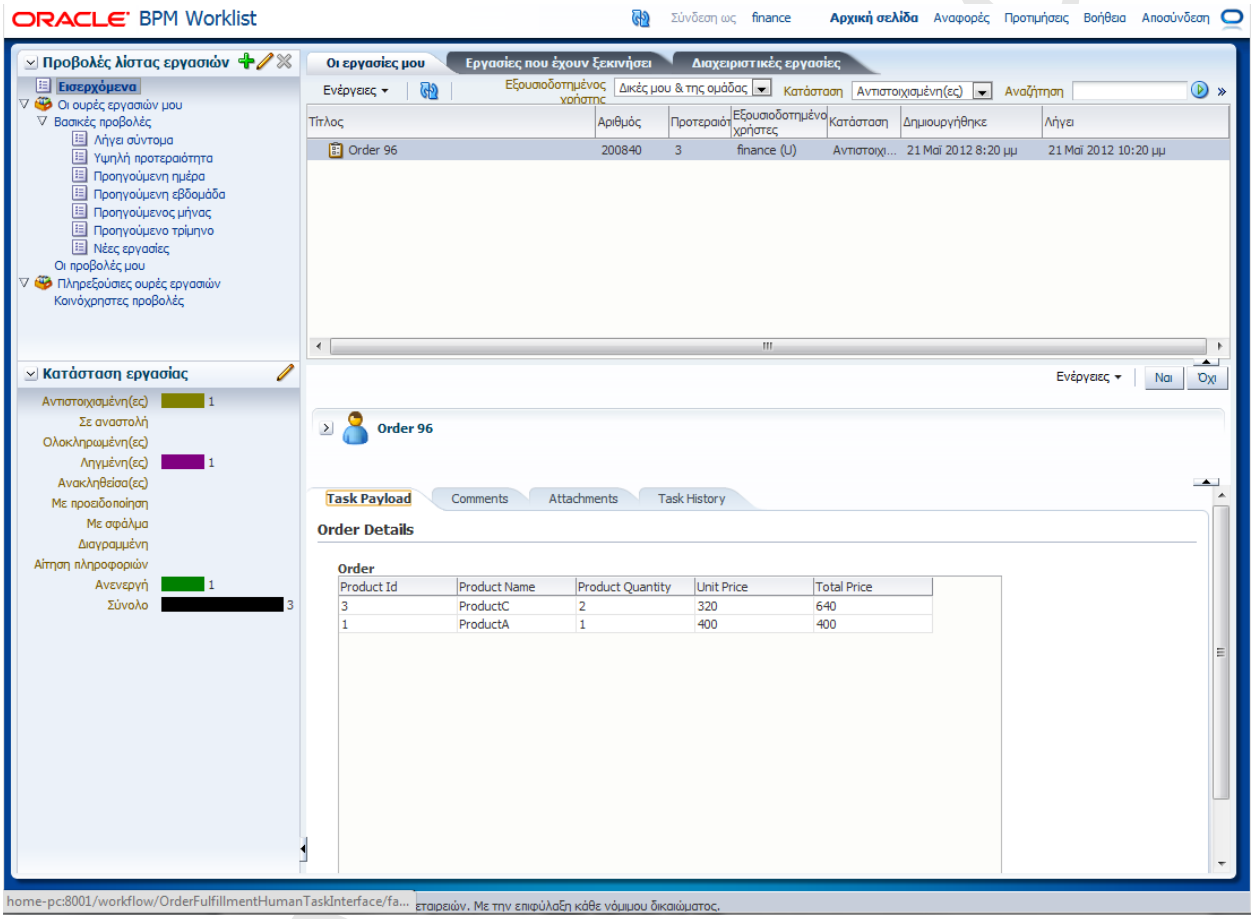
Το human task του σχήματος 93 αποτελείται από δύο ξεχωριστές δραστηριότητες η μια εκτελείτε από κάποιον υπάλληλο του λογιστηρίου (FinanceStaff) και η άλλη από κάποιο υπάλληλο του Τμήματος διεκπεραίωσης παραγγελιών (OrderFulfillmentStaff).



Σχήμα 91: Human Tasks Λογιστηρίου και Τμήματος διεκπεραίωσης παραγγελιών

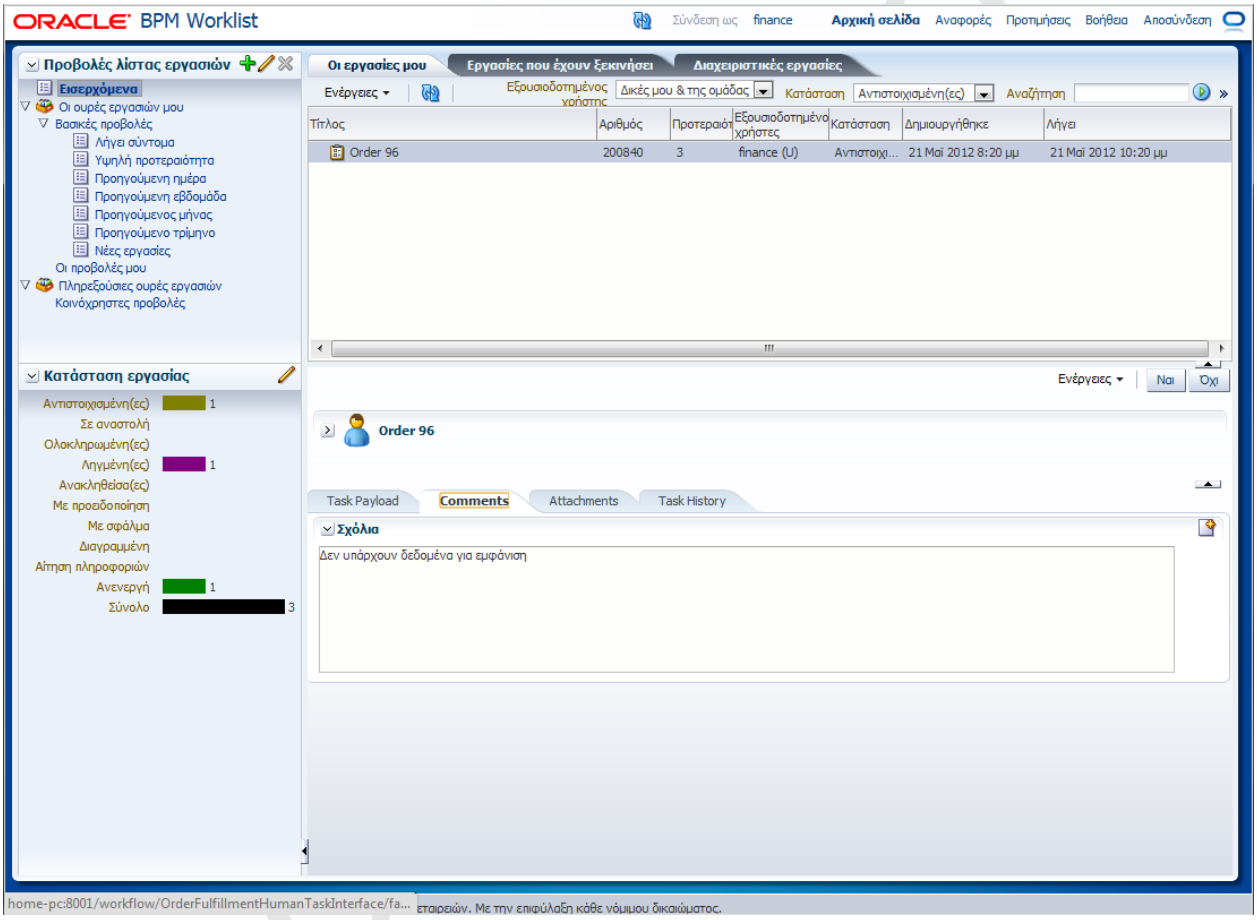
Ο υπάλληλος του λογιστηρίου (σχήμα 94, Σύνδεση ως finance) εισέρχεται στο σύστημα και μπορεί:

- Να ελέγξει τα προϊόντα μιας παραγγελίας και την σωστή τιμολόγηση τους



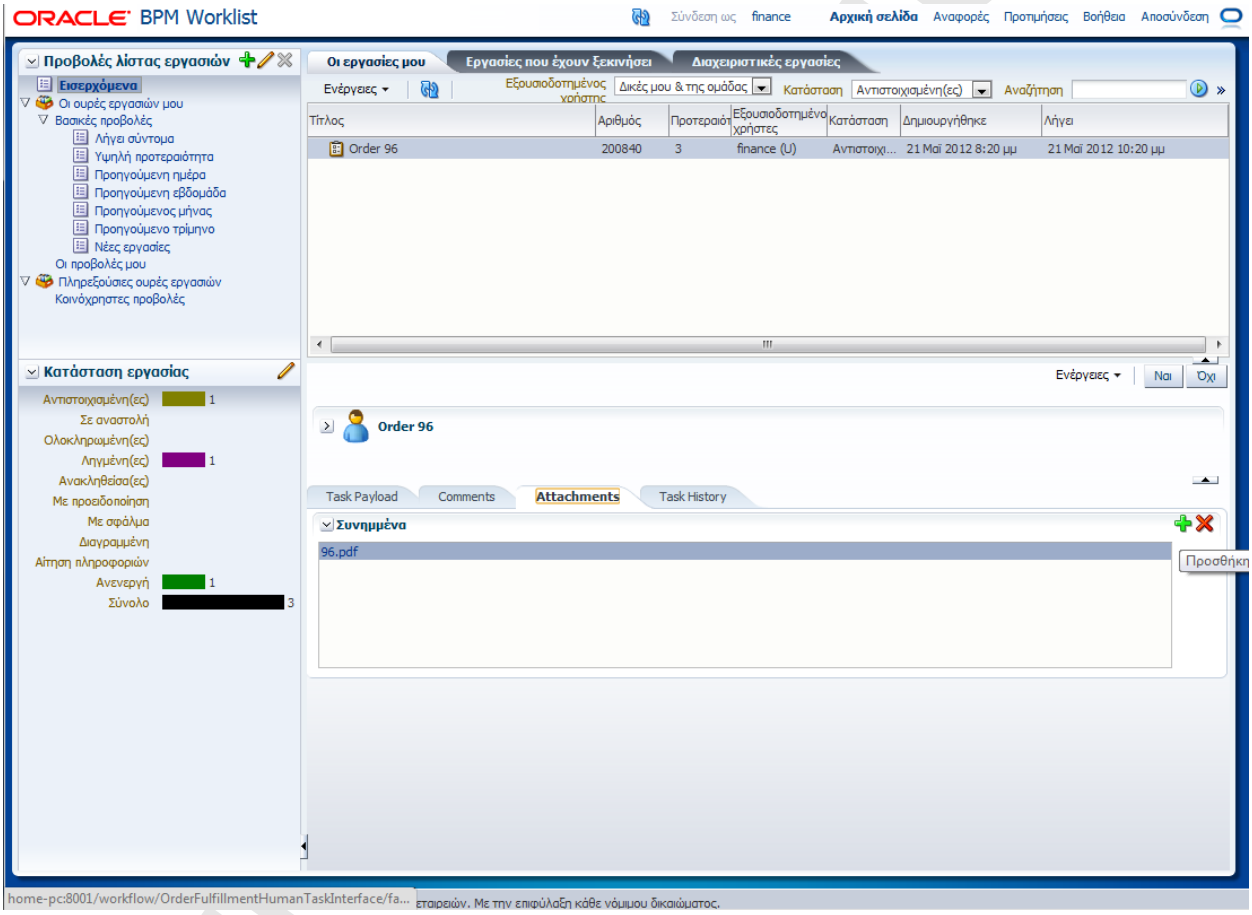
Σχήμα 92: Γραφικό περιβάλλον του ελέγχου και της επισύναψης της τιμολόγησης (1/3)

- Να προσθέσει κάποιο σχόλιο, όπως φαίνεται σχήμα 95

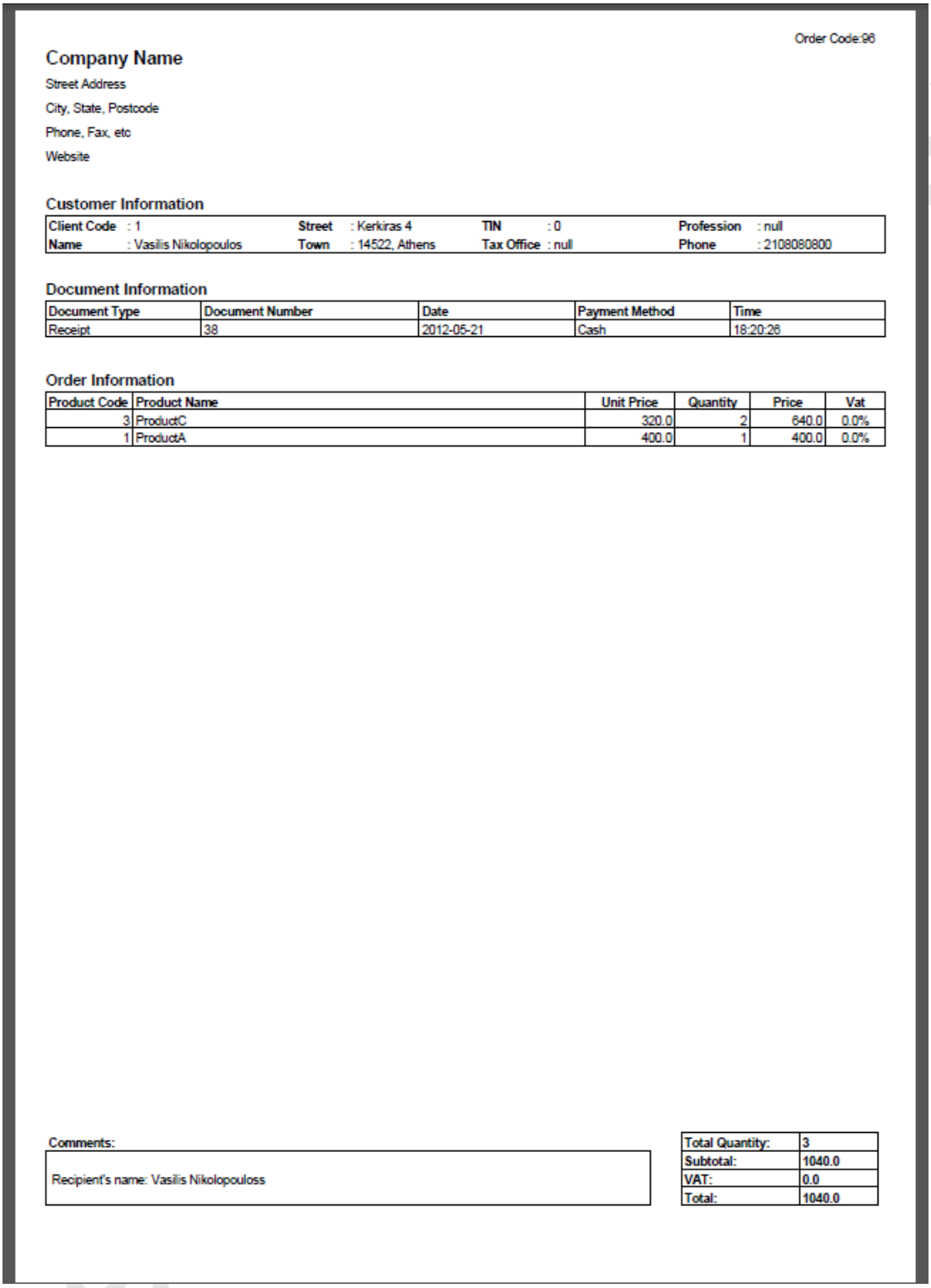


Σχήμα 93: Γραφικό περιβάλλον του ελέγχου και της επισύναψης της τιμολόγησης (2/3)

- Να επισυνάψει το παραστατικό της παραγγελίας, στην περίπτωση αυτή (σχήμα 96) επισύναψε το παραστατικό 96.pdf (σχήμα 97). Το παραστατικό αυτό δημιουργήθηκε αυτόματα σε ηλεκτρονική μορφή με την κλήση της μεθόδου produceReceiptPDF ενός Web service από την διαδικασία BPEL.
- Και τέλος επιλέγοντας το "Ναι" στις Ενέργειες να διεκπεραιώσει το δικό του κομμάτι στην διαδικασία BPEL4People και να στείλει την παραγγελία στο επόμενο στάδιο της διαδικασίας που είναι αυτό της διεκπεραίωσης της παραγγελίας.



Σχήμα 94: Γραφικό περιβάλλον του ελέγχου και της επισύναψης της τιμολόγησης (3/3)



Order Code:96

Company Name

Street Address

City, State, Postcode

Phone, Fax, etc

Website

Customer Information

Client Code	: 1	**Street**	: Kerkiras 4	**TIN**	: 0	**Profession**	: null
Name	: Vasilis Nikolopoulos	**Town**	: 14522, Athens	**Tax Office**	: null	**Phone**	: 2108080800

Document Information

Document Type	Document Number	Date	Payment Method	Time
Receipt	38	2012-05-21	Cash	18:20:26

Order Information

Product Code	Product Name	Unit Price	Quantity	Price	Vat
3	ProductC	320.0	2	640.0	0.0%
1	ProductA	400.0	1	400.0	0.0%

Comments:

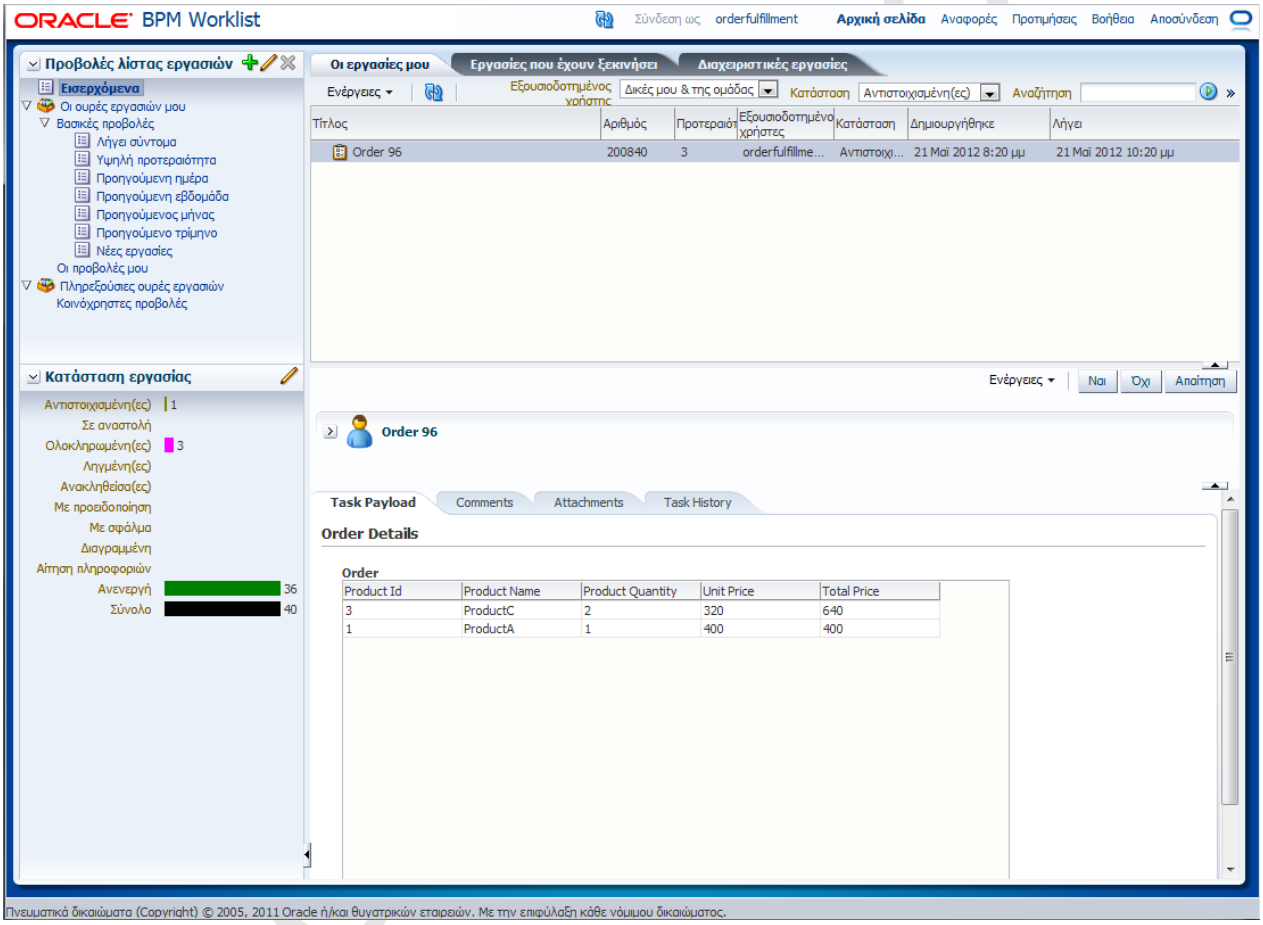
Recipient's name: Vasilis Nikolopouloss

Total Quantity:	3
Subtotal:	1040.0
VAT:	0.0
Total:	1040.0

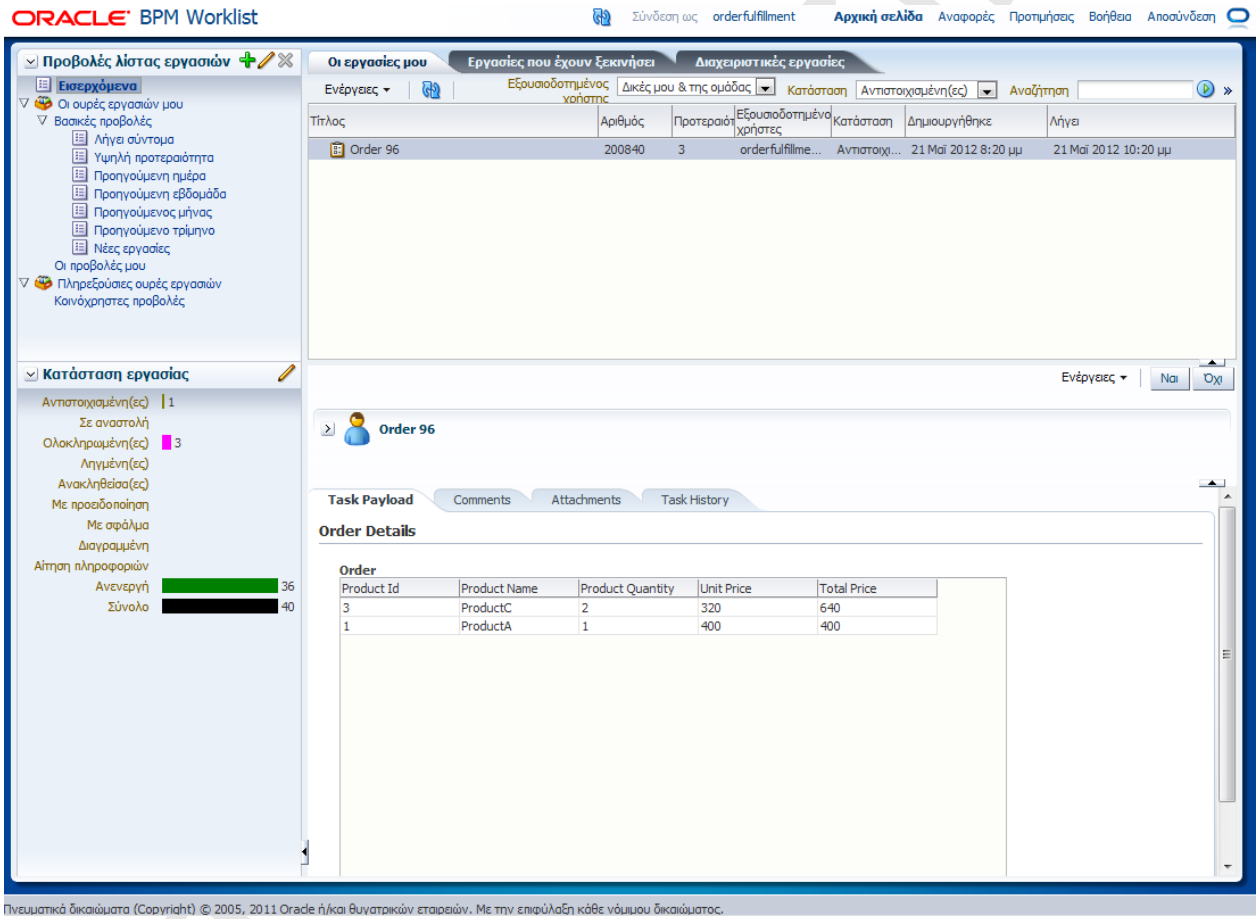
Σχήμα 95: Παράδειγμα παραστατικού Order 96.pdf

Ο υπάλληλος του τμήματος διεκπεραίωσης (σχήμα 98, Σύνδεση orderfulfillment) παραγγελιών μπορεί:

- Να δει και να δεσμεύσει μία παραγγελία επιλέγοντας το "Απαίτηση" στις ενέργειες έτσι ώστε κανένας άλλος υπάλληλος να μην μπορεί να την εκτελέσει, και αυτό προς αποφυγή εκτέλεσης μιας παραγγελίας δύο ή περισσότερες φορές από άλλον/ους υπάλληλο/ους. Αφού η παραγγελία δεσμευτεί από κάποιον υπάλληλο η επιλογή "Απαίτηση" δεν αποτελεί πλέον επιλέξιμη επιλογή (σχήμα 99) .

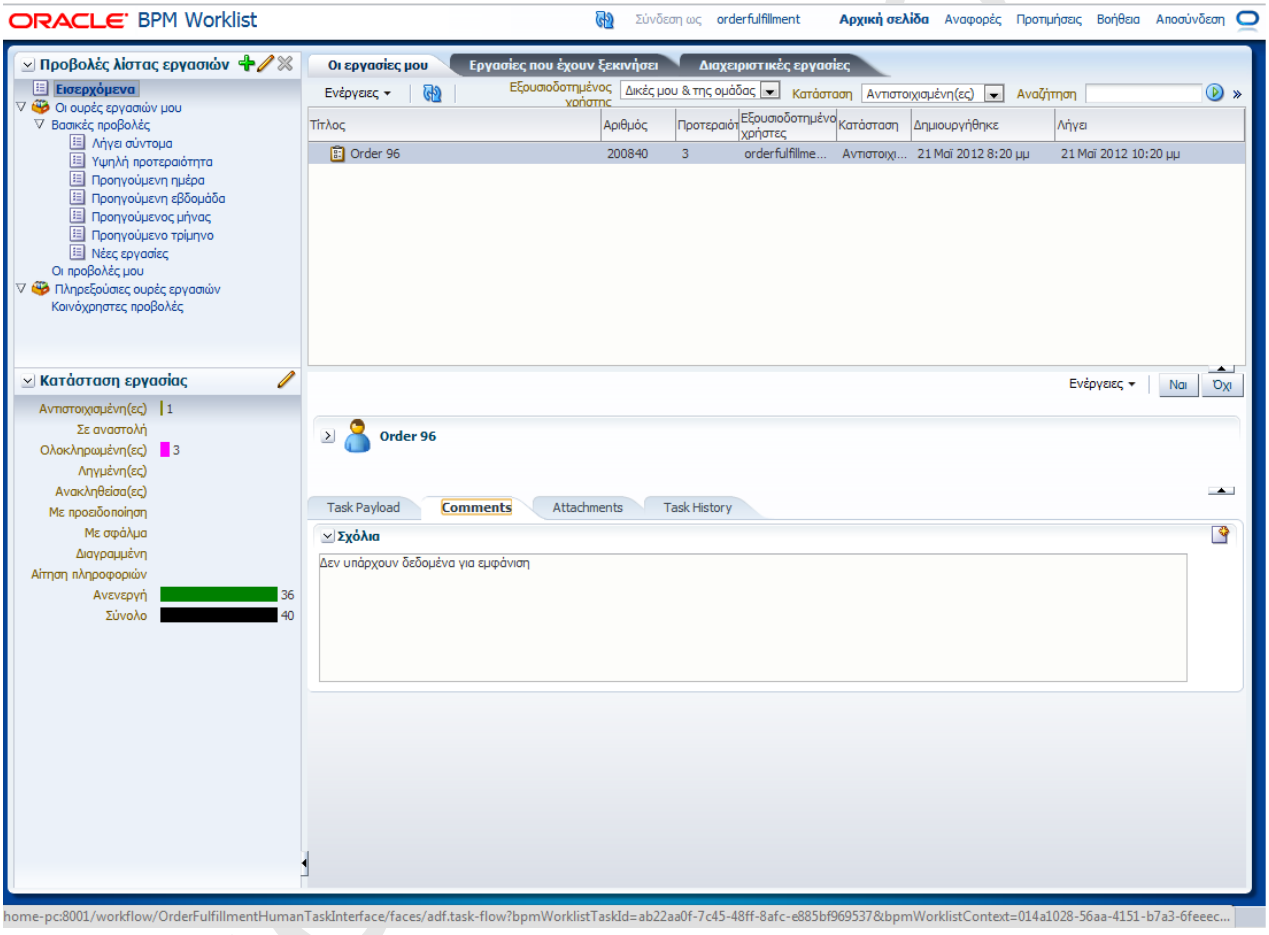


Σχήμα 96: Γραφικό περιβάλλον διεκπεραίωσης παραγγελίας (1/4)



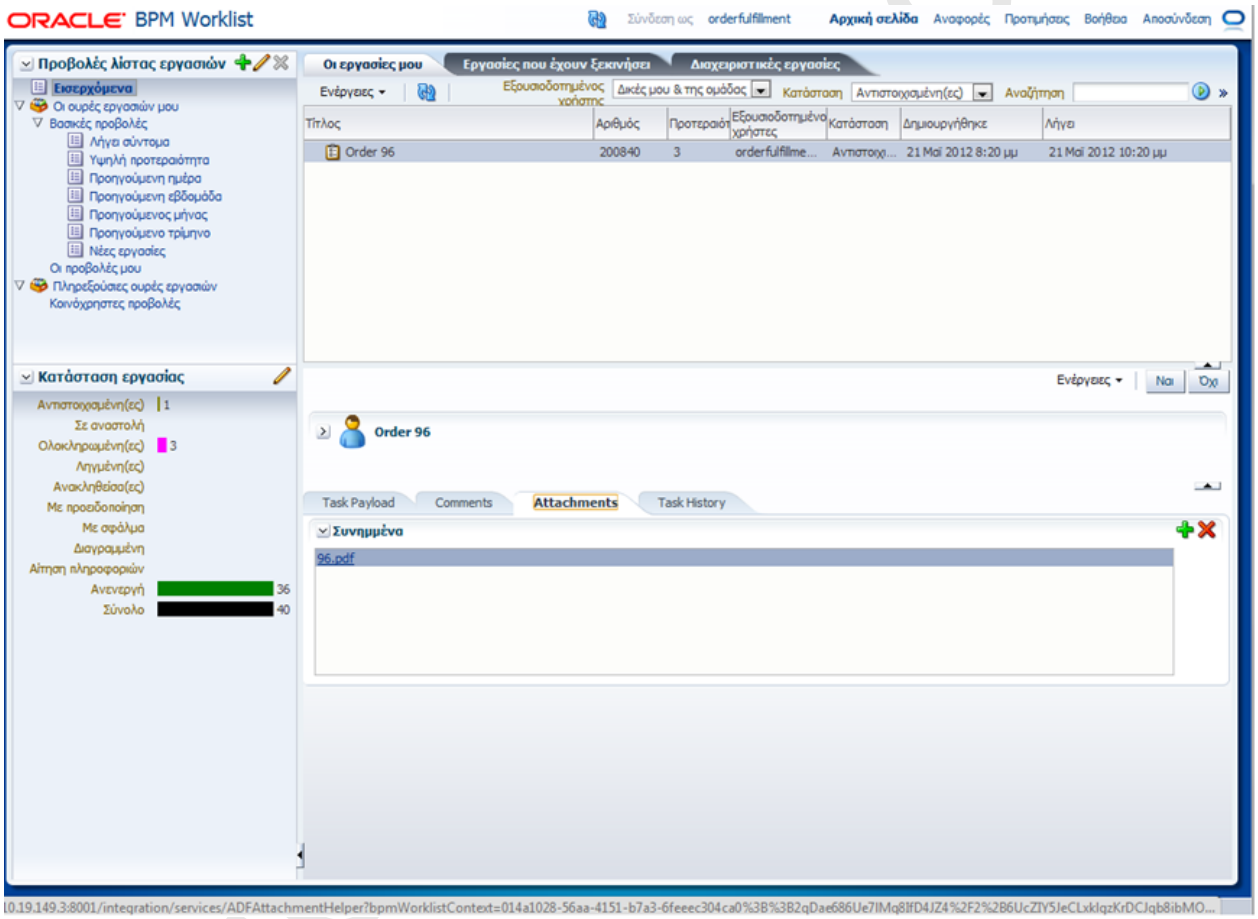
Σχήμα 97: Γραφικό περιβάλλον διεκπεραίωσης παραγγελίας (2/4)

- Να διαβάσει ή να προσθέσει κάποιο σχόλιο, όπως φαίνεται στο σχήμα 100

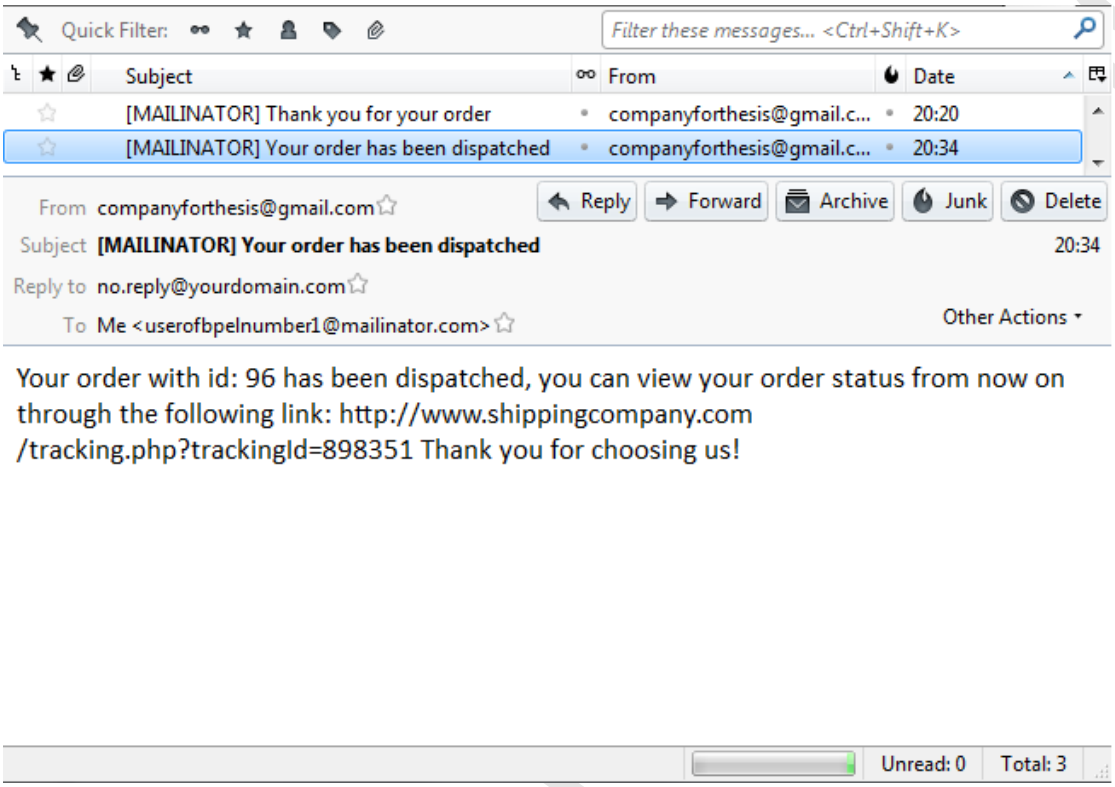


Σχήμα 98: : Γραφικό περιβάλλον διεκπεραίωσης παραγγελίας (3/4)

- Κατεβάσει και να εκτυπώσει το παραστατικό, όπως φαίνεται στο σχήμα 101.
- Και τέλος επιλέγοντας το "Ναι" στις Ενέργειες να διεκπεραιώσει το δικό του κομμάτι στην διαδικασία BPEL4People και να στείλει την παραγγελία στο επόμενο στάδιο της διαδικασίας που είναι αυτό της επικοινωνία μέσω του Web service της μεταφορικής εταιρίας, με την μεταφορική εταιρία για την παραλαβή και παράδοση της παραγγελίας στον πελάτη και αποστέλλεται στον πελάτη σχετικό ενημερωτικό e-mail σχήμα 102.



Σχήμα 99: : Γραφικό περιβάλλον διεκπεραίωσης παραγγελίας (4/4)



Σχήμα 100: Ενημερωτικό email αποστολής της παραγγελίας.

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

5 Συμπεράσματα

Σε αυτή την διπλωματική εργασία εξετάστηκαν όλες οι πτυχές της Service Oriented Architecture. Η SOA είναι αυτό που επιτρέπει και κάνει λειτουργική την επιχειρηματική ολοκλήρωση, είναι αυτή που παρέχει μια κοινή πλατφόρμα και ένα περιβάλλον εκτέλεσης για ετερογενής εφαρμογές που αναπτυχθεί με διαφορετικές τεχνολογίες. Η SOA επιτρέπει την διαλειτουργικότητα μεταξύ των διαφορετικών εφαρμογών αποκρύπτοντας την εσωτερική δομή τους δημιουργώντας έτσι ευέλικτα και χαλαρά διασυνδεδεμένα κατανεμημένα συστήματα.

Η SOA αρχιτεκτονική είναι ανεξάρτητη τεχνολογιών, αλλά οι τεχνολογίες των Web services και της BPEL αποτελούν τεχνολογίες με τις οποίες υλοποιείται κατά τον καλύτερο δυνατό τρόπο. Στην διπλωματική αυτή υλοποιήθηκε μια εφαρμογή η οποία καταδεικνύει τα θεμελιώδη χαρακτηριστικά της SOA. Αποδεικνύεται η διαλειτουργικότητα των διαφορετικών πλατφόρμων, η επαναχρησιμοποίηση των χαλαρά συνδεδεμένων υπηρεσιών, και η κατανάλωση των υπηρεσιών από άλλες ενότητες του λογισμικού για να σχηματίσουν μια υπηρεσία που βασίζεται στην αλληλεπίδραση μεταξύ διαφορετικών εφαρμογών. Επίσης μέσω της BPEL4People και των Human Task καταδεικνύεται η συμβατότητα της BPEL με την φιλοσοφία του BPM και η ολοκλήρωση/αλληλεπίδραση της με εφαρμογές που σχετίζονται με αυτό.

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

Βιβλιογραφία

Active Endpoints, Adobe, BEA, IBM, Oracle, SAP. (2007, June). *WS-BPEL Extension for People (BPEL4People), Version 1.0.* Ανάκτηση Ιανουάριος 7, 2012, από http://public.dhe.ibm.com/software/dw/specs/ws-bpel4people/BPEL4People_v1.pdf

Arsanjani, A. (2005). *How to Identify, Specify and Realize Services for your SOA.* Ανάκτηση Οκτώβριος 12, 2011, από http://www.ebizq.net/topics/dev_tools/features/5632.html

Barros, A., Dumas, M., & Oaks, P. (2006). Standards for Web Service Choreography and Orchestration: Status and Perspectives. Στο C. J. Bussler, & A. Haller, *LECTURE NOTES IN COMPUTER SCIENCE* (Τόμ. 3182, σσ. 61-74). SpringerLink.

BEA, IBM, Microsoft, SAP, & Siebel. (2003, May 5). Business Process Execution Language for Web Services.

Behara, G. K. (2006, May). *BPM and SOA: A Strategic Alliance.* Ανάκτηση Ιανουάριος 24, 2012, από http://www.bptrends.com/publicationfiles/05-06-wp-bpm-soa-behara.pdf

Bellwood, T. (2002, July 01). *Understanding UDDI, Tracking the evolving specification*. (IBM) Ανάκτηση Μάϊος 4, 2011, από http://www.ibm.com/developerworks/webservices/library/ws-featuddi/

Brittenham, P., Cubera, F., Ehnebuske, D., & Graham, S. (2001, September 1). (IBM) Ανάκτηση Μάϊος 4, 2011, από Understanding WSDL in a UDDI registry, Part 1 - How to publish and find WSDL service descriptions: http://www.ibm.com/developerworks/webservices/library/ws-wsdl/

Carter, S. (2007). *The New Language of Business SOA & Web 2.0.* IBM press.

Cerami, E. (2002). *Web Services Essentials* (First εκδ.). O'Reilly.

Channabasavaiah, K., Holley, K., & Tuggle , E. M. (2004). *Migrating to a service oriented architecture.* IBM.

Davenport, T. H. (1993). *Process Innovation: Reengineering Work through Information Technology.* Cambridge: Harvard Business Press.

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

Erl, T. (2005). *Service Oriented Architecture – Concepts, Technology.* Prentice Hall PTR.

Erl, T. (n.d). *SOA principles*. Ανάκτηση Οκτώβριος 11, 2011, από http://www.soaprinciples.com/p19.php

Erl, T., Bennett, S. G., Carlyle, B., Gee, C., Manes, A. T., Moores, R., και συν. (2011). *SOA Governance: Governing Shared Services On-Premise & in the Cloud* (1st Edition εκδ.). Prentice Hall.

Greek Interoperability Centre. (2010, March 10). *Διάλεξη 12: Μοντέλο Τεκμηρίωσης Υπηρεσιών, Εγγράφων και Διαδικτυακών Υπηρεσιών.* Ανάκτηση Ιανουάριος 29, 2012, από http://www.iocenter.eu/media/12300/pres12%20[compatibility%20mode].pdf

Πανεπιστήμιο Πειραιώς

Gunathilake, K. W. (2011, Aprill 16). *Kasun's Blog*. Ανάκτηση February 8, 2012, από http://kasunweranga.blogspot.com/2011/04/wsdl-to-uddi-mapping.html

Hammer, M., & Champy, J. (1993). *Reengineering the Corporation: A Manifesto for Business Revolution .* New York: HarperColling.

Harrington, J. H. (1991). *Business Process Improvement.* New York: McGraw-Hill.

Hill Janelle B., S. J., Flint, D., & Melenovsky, M. J. (2006). *Gartner's Position on Business Process Management, 2006.* Gartner.

IBM. (2000, September 06). *Web Services architecture overview*. Ανάκτηση Ιανουάριος 03, 2011, από http://www.ibm.com/developerworks/webservices/library/w-ovr/

IBM. (n.d). *New to SOA and web services*. Ανάκτηση Νοέμβριος 2, 2011, από http://www.ibm.com/developerworks/webservices/newto/service.html

IBM. (n.d). *Web services: Key roles* . Ανάκτηση Μάϊος 08, 2011, από http://publib.boulder.ibm.com/infocenter/tpfhelp/current/index.jsp?topic=%2Fcom.ibm.ztpf-ztpfdf.doc_put.cur%2Fgtps6%2Fs6wsrol.html

IBM, Microsoft, & BEA. (2002, July 31). *Business Process Execution Language for Web Services, Version 1.0.* Ανάκτηση Δεκέμβριος 22, 2011, από http://public.dhe.ibm.com/software/dw/specs/ws-bpel/ws-bpel1.pdf

Jewell, T., & Chappell, D. (2002). *Java Web Services.* O'REILLY.

Josuttis, N. M. (2007). *SOA in Practice* (First εκδ.). O'Reilly.

Juric, M. B., Chandrasekaran, S., Frece, A., Hertis, M., & Gregor, S. (2010). *WS-BPEL 2.0 for SOA Composite Applications with IBM WebSphere 7* (First εκδ.). Packt Publishing Ltd.

Juric, M. B., Mathew, B., & Sarang, P. (2006). *Business Process Execution Language for Web Services* (2nd Edition εκδ.). Packt Publishing .

Juric, M. B. (n.d). *A Hands-on Introduction to BPEL*. Ανάκτηση Δεκέμβριος 26, 2011, από http://www.oracle.com/technetwork/articles/matjaz-bpel1-090575.html

Ko, R. K., Lee, S. S., & Lee, E. W. (2009). Business process management (BPM) standards: a survey. *Business Process Management Journal, 15*(5), 744-791.

Kreger, H. (2001, May). Ανάκτηση Μάϊος 17, 2011, από Web Services Conceptual Architecture (WSCA 1.0): http://www.csd.uoc.gr/~hy565/newpage/docs/pdfs/papers/wsca.pdf

Latha, S., & Treadwell, J. (2005). *An Overview of Service-oriented Architecture, Web Services and Grid Computing.* HP Software Global Business Unit.

Linthicum, D. (2007, October 15). *SOA - Loosely Coupled...What?* Ανάκτηση Δεκέμβριος 10, 2011, από SOA WORLD MAGAZINE : http://soa.sys-con.com/node/439723

Mahmoud, Q. H. (2005, April). (Oracle) Ανάκτηση Οκτώβριος 21, 2011, από Service-Oriented Architecture (SOA) and Web Services: The Road to Enterprise Application Integration (EAI): http://www.oracle.com/technetwork/articles/javase/soa-142870.html

McGovern, J., Tyagi, S., Stevens E, M., & Mathew, S. (2003). *Java Web Services Architecture .* Morgan Kaufmann.

Microsoft. (n.d). Ανάκτηση Μάϊος 7, 2011, από UDDI Business Registry Shutdown FAQ: http://uddi.microsoft.com/about/FAQshutdown.htm

Πανεπιστήμιο Πειραιώς

Mitra, T. (2008, January). *Architecture in practice, Part 6: Why business process management (BPM) is important to an enterprise.*

Newcomer, E., & Lomow, G. (2004). *Understanding SOA with Web Services.* Addison Wesley Professional.

Numnonda, T. (2009, June 26). *Topic 3 Business Process Management using BPEL.* Ανάκτηση Ιανουάριος 30, 2012, από http://www.slideshare.net/thananum/business-process-management-using-bpel

O'Brien, R. (1998). *An Overview of the Methodological Approach of Action Research*. Ανάκτηση 3 12, 2012, από http://www.web.ca/robrien/papers/arfinal.html

OASIS. (2004). *UDDI Version 3.0.2, UDDI Spec Technical Committee Draft, Dated 20041019*. Ανάκτηση Μάϊος 4, 2011, από http://uddi.org/pubs/uddi_v3.htm

OASIS. (2006, February 10). *Reference Model for Service Oriented Architecture 1.0.* Ανάκτηση Οκτώβριος 12, 2011, από http://www.oasis-open.org/committees/download.php/16630/wd-soa-rm-pr1.doc

OASIS. (2007, April 11). *Web Services Business Process Execution Language Version 2.0.* Ανάκτηση Δεκέμβριος 22, 2011, από http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf

OASIS. (2010, August 17). *WS-BPEL Extension for People (BPEL4People), Version 1.0.* Ανάκτηση Ιαναούάριος 6, 2012, από http://docs.oasis-open.org/bpel4people/bpel4people-1.1.html

OMG. (2011, January). *Business Process Model and Notation Version 2.0.* Ανάκτηση Ιανουάριος 27, 2012, από http://www.omg.org/spec/BPMN/2.0/PDF

Oracle. (2005). *Instituto de Informatica.* Ανάκτηση Ιανουάριος 27, 2011, από http://www.inst-informatica.pt/servicos/informacao-e-documentacao/biblioteca-digital/arquitectura-e-desenvolvimento-de-aplicacoes/soa/strategies-for-soa-success.pdf

Papazoglou, M. (2008). *Web Services: Principles and Technology* (1st edition εκδ.). Prentice Hall.

Peltz, C. (2003, January). *Web services orchestration: A review of emerging technologies, tools, and standards.* Ανάκτηση Ιανουάριος 18, 2011, από http://itee.uq.edu.au/~infs3204/interesting_websites/WSOrchestration.pdf

Poduval, A., Todd, D., Gaur, H., Bolie, J., Geminiuc, K., Pravin, L., και συν. (2011). *Do More with SOA Integration: Best of Packt.* Pact Publishing.

Potts, S., & Kopack, M. (2003). *Sams Teach Yourself Web Services in 24 Hours.* Indianapolis, Indiana, USA: Sams Publishing.

Reason, p., & Bradbury, H. (2001). *Handbook of action research: participative inquiry and practice.* London Sage.

Rosenberg, J., & Remy, D. L. (2004). *Securing Web Services with WS-Security .* Sams Publishing.

SAP. (n.d). Ανάκτηση Οκτώβριο 19, 2011, από SAP - Service-Oriented Architecture: Business Benefits: http://www.sap.com/platform/soa/businessbenefits/index.epx

SAP, IBM. (2005, July). *WS-BPEL Extension for People – BPEL4People.* Ανάκτηση Ιανουάριος 7, 2012, από

Πανεπιστήμιο Πειραιώς

http://public.dhe.ibm.com/software/dw/specs/ws-bpel4people/BPEL4People_white_paper.pdf

Sotomayor, B. (2005). *The Globus Toolkit 4 Programmer's Tutorial.* Ανάκτηση Μάϊος 17, 2011, από http://gdp.globus.org/gt4-tutorial/multiplehtml/index.html

SUN. (n.d). *Service-Oriented Architecture – Overview*. Ανάκτηση Οκτώβριος 12, 2011, από http://cs.sun.com/practice/software/soa/index.jsp

Swenson, K. D. (2005). Workflow and web service standards. *Business Process Management Journal , 11*(3), 218-223.

The Open Group. (2009). *SOA Governance Framework.* The Open Group.

The Open Group. (n.d). Ανάκτηση Οκτώβριο 20, 2011, από The SOA Source Book — Service Oriented Architecture : SOA Features and Benefits: http://www.opengroup.org/soa/source-book/soa/soa_features.htm

The Open Group. (n.d). *SOA Work Group - Definition of SOA*. Ανάκτηση Νοεμβριος 12, 2011, από http://www.opengroup.org/projects/soa/page.tpl?CALLER=newpage.tpl&ggid=1575

The Stencil Group. (2002, July 19). *The Evolution of UDDI UDDI.org White Paper.* Ανάκτηση Μάϊος 4, 2011, από http://www.uddi.org/pubs/the_evolution_of_uddi_20020719.pdf

Vasiliev, Y. (2007). *SOA and WS-BPEL: Composing Service-Oriented Architecture Solutions with PHP and Open-Source ActiveBPEL.* Packt Publishing.

Vasudevan, V. (2001, April 04). *A Web Services Primer.* Ανάκτηση Μάϊος 2, 2011, από http://www.xml.com/pub/a/ws/2001/04/04/webservices/index.html

W3C. (2001, March 15). *Web Services Description Language (WSDL) 1.1* . Ανάκτηση Νοέμβριος 4, 2011, από http://www.w3.org/TR/wsdl

W3C. (2004). *Spinning Threads for the Next Generation of the Web - slide "SOAP Structure"*. Ανάκτηση Νοέμβριος 04, 2011, από http://www.w3.org/2004/Talks/0611-sb-wsswintro/slide9-0.html

W3C. (2004, February 11). *Web Services Architecture*. Ανάκτηση Μάϊος 23, 2011, από http://www.w3.org/TR/ws-arch/

W3C. (2004, February 11). *Web Services Glossary* . Ανάκτηση Δεκέμβιος 17, 2011, από http://www.w3.org/TR/ws-gloss/

W3C. (2007, April 27). *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)*. Ανάκτηση ΜάΪος 22, 2011, από http://www.w3.org/TR/soap12-part1/

W3C. (2007, April 27). *SOAP Version 1.2 Part 2: Adjuncts (Second Edition).* Ανάκτηση Μάϊος 22, 2011, από http://www.w3.org/TR/2007/REC-soap12-part2-20070427/

W3C. (2007, June 26). *Web Services Description Language (WSDL) Version 2.0 Part 2: Adjuncts*. Ανάκτηση Νοέμβριος 4, 2011, από http://www.w3.org/TR/wsdl20-adjuncts/

Weerawarana, S., & Curbera, F. (2002, August 01). *Business Process with BPEL4WS: Understanding BPEL4WS, Part 1.* Ανάκτηση Ιανουάριος 14, 2011, από http://www.ibm.com/developerworks/webservices/library/ws-bpelcol1/index.html

Πανεπιστήμιο Πειραιώς

Weerawarana, S., Curbera, F., Leymann, F., Storey, T., & Ferguson, D. (2005). *Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More* (1st Edition εκδ.). Prentice Hall.

Weske, M. (2007). *Business Process Management Concepts, Languages, Architectures.* Potsdam, Germany: Springer.

Wikipedia. (2011, August 22). *Web Service Choreography*. Ανάκτηση Ιανουάριος 5, 2012, από http://en.wikipedia.org/wiki/Web_Service_Choreography#cite_note-5

Wikipedia. (2011, November 17). *Web Services Description Language*. Ανάκτηση Οκτώβριος 2, 2011, από http://en.wikipedia.org/wiki/Web_Services_Description_Language

Θεμιστοκλέους, Μ., & Μαντζάνα, Β. (2010). *Υπηρεσίες Παγκόσμιου Ιστού και Υπηρεσιοστρεφείς Αρχιτεκτονικές.* αυτό-έκδοση.

Μπάρδης, Γ. Ν. (2009). Πολυεπίπεδη μοντελοποίηση παραγωγής για το βέλτιστο συγχρονισμό επιχειρηματικών διαδικασιών, τεχνολογογικής υποδομής και ανθρώπινου δυναμικού. *Διδακτορική Διατριβή*. Αθήνα: Ε.Μ.Π.

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

Ακρώνυμα

A	
API	Application Programming Interface
B	
B2B	Business to Business
BAM	Business Activity Monitoring
BPEL	Business Process Execution Language
BPEL4WS	Business Process Execution Language for Web Services
BPEL4People	Business Process Execution Language for People
BPMN	Business Process Modeling Notation
C	
CORBA	Common Object Request Broker Architecture
CRM	Customer relationship management
D	
DCOM	Distributed Component Object Model
E	
EAI	Enterprise Application Integration
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
H	
HTTP	Hypertext Transfer Protocol
I	
IT	Information Technology
K	
KPI	Key Performance Indicators
M	
MEP	Message Exchange Pattern
MIME	Multipurpose Internet Mail Extensions
MOM	Messaging Oriented Middleware
N	
NASSL	Network Application Service Specification Language
O	
OASIS	Organization for the Advancement of Structured Information Standards
Q	
QoS	Quality of Service

Πανεπιστήμιο Πειραιώς

R

RMI	Remote Method Invocation
ROI	Return Of Investment
RPC	Remote Procedure Call

S

SDL	Service Description Language
SLA	Service Level Agreement
SOA	Service Oriented Architecture
SSL	Secure Sockets Layer

U

UDDI	Universal Description Discovery and Integration

W

W3C	World Wide Web Consortium
WS	Web Service
WS-CDL	Web Services Choreography Description Language
WSBPEL	Web Services Business Process Execution Language
WSDL	Web Service Description Language
WSFL	Web Service Flow Language
WSRP	Web Services for Remote Portlet

X

XML	Extensible markup language

Ελληνικά

ΕΑ	Επιχειρησιακή Αρχιτεκτονική

Πανεπιστήμιο Πειραιώς

Παράρτημα Α: Δραστηριότητες BPEL

<receive>

Με την <receive> δραστηριότητα η επιχειρηματική διαδικασία περιμένει για ένα εισερχόμενο μήνυμα.

<reply>

Η <reply> δραστηριότητα χρησιμοποιείται για να σταλθεί μια απάντηση σε ένα αίτημα που έχει προηγουμένως αποδεχθεί μέσω μιας <receive> δραστηριότητας. Ο συνδυασμός της <receive> και της <reply> λειτουργίας διαμορφώνει μια λειτουργία αίτησης/απάντησης. Μια ασύγχρονη απάντηση αποστέλλεται πάντα με την κλήση της αντίστοιχης μονόδρομης λειτουργίας, σε ένα partnerLink.

<invoke>

Η <invoke> δραστηριότητα χρησιμοποιείται για την κλήση των λειτουργιών ενός web service που προσφέρεται από τους συνεργάτες. Μπορεί να είναι είτε μονόδρομη είτε αμφίδρομη η λειτουργία η οποία καλείται.

<assign>, <copy>, <from>, <to>:

Μια δραστηριότητα <assign> χρησιμοποιείται για:

- Την αντιγραφή των δεδομένων μιας μεταβλητής σε μια άλλη
- Την δημιουργία και την εισαγωγή νέων δεδομένων μέσω της χρήσης εκφράσεων “expressions” και λεκτικών τιμών.
- Την αντιγραφή των αναφορών των endpoint των partner link

<throw>

Η <throw> δραστηριότητα χρησιμοποιείται για την σηματοδότηση των εσωτερικών σφαλμάτων μιας επιχειρηματικής διαδικασίας.

<terminate>

Η <terminate> δραστηριότητα χρησιμοποιείται για τον άμεσο τερματισμό ενός στιγμιότυπου μιας επιχειρηματικής διαδικασίας

<wait>

Η <wait> δραστηριότητα επιτρέπει την αναμονή για ένα προκαθορισμένο χρονικό διάστημα ή μέχρι κάποια χρονική στιγμή.

Πανεπιστήμιο Πειραιώς

<empty>

Η <empty> είναι μια δραστηριότητα που δεν εκτελεί καμία λειτουργία. Χρησιμοποιείται για να δηλώσει κενά μπλοκ εντολών (π.χ ένα catch στο οποίο δεν είναι επιθυμητό να γίνει καμία ενέργεια).

<sequence>

Μια <sequence> δραστηριότητα χρησιμοποιείται για τον ορισμό των δραστηριοτήτων οι οποίες είναι αναγκαίο να εκτελεστούν με σειρά ακολουθίας.

<switch>

Για να εκφραστεί μια συμπεριφορά η οποία υπόκεινται σε όρους, χρησιμοποιείται η <switch> δραστηριότητα. Αποτελείται από ένα ή περισσότερα υπό συνθήκη κλαδιά τα οποία ορίζονται από στοιχεία <case>, ακολουθούμενα από ένα προαιρετικό <otherwise> στοιχείο.

<while>

Μια <while> δραστηριότητα χρησιμοποιείται για τον ορισμό μιας επαναληπτικής δραστηριότητας. Η επαναληπτική δραστηριότητα ασκείται μέχρι η Boolean συνθήκη που έχει οριστεί δεν κατέχει πλέον την τιμή true.

<pick>

Η <pick> δραστηριότητα χρησιμοποιείται για την αναμονή για την εμφάνιση ενός συμβάντος από ένα σύνολο συμβάντων και στη συνέχεια την εκτέλεση μιας δραστηριότητας που σχετίζεται με το συμβάν αυτό.

<flow>

Η <flow> δραστηριότητα παρέχει ταυτόχρονη εκτέλεση των εσώκλειστων δραστηριοτήτων και το συγχρονισμό τους.

<scope>

Τα scopes καθορίζουν τη περιβάλλοντα συμπεριφορών για τις δραστηριότητες. Παρέχουν fault handlers, event handlers, compensation handlers, μεταβλητές δεδομένων και correlation sets για τις δραστηριότητες.

<compensate>

Για να κληθεί ένας compensation handler, μπορεί να χρησιμοποιηθεί η δραστηριότητα <compensate>. Η δραστηριότητα <compensate> έχει ένα χαρακτηριστικό προαιρετικό scope που μπορεί να χρησιμοποιηθεί για τον καθορισμό του compensation handler που θα κληθεί. Το όνομα του scope ή το όνομα της

Πανεπιστήμιο Πειραιώς

δραστηριότητας (για τους ενσωματωμένους compensation handlers) πρέπει να διευκρινιστεί.

<catch>, <catchAll>

Οι <catch> δραστηριότητες ορίζονται εντός των χειριστών σφαλμάτων για να καθορίσουν τα σφάλματα τα οποία ανιχνεύονται και να γίνει ο χειρισμός τους. Η δραστηριότητα <catchAll> χρησιμοποιείται για να ανιχνευτούν όλα τα σφάλματα. Μέσα σε ένα δείκτη χειρισμού σφαλμάτων, πρέπει να καθοριστεί τουλάχιστον μία <catch> δραστηριότητα. Η προαιρετική <catchAll> δραστηριότητα μπορεί να καθοριστεί επίσης.

Η <catch> δραστηριότητα έχει δύο χαρακτηριστικά που μπορούν να χρησιμοποιηθούν για να καθοριστεί ποιο σφάλμα θα διαχειριστεί. Τουλάχιστον ένα από τα δύο πρέπει να καθοριστεί:

- faultName: καθορίζει το όνομα της βλάβης προς διαχείριση
- faultVariable: καθορίζει τον τύπο μεταβλητής που χρησιμοποιείται για δεδομένα σφάλματος

<partnerLink>, <partnerLinks>

Μια επιχειρηματική διαδικασία αλληλεπιδρά με υπηρεσίες που έχουν μοντελοποιηθεί σαν <partnerLink>. Κάθε <partnerLink> χαρακτηρίζεται από ένα <partnerLinkType>. Περισσότερα από ένα συνδέσεις <partnerLink> μπορεί να χαρακτηριστούν από το ίδιο <partnerLinkType>. Ανατρέξτε στην επόμενη ενότητα για περισσότερες σχετικά με <partnerLinkType>

<partnerLinkType>, <role>

Ένα <partnerLinkType> χαρακτηρίζει τη σχέση μεταξύ των δύο υπηρεσιών. Καθορίζει τους <role> για καθεμία από τις υπηρεσίες της συνομιλίας όταν αυτές συνομιλούν μεταξύ τους και καθορίζει το <portType> που παρέχεται από κάθε υπηρεσία για να λαμβάνει μηνύματα. Τα <partnerLinkType> και οι <role> καθορίζονται στο WSDL.

<correlations>, <correlation>

Ένα στοιχείο <correlation> χρησιμοποιείται για τον συσχετισμό ενός <correlationSet> με μια δραστηριότητα. Μπορεί να χρησιμοποιηθεί εντός των δραστηριοτήτων <receive>, <reply>, <invoke> και <onMessage>.

<correlationSets>, <correlationSet>

Η λειτουργία του είναι σημαντική και χρησιμοποιείται για τον συσχετισμό ενός μηνύματος με ένα στιγμιότυπο της BPEL διαδικασίας. Κάθε correlation set έχει ένα name χαρακτηριστικό.

Πανεπιστήμιο Πειραιώς

Πανεπιστήμιο Πειραιώς

Παράρτημα Β: Κώδικας Εργασίας

6.1 Κώδικας προσομοίωσης τραπεζικών Web services.

Κώδικας 1: AlphaBankGrWS.java

```
package com.alphabankgr.webservice;

import com.jdbcquery.code.JdbcConnect;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Timestamp;
import java.util.Date;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

/**
*
* @author Epameinondas Nikolopoulos
*/

@WebService(serviceName = "AlphaBankGrWS")
public class AlphaBankGrWS  {
JdbcConnect con = new JdbcConnect();

   @WebMethod(operationName = "InsertCardTransaction")
   public boolean ValidateCreditCard(
         @WebParam(name = "cardNumber") String cardNumber,
         @WebParam(name = "cardHoldersName") String cardHolderName,
         @WebParam(name = "cardExpirationMonth") int cardExpirationMonth,
         @WebParam(name = "cardExpirationYear") int cardExpirationYear,
         @WebParam(name = "cardVerificationCode")int cardVerificationCode,
         @WebParam(name = "cardBillingAmount") float cardBillingAmount,
         @WebParam(name = "businessPartner") int businessAssociate)

   {


   float cardLimit = 0;
   float cardPastTransactions = 0;
   String cardStatus="Active";
   boolean cardValidationStatus= false;
```


Πανεπιστήμιο Πειραιώς

```
Connection conn = null;
PreparedStatement pstmt = null;
ResultSet rs = null;


int queryStatus1 = 0;
int queryStatus2 = 0;
Float trans = null;
Float newtrans = null;
String returnValue = null;

double    CardNumber = Double.parseDouble(cardNumber);


   System.out.println("You called Validate card Service of Ethniki "
       + "Bank with input "
       + ""+ cardNumber +" "
       + ""+cardHolderName+" "
       + ""+cardExpirationMonth+" "
       + ""+cardExpirationYear+" "
       + ""+cardVerificationCode+" "
       + " "+CardNumber
       );


      try{
      conn = con.GetConnection("alphabankgrdb");

      String query = "SELECT cardLimit, cardMonthTransactions"
            + " FROM cardlistrecords WHERE "
            + "cardNumber=? "
            + "AND cardHolderName=? "
            + "AND cardExpirationMonth=? "
            + "AND cardExpirationYear=? "
            + "AND cardVerificationCode=? "
            + "AND cardStatus=? ";
       pstmt = conn.prepareStatement(query); // create a statement
       pstmt.setDouble(1, CardNumber); // set input parameter 1
       pstmt.setString(2, cardHolderName); // set input parameter 1
       pstmt.setInt(3, cardExpirationMonth); // set input parameter 1
       pstmt.setInt(4, cardExpirationYear); // set input parameter 1
       pstmt.setInt(5, cardVerificationCode); // set input parameter 1
       pstmt.setString(6, cardStatus);
       rs = pstmt.executeQuery();
```


Πανεπιστήμιο Πειραιώς

```
        while(rs.next()){
        cardLimit = rs.getFloat(1);
        cardPastTransactions = rs.getFloat(2);
        }

        pstmt.close();
        con.closeConnection(conn);


    }catch(Exception ex){System.err.print(ex);}
       System.out.println(cardLimit +"  "+cardPastTransactions);


        if  (cardLimit>(cardPastTransactions+cardBillingAmount)){
        cardValidationStatus=true;
        }

        if(cardValidationStatus){



    Date today = new Date();
    Timestamp ts = new java.sql.Timestamp(today.getTime());

    try{
        conn = con.GetConnection("alphabankgrdb");

        String query = "INSERT "
              + "INTO"
              + " cardtransactions"
              + "(cardNumber,"
              + " cardBillingAmount, "
              + "companyId, "
              + "date) "
              + "VALUES "
              + "(?,?,?,?)";
        pstmt = conn.prepareStatement(query); // create a statement
        pstmt.setDouble(1, CardNumber); // set input parameter 1
        pstmt.setFloat(2, cardBillingAmount);
        pstmt.setInt(3, businessAssociate);
        pstmt.setTimestamp(4, ts);

        queryStatus1= pstmt.executeUpdate();
        if(queryStatus1==1){
        System.out.println("New record added to Transactions table");
        }
```


Πανεπιστήμιο Πειραιώς

```
        pstmt.close();

        query="SELECT "
              + "cardMonthTransactions "
              + "FROM "
              + "cardlistrecords "
              + "WHERE "
              + "cardNumber=?";
        pstmt = conn.prepareStatement(query);
        pstmt.setDouble(1, CardNumber);

        rs = pstmt.executeQuery();

        while(rs.next()){
        trans = rs.getFloat(1);
         }

        newtrans = cardBillingAmount+trans;

        pstmt.close();

        query="UPDATE "
              + "cardlistrecords "
              + "SET "
              + "cardMonthTransactions=? "
              + "WHERE "
              + "cardNumber=?";
        pstmt = conn.prepareStatement(query);
        pstmt.setFloat(1, newtrans);
        pstmt.setDouble(2, CardNumber);

        queryStatus2= pstmt.executeUpdate();

        if(queryStatus2==1){
        System.out.println("Cardrecords table updated ");
        }

        pstmt.close();

        con.closeConnection(conn);


    }catch(Exception ex){
       System.err.print(ex);
    }
    if(queryStatus1==1 && queryStatus2==1){
```


Πανεπιστήμιο Πειραιώς

```
      cardValidationStatus=true;
   }else{
      cardValidationStatus=false;
   }

         }
          return cardValidationStatus;
   }

}
```

Κώδικας 2:EthnikiGrWS.java

```
package com.ethnikigr.webservice;

import com.jdbcquery.code.JdbcConnect;
import java.math.BigDecimal;
import java.sql.*;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.Date;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

@WebService(serviceName = "EthnikiGrWS")
public class EthnikiGrWS {
JdbcConnect con = new JdbcConnect();

   @WebMethod(operationName = "InsertCardTransaction")
   public boolean ValidateCreditCard(
         @WebParam(name = "cardNumber") String cardNumber,
         @WebParam(name = "cardHoldersName") String cardHolderName,
         @WebParam(name = "cardExpirationMonth") int cardExpirationMonth,
         @WebParam(name = "cardExpirationYear") int cardExpirationYear,
         @WebParam(name = "cardVerificationCode")int cardVerificationCode,
         @WebParam(name = "cardBillingAmount") float cardBillingAmount,
         @WebParam(name = "businessPartner") int businessAssociate)

   {


   float cardLimit = 0;
   float cardPastTransactions = 0;
   String cardStatus="Active";
   boolean cardValidationStatus= false;
```


Πανεπιστήμιο Πειραιώς

```
Connection conn = null;
PreparedStatement pstmt = null;
ResultSet rs = null;


int queryStatus1 = 0;
int queryStatus2 = 0;
Float trans = null;
Float newtrans = null;
String returnValue = null;


//BigDecimal bdCardNumber = new BigDecimal();
double    CardNumber = Double.parseDouble(cardNumber);


   System.out.println("You called Validate card Service of Ethniki "
        + "Bank with input "
        + ""+ cardNumber +" "
        + ""+cardHolderName+" "
        + ""+cardExpirationMonth+" "
        + ""+cardExpirationYear+" "
        + ""+cardVerificationCode+" "
        + " "+CardNumber
        );


      try{
      conn = con.GetConnection("ethnikigrdb");

     String query = "SELECT cardLimit, cardMonthTransactions"
           + " FROM cardlistrecords WHERE "
           + "cardNumber=? "
           + "AND cardHolderName=? "
           + "AND cardExpirationMonth=? "
           + "AND cardExpirationYear=? "
           + "AND cardVerificationCode=? "
           + "AND cardStatus=? ";
      pstmt = conn.prepareStatement(query); // create a statement
      pstmt.setDouble(1, CardNumber); // set input parameter 1
      pstmt.setString(2, cardHolderName); // set input parameter 1
      pstmt.setInt(3, cardExpirationMonth); // set input parameter 1
      pstmt.setInt(4, cardExpirationYear); // set input parameter 1
      pstmt.setInt(5, cardVerificationCode); // set input parameter 1
      pstmt.setString(6, cardStatus);
```


Πανεπιστήμιο Πειραιώς

```
        rs = pstmt.executeQuery();


        while(rs.next()){
        cardLimit = rs.getFloat(1);
        cardPastTransactions = rs.getFloat(2);
        }

        pstmt.close();
        con.closeConnection(conn);


   }catch(Exception ex){System.err.print(ex);}
      System.out.println(cardLimit +"  "+cardPastTransactions);


        if  (cardLimit>(cardPastTransactions+cardBillingAmount)){
        cardValidationStatus=true;
        }

        if(cardValidationStatus){



   Date today = new Date();
   Timestamp ts = new java.sql.Timestamp(today.getTime());

   try{
        conn = con.GetConnection("ethnikigrdb");

        String query = "INSERT INTO cardtransactions(cardNumber, cardBillingAmount,
companyId, date) VALUES (?,?,?,?)";
        pstmt = conn.prepareStatement(query); // create a statement
        pstmt.setDouble(1, CardNumber); // set input parameter 1
        pstmt.setFloat(2, cardBillingAmount);
        pstmt.setInt(3, businessAssociate);
        pstmt.setTimestamp(4, ts);

        queryStatus1= pstmt.executeUpdate();
        if(queryStatus1==1){
        System.out.println("New record added to Transactions table");
        }
        pstmt.close();

        query="SELECT cardMonthTransactions FROM cardlistrecords WHERE
cardNumber=?";
```


Πανεπιστήμιο Πειραιώς

```
        pstmt = conn.prepareStatement(query);
        pstmt.setDouble(1, CardNumber);

        rs = pstmt.executeQuery();

        while(rs.next()){
        trans = rs.getFloat(1);
         }

        newtrans = cardBillingAmount+trans;

        pstmt.close();

        query="UPDATE cardlistrecords SET cardMonthTransactions=? WHERE
cardNumber=?";
        pstmt = conn.prepareStatement(query);
        pstmt.setFloat(1, newtrans);
        pstmt.setDouble(2, CardNumber);

        queryStatus2= pstmt.executeUpdate();

        if(queryStatus2==1){
        System.out.println("Cardrecords table updated ");
        }

        pstmt.close();

        con.closeConnection(conn);


   }catch(Exception ex){
      System.err.print(ex);
   }
   if(queryStatus1==1 && queryStatus2==1){
      cardValidationStatus=true;
   }else{
      cardValidationStatus=false;
   }

        }
         return cardValidationStatus;
   }

}
```


Πανεπιστήμιο Πειραιώς

Κώδικας 3: Status.java

```
package com.visa.src;

public class Status {

   boolean status;
   String bank;

   public Status(String bank, boolean status){
   this.bank=bank;
   this.status=status;
   }

   public Status() {
   }

   public void setBank(String bank){
   this.bank =bank;
   }

   public String getBank(){
   return this.bank;
   }

    public void setStatus(boolean status){
   this.status =status;
   }

   public boolean getStatus(){
   return this.status;
   }

}
```

Κώδικας 4: JdbcConnect.java

```
package com.jdbcquery.code;

import java.sql.*;

/**
 *
 * @author nodas
 */
public class JdbcConnect {
```


Πανεπιστήμιο Πειραιώς

```
public Connection  GetConnection(String DbName){

  Connection conn = null;

      try
      {
         String userName = "thesisTestDb";
         String password = "12345678";
         String url = "jdbc:mysql://176.34.182.139/"+DbName;
         Class.forName ("com.mysql.jdbc.Driver").newInstance ();
         conn = DriverManager.getConnection (url, userName, password);
      }
      catch (Exception e)
      {
         System.err.println (e);
      }
      return conn;
   }

   public void closeConnection(Connection conn){

       if (conn != null)
        {
           try
           {
              conn.close ();
             // System.out.println ("Database connection terminated");
           }
           catch (Exception e) { /* ignore close errors */ }
        }
   }
}
```


Πανεπιστήμιο Πειραιώς

6.2 Κώδικας προσομοίωσης Web services προμηθευτών

Κώδικας 5: OrderReturn.java

```
package com.supplier1.clasees.orders;

public class OrderReturn {
   private boolean status;
   private String orderId;
   private float orderPrice;

   public OrderReturn() {
   }

   public OrderReturn(boolean status, String orderId, float orderPrice) {
      this.status = status;
      this.orderId = orderId;
      this.orderPrice = orderPrice;
   }

   public String getOrderId() {
      return orderId;
   }

   public void setOrderId(String orderId) {
      this.orderId = orderId;
   }

   public float getOrderPrice() {
      return orderPrice;
   }

   public void setOrderPrice(float orderPrice) {
      this.orderPrice = orderPrice;
   }

   public boolean isStatus() {
      return status;
   }

   public void setStatus(boolean status) {
      this.status = status;
   }

}
```

Κώδικας 6: OrderSubmit.java

Πανεπιστήμιο Πειραιώς

```
package com.supplier1.clasees.orders;

public class OrderSubmit {
   private int productId;
   private int productQuantity;
   private int partnerId;

   public OrderSubmit() {
   }

   public OrderSubmit(int productId, int productQuantity, int partnerId) {
      this.productId = productId;
      this.productQuantity = productQuantity;
      this.partnerId = partnerId;
   }

   public int getPartnerId() {
      return partnerId;
   }

   public void setPartnerId(int partnerId) {
      this.partnerId = partnerId;
   }

   public int getProductId() {
      return productId;
   }

   public void setProductId(int productId) {
      this.productId = productId;
   }

   public int getProductQuantity() {
      return productQuantity;
   }

   public void setProductQuantity(int productQuantity) {
      this.productQuantity = productQuantity;
   }

}
```

Πανεπιστήμιο Πειραιώς

Κώδικας 7: Supplier1WS

```
package com.supplier1.webservice;

import com.jdbcquery.code.JdbcConnect;
import com.supplier1.clasees.orders.OrderReturn;
import com.supplier1.clasees.orders.OrderSubmit;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

@WebService(serviceName = "Supplier1WS")
public class Supplier1WS {
   private  JdbcConnect con = new JdbcConnect();
   private   Connection conn;
   private   PreparedStatement pstmt;
   private  ResultSet rs;
   /**
    * This a Web Service for partners, to submit their orders
    */
   @WebMethod(operationName = "OrderSubmit")
   public OrderReturn OrderSumbit(@WebParam(name = "order") OrderSubmit order) {

   OrderReturn returnValue = new OrderReturn();
   int queryStatus1=0;

   System.out.println(""+
         "Order Submit Supplier1:"+
         "\n Produtct ID: "+order.getProductId()+
         "\n Product Quantity: "+order.getProductQuantity()+
         "\n Partner ID: "+ order.getPartnerId()
         );

    try{
         conn = con.GetConnection("supplier01db");

         String query = "INSERT INTO orders ("
               + "PRODUCT_ID," //1
               + "PRODUCT_QUANTITY," //2
               + "PARTNER_ID " //3
               + ") VALUES (?,?,?)";

         pstmt = conn.prepareStatement(query, Statement.RETURN_GENERATED_KEYS);
```


Πανεπιστήμιο Πειραιώς

```
// create a statement
        pstmt.setInt(1, order.getProductId());
        pstmt.setInt(2, order.getProductQuantity());
        pstmt.setInt(3, order.getPartnerId());

        queryStatus1=pstmt.executeUpdate();
        rs=pstmt.getGeneratedKeys();

        while(rs.next())
        {
           int i =0;
        returnValue.setOrderId("SUP-1111-"+rs.getInt(1));
        }
        if(queryStatus1==1){
        System.out.println("New row added to Supplier1's Order table");
        returnValue.setStatus(true);

        String query2 = "SELECT "
              + "PRODUCT_PRICE "
              + "FROM "
              + "products "
              + "WHERE "
              + "PRODUCT_ID=?";


        pstmt = conn.prepareStatement(query2); // create a statement
        pstmt.setInt(1, order.getProductId());
        rs=pstmt.executeQuery();

         while(rs.next())
          {
          returnValue.setOrderPrice(rs.getFloat(1)*order.getProductQuantity());
          }

        }



       pstmt.close();
       con.closeConnection(conn);

  }catch(Exception ex){
     System.err.print(ex);
  }
      System.out.println("Supplier1 Output:"
          + "\n Order ID: "+returnValue.getOrderId()
```


Πανεπιστήμιο Πειραιώς

```
            + "\n Order Price: "+returnValue.getOrderPrice()
            + "\n Order Status: "+returnValue.isStatus()
            );
      return returnValue;
   }
}
```

Κώδικας 8: Supplier2WS.java

```
package com.supplier2.webservice;

import com.jdbcquery.code.JdbcConnect;
import com.supplier1.classes.orders.OrderReturn;
import com.supplier1.classes.orders.OrderSubmit;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

@WebService(serviceName = "Supplier2WS")
public class Supplier2WS {
   private  JdbcConnect con = new JdbcConnect();
   private   Connection conn;
   private   PreparedStatement pstmt;
   private  ResultSet rs;
   /**
    * This a Web Service for partners, to submit their orders
    */
   @WebMethod(operationName = "OrderSubmit")
   public OrderReturn OrderSumbit(@WebParam(name = "order") OrderSubmit order) {

   OrderReturn returnValue = new OrderReturn();
   int queryStatus1=0;

   System.out.println(""+
         "Order Submit Supplier2:"+
         "\n Produtct ID: "+order.getProductId()+
         "\n Product Quantity: "+order.getProductQuantity()+
         "\n Partner ID: "+ order.getPartnerId()
         );
    try{
         conn = con.GetConnection("supplier02db");
```


Πανεπιστήμιο Πειραιώς

```
        String query = "INSERT INTO orders ("
              + "PRODUCT_ID," //1
              + "PRODUCT_QUANTITY," //2
              + "PARTNER_ID " //3
              + ") VALUES (?,?,?)";

        pstmt = conn.prepareStatement(query, Statement.RETURN_GENERATED_KEYS);
// create a statement
        pstmt.setInt(1, order.getProductId());
        pstmt.setInt(2, order.getProductQuantity());
        pstmt.setInt(3, order.getPartnerId());

        queryStatus1=pstmt.executeUpdate();
        rs=pstmt.getGeneratedKeys();

        while(rs.next())
        {
           int i =0;
        returnValue.setOrderId("SUP-2222-"+rs.getInt(1));
        }
        if(queryStatus1==1){
        System.out.println("New row added to Supplier2's Order table");
        returnValue.setStatus(true);

        String query2 = "SELECT "
              + "PRODUCT_PRICE "
              + "FROM "
              + "products "
              + "WHERE "
              + "PRODUCT_ID=?";


        pstmt = conn.prepareStatement(query2); // create a statement
        pstmt.setInt(1, order.getProductId());
        rs=pstmt.executeQuery();

         while(rs.next())
          {
          returnValue.setOrderPrice(rs.getFloat(1)*order.getProductQuantity());
          }

        }



       pstmt.close();
```


Πανεπιστήμιο Πειραιώς

```
        con.closeConnection(conn);

    }catch(Exception ex){
       System.err.print(ex);
    }
       System.out.println("Supplier2 Output:"
            + "\n Order ID: "+returnValue.getOrderId()
            + "\n Order Price: "+returnValue.getOrderPrice()
            + "\n Order Status: "+returnValue.isStatus()
            );

       return returnValue;
    }
}
```

Κώδικας 9:JdbcConnect.java

```
package com.jdbcquery.code;

import java.sql.*;

public class JdbcConnect {

public Connection  GetConnection(String DbName){

   Connection conn = null;

        try
        {
           String userName = "thesisTestDb";
           String password = "12345678";
           String url = "jdbc:mysql://176.34.182.139/"+DbName;
           Class.forName ("com.mysql.jdbc.Driver").newInstance ();
           conn = DriverManager.getConnection (url, userName, password);
        }
        catch (Exception e)
        {
           System.err.println (e);
        }
        return conn;
     }

     public void closeConnection(Connection conn){

        if (conn != null)
          {
```


Πανεπιστήμιο Πειραιώς

```
            try
            {
               conn.close ();
              // System.out.println ("Database connection terminated");
            }
            catch (Exception e) { /* ignore close errors */ }
         }
   }
}
```

6.3 Κώδικας προσομοίωσης Web service μεταφορικής εταιρίας

Κώδικας 10: OrderReturn.java

```
package com.shippingcompany.classes;

public class OrderReturn {
   int orderId;
   boolean status;

   public OrderReturn() {
   }

   public OrderReturn(int orderId, boolean status) {
      this.orderId = orderId;
      this.status = status;
   }

   public int getOrderId() {
      return orderId;
   }

   public void setOrderId(int orderId) {
      this.orderId = orderId;
   }

   public boolean isStatus() {
      return status;
   }

   public void setStatus(boolean status) {
```


Πανεπιστήμιο Πειραιώς

```
        this.status = status;
    }


}
```

Κώδικας 11:ShippingCompanyWS.java

```
package com.shippingcompany.web.services;

import com.jdbcquery.code.JdbcConnect;
import com.shippingcompany.classes.OrderReturn;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

/**
 *
 * @author Nodas
 */
@WebService(serviceName = "ShippingCompanyWS")
public class ShippingCompanyWS {
private  JdbcConnect con = new JdbcConnect();
private   Connection conn;
private   PreparedStatement pstmt;
private  ResultSet rs;
    /**
     * This is a sample web service operation
     */
    @WebMethod(operationName = "GoodsReceptionRequest")
    public OrderReturn goodsReceptionFromPatners(
            @WebParam(name = "PartnerId") String partnerId,
            @WebParam(name = "PartnerLocationId") int partnerLocationId) {

    OrderReturn returnValue = new OrderReturn();
    int queryStatus1=0;

    System.out.println(""+
            "Request Submit For Shipping:"+
            "\n Partner ID: "+partnerId+
            "\n Parthner Location ID: "+partnerLocationId
            );
```

Πανεπιστήμιο Πειραιώς

```
    try{
        conn = con.GetConnection("shippingcompany01db");

        String query = "INSERT INTO orders ("
              + "PARTNER_ID,"
              + "PARTNER_LOCATION_ID "
              + ") VALUES (?,?)";

        pstmt = conn.prepareStatement(query, Statement.RETURN_GENERATED_KEYS);
// create a statement
        pstmt.setString(1, partnerId);
        pstmt.setInt(2, partnerLocationId);

        queryStatus1=pstmt.executeUpdate();
        rs=pstmt.getGeneratedKeys();

        while(rs.next())
        {
           int i =0;
        returnValue.setOrderId(rs.getInt(1));
        }
        if(queryStatus1==1){
        System.out.println("New row added to Supplier1's Order table");
        returnValue.setStatus(true);

        }

        rs.close();
        pstmt.close();
        con.closeConnection(conn);

   }catch(Exception ex){
      System.err.print(ex);
   }
       System.out.println("Supplier1 Output:"
            + "\n Order ID: "+returnValue.getOrderId()
            + "\n Order Status: "+returnValue.isStatus()
            );

      return returnValue;
   }
}
```

Κώδικας 12:JdbcConnect.java

```
package com.jdbcquery.code;
```


Πανεπιστήμιο Πειραιώς

```
import java.sql.*;

public class JdbcConnect {

public Connection  GetConnection(String DbName){

   Connection conn = null;

        try
        {
           String userName = "thesisTestDb";
           String password = "12345678";
           String url = "jdbc:mysql://176.34.182.139/"+DbName;
           Class.forName ("com.mysql.jdbc.Driver").newInstance ();
           conn = DriverManager.getConnection (url, userName, password);
        }
        catch (Exception e)
        {
           System.err.println (e);
        }
        return conn;
    }

    public void closeConnection(Connection conn){

         if (conn != null)
          {
             try
             {
                conn.close ();
               // System.out.println ("Database connection terminated");
             }
             catch (Exception e) { /* ignore close errors */ }
          }
    }
 }
```

6.4 Κώδικας Web services παραγγελίας

Κώδικας 13: OrderProductDetails.java

```
package com.store.brandname.classes.repository;

public class OrderProductDetails {
```


Πανεπιστήμιο Πειραιώς

```
  private int productId;
  private String productName;
  private int productQuantity;
  private float productUnitPrice;

  public OrderProductDetails() {
  }

  public OrderProductDetails(int productId, String productName, int productQuantity,
float productUnitPrice) {
    this.productId = productId;
    this.productName = productName;
    this.productQuantity = productQuantity;
    this.productUnitPrice = productUnitPrice;
  }

  public int getProductId() {
    return productId;
  }

  public void setProductId(int productId) {
    this.productId = productId;
  }

  public String getProductName() {
    return productName;
  }

  public void setProductName(String productName) {
    this.productName = productName;
  }

  public int getProductQuantity() {
    return productQuantity;
  }

  public void setProductQuantity(int productQuantity) {
    this.productQuantity = productQuantity;
  }

  public float getProductUnitPrice() {
    return productUnitPrice;
  }

  public void setProductUnitPrice(float productUnitPrice) {
```


Πανεπιστήμιο Πειραιώς

```
        this.productUnitPrice = productUnitPrice;
    }
}
```

Κώδικας 14: OrderTable.java

```
package com.store.brandname.classes.repository;

public class OrderTable {
    private int userId;
    private float orderPrice;
    private int paymentMethodId;
    private int deliveryAddressId;

    public OrderTable() {
    }

    public OrderTable(int userId, float orderPrice, int paymentMethodId, int
deliveryAddressId) {
        this.userId = userId;
        this.orderPrice = orderPrice;
        this.paymentMethodId = paymentMethodId;
        this.deliveryAddressId = deliveryAddressId;
    }

    public int getDeliveryAddressId() {
        return deliveryAddressId;
    }

    public void setDeliveryAddressId(int deliveryAddressId) {
        this.deliveryAddressId = deliveryAddressId;
    }

    public float getOrderPrice() {
        return orderPrice;
    }

    public void setOrderPrice(float orderPrice) {
        this.orderPrice = orderPrice;
    }

    public int getPaymentMethodId() {
        return paymentMethodId;
    }

    public void setPaymentMethodId(int paymentMethodId) {
        this.paymentMethodId = paymentMethodId;
```


Πανεπιστήμιο Πειραιώς

```
    }

    public int getUserId() {
        return userId;
    }

    public void setUserId(int userId) {
        this.userId = userId;
    }


}
```

Κώδικας 15: OrderTableReturn.java

```
package com.store.brandname.classes.repository;

public class OrderTableReturn {
    private boolean status;
    private int orderId;

    public OrderTableReturn() {
    }

    public OrderTableReturn(boolean status, int orderId) {
        this.status = status;
        this.orderId = orderId;
    }

    public int getOrderId() {
        return orderId;
    }

    public void setOrderId(int orderId) {
        this.orderId = orderId;
    }

    public boolean isStatus() {
        return status;
    }

    public void setStatus(boolean status) {
        this.status = status;
    }

}
```

Πανεπιστήμιο Πειραιώς

Κώδικας 16:OrderToProductTable.java

```
package com.store.brandname.classes.repository;

public class OrderToProductTable {
   private int orderId;
   private int productId;
   private int productQuantity;

   public OrderToProductTable() {
   }

   public OrderToProductTable(int orderId, int productId, int productQuantity) {
      this.orderId = orderId;
      this.productId = productId;
      this.productQuantity = productQuantity;
   }

   public int getOrderId() {
      return orderId;
   }

   public void setOrderId(int orderId) {
      this.orderId = orderId;
   }

   public int getProductId() {
      return productId;
   }

   public void setProductId(int productId) {
      this.productId = productId;
   }

   public int getProductQuantity() {
      return productQuantity;
   }

   public void setProductQuantity(int productQuantity) {
      this.productQuantity = productQuantity;
   }
```

Πανεπιστήμιο Πειραιώς

```
}
```

Κώδικας 17:StatusAndSupplierOrderProductsReturn.java

```
package com.store.brandname.classes.repository;

public class StatusAndSupplierOrderProductsReturn {

   private boolean needSupplierStatus;
   private int productQuantity;
   private int supplierId;
   private int supplierProductId;

   public StatusAndSupplierOrderProductsReturn() {
   }

   public StatusAndSupplierOrderProductsReturn(boolean needSupplierStatus, int
productQuantity, int supplierId, int supplierProductId) {
      this.needSupplierStatus = needSupplierStatus;
      this.productQuantity = productQuantity;
      this.supplierId = supplierId;
      this.supplierProductId = supplierProductId;
   }

   public boolean isNeedSupplierStatus() {
      return needSupplierStatus;
   }

   public void setNeedSupplierStatus(boolean needSupplierStatus) {
      this.needSupplierStatus = needSupplierStatus;
   }

   public int getProductQuantity() {
      return productQuantity;
   }

   public void setProductQuantity(int productQuantity) {
      this.productQuantity = productQuantity;
   }

   public int getSupplierId() {
      return supplierId;
   }
```


Πανεπιστήμιο Πειραιώς

```
    public void setSupplierId(int supplierId) {
        this.supplierId = supplierId;
    }

    public int getSupplierProductId() {
        return supplierProductId;
    }

    public void setSupplierProductId(int supplierProductId) {
        this.supplierProductId = supplierProductId;
    }

}
```

Κώδικας 18:RepositoryWS

```
package com.store.brandname.webservice;

import com.jdbcquery.code.JdbcConnect;
import com.store.brandname.classes.repository.*;
import java.sql.*;
import java.util.ArrayList;
import java.util.Date;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

@WebService(serviceName = "RepositoryWS")
public class RepositoryWS {
private JdbcConnect con = new JdbcConnect();
private Connection conn;
private PreparedStatement pstmt;
private ResultSet rs;
    /**
     * Web service operation
     */
    @WebMethod(operationName = "InsertOrder")
    public OrderTableReturn InsertOrder(
            @WebParam(name = "order") OrderTable order)

    {

    int queryStatus1 = 0;

    OrderTableReturn returnValue = new OrderTableReturn();
    returnValue.setStatus(false);
```


Πανεπιστήμιο Πειραιώς

```
  Date today = new Date();
  Timestamp ts = new java.sql.Timestamp(today.getTime());

  System.out.println(""+
       "InsertOrderReceived:"+
       "\n UserID: "+order.getUserId()+
       "\n OrderPrice: "+order.getOrderPrice()+
       "\n PaymentMethodId: "+ order.getPaymentMethodId()+
       "\n DeliveryAddressId: "+ order.getDeliveryAddressId()
       );

   try{
       conn = con.GetConnection("store01db");

       String query = "INSERT INTO orders ("
            + "USER_ID," //1
            + "ORDER_PRICE," //2
            + "ORDER_DATE, " //3
            + "ADDRESS_ID," //5
            + "PAYMENT_METHOD_ID" //6
            + ") VALUES (?,?,?,?,?)";

       pstmt = conn.prepareStatement(query, Statement.RETURN_GENERATED_KEYS);
// create a statement
       pstmt.setInt(1, order.getUserId()); // set input parameter 1
       pstmt.setFloat(2, order.getOrderPrice());
       pstmt.setTimestamp(3, ts);
       pstmt.setInt(4, order.getDeliveryAddressId());
       pstmt.setInt(5, order.getPaymentMethodId());

       queryStatus1= pstmt.executeUpdate();
       rs=pstmt.getGeneratedKeys();

       while(rs.next())
       {
       returnValue.setOrderId(rs.getInt(1));
       }
       if(queryStatus1==1){
       System.out.println("New row added to Order table");
       returnValue.setStatus(true);
       }
       pstmt.close();
      con.closeConnection(conn);

  }catch(Exception ex){
```


Πανεπιστήμιο Πειραιώς

```
    System.err.print(ex);
}

        return returnValue;
}


/*
 *
 * Web service
 *
 */
@WebMethod(operationName = "InsertProductToOrder")
public boolean InsertProductToOrder(
        @WebParam(name = "productToOrder") OrderToProductTable product)

{


int queryStatus1 = 0;
boolean returnValue=false;

System.out.println(""+
        "InsertOrderToProductReceived:"+
        "\n OrderId: "+product.getOrderId()+
        "\n ProductId: "+product.getProductId()+
        "\n ProductQuantity "+product.getProductQuantity()
        );

 try{
        conn = con.GetConnection("store01db");

        String query = "INSERT INTO products_to_order ("
                + "ORDER_ID," //1
                + "PRODUCT_ID," //2
                + "QUANTITY " //3
                + ") VALUES (?,?,?)";

        pstmt = conn.prepareStatement(query); // create a statement
        pstmt.setInt(1, product.getOrderId()); // set input parameter 1
        pstmt.setInt(2, product.getProductId());
        pstmt.setInt(3, product.getProductQuantity());

        queryStatus1= pstmt.executeUpdate();

        if(queryStatus1==1){
```


Πανεπιστήμιο Πειραιώς

```
        System.out.println("New row added to ProductToOrder table");
        returnValue=true;
        }
        pstmt.close();
        con.closeConnection(conn);

  }catch(Exception ex){
     System.err.print(ex);
  }

         return returnValue;
  }


   /*
   *
   * Web service
   *
   */
  @WebMethod(operationName = "AvailabilityAndCommitment")
  public StatusAndSupplierOrderProductsReturn AvailabilityAndCommitment(
        @WebParam(name = "productToOrder") OrderToProductTable product)
  {
     boolean re=false;
     int quantity=0, supplierId=0, supplierProductId=0;
     int checkInsertResult=0;
     int repositoryCase=0;
     StatusAndSupplierOrderProductsReturn returnValue =
          new StatusAndSupplierOrderProductsReturn();
     try{

        conn = con.GetConnection("store01db");

        String query = "SELECT "
             + "REPOSITORY_QUANTITY, "
             + "SUPPLIER_ID, "
             + "SUPPLIER_PRODUCT_ID "
             + "FROM "
             + "products "
             + "WHERE "
             + "PRODUCT_ID=?";


        pstmt = conn.prepareStatement(query); // create a statement
        pstmt.setInt(1, product.getProductId());
        rs=pstmt.executeQuery();
```


Πανεπιστήμιο Πειραιώς

```
while(rs.next())
{
quantity=rs.getInt(1);
supplierId=rs.getInt(2);
supplierProductId=rs.getInt(3);
}
//
//Commited
//
if(quantity>product.getProductQuantity() ||
      quantity==product.getProductQuantity())
{
 repositoryCase=1;
 System.out.println("Product Commited...");
 quantity=quantity-product.getProductQuantity();

   String query1="UPDATE "
         + "products "
         + "SET "
         + "REPOSITORY_QUANTITY=? "
         + "WHERE "
         + "PRODUCT_ID=?";

pstmt = conn.prepareStatement(query1); // create a statement
pstmt.setInt(1, quantity);
pstmt.setInt(2, product.getProductId());
checkInsertResult=pstmt.executeUpdate();

if(checkInsertResult!=0){

   String query2="UPDATE "
         + "products_to_order "
         + "SET "
         + "STATUS_ID=? "
         + "WHERE "
         + "PRODUCT_ID=? "
         + "AND "
         + "ORDER_ID=?";

pstmt = conn.prepareStatement(query2); // create a statement
pstmt.setInt(1, 3);
pstmt.setInt(2, product.getProductId());
pstmt.setInt(3, product.getOrderId());
pstmt.executeUpdate();
System.out.println("Product Commited Ended");
```


Πανεπιστήμιο Πειραιώς

```
quantity=0;
returnValue.setNeedSupplierStatus(false);
}

//Partially Commited
}else if(quantity<product.getProductQuantity() &&
      quantity>0){

System.out.println("Product Partially Commited...");



String query1="UPDATE "
        + "products "
        + "SET "
        + "REPOSITORY_QUANTITY=? "
        + "WHERE "
        + "PRODUCT_ID=?";

pstmt = conn.prepareStatement(query1); // create a statement
pstmt.setInt(1, 0);
pstmt.setInt(2, product.getProductId());
checkInsertResult=pstmt.executeUpdate();

if(checkInsertResult!=0){

   String query2="UPDATE "
        + "products_to_order "
        + "SET "
        + "STATUS_ID=? "
        + "WHERE "
        + "PRODUCT_ID=? "
        + "AND "
        + "ORDER_ID=?";

pstmt = conn.prepareStatement(query2); // create a statement
pstmt.setInt(1, 2);
pstmt.setInt(2, product.getProductId());
pstmt.setInt(3, product.getOrderId());
pstmt.executeUpdate();

quantity=product.getProductQuantity()-quantity;

System.out.println("Product Partially Commited Ended"+quantity);
}
repositoryCase=2;
```


Πανεπιστήμιο Πειραιώς

```
        //
        //Uncommited or Partially Commited
        //
        }

        if(repositoryCase==0 || repositoryCase==2){
           System.out.println("Product Uncommited or Partially Commited"
             + " Return Supplier Information");
           if(repositoryCase==0){
             quantity=product.getProductQuantity();
           }


        returnValue.setProductQuantity(quantity);
        returnValue.setSupplierId(supplierId);
        returnValue.setSupplierProductId(supplierProductId);
        returnValue.setNeedSupplierStatus(true);
        System.out.println("\nOrder For Supplier: \nQuantity: "+quantity+"\nSupplier:
"+supplierId+
             "\nSupplier PID "+supplierProductId);
        }

        pstmt.close();
        con.closeConnection(conn);

      }catch(Exception ex){
      System.err.print(ex);
   }


      return returnValue;
   }

   @WebMethod(operationName = "updateOrderWithSupplierOrder")
   public void updateOrderWithSupplierOrder(
        @WebParam(name = "ProductToOrder") OrderToProductTable product,
        @WebParam(name = "SupplierOrderId") String supplierOrderId)
        {
        System.out.println("update Order With Supplier Order");

          try{

        conn = con.GetConnection("store01db");
           String query1="UPDATE "
              + "products_to_order "
              + "SET "
```


Πανεπιστήμιο Πειραιώς

```
                + "SUPPLIER_ORDER_ID=? "
                + "WHERE "
                + "PRODUCT_ID=? "
                + "AND "
                + "ORDER_ID=? ";

        pstmt = conn.prepareStatement(query1); // create a statement
        pstmt.setString(1, supplierOrderId);
        pstmt.setInt(2, product.getProductId());
        pstmt.setInt(3, product.getOrderId());
        pstmt.executeUpdate();

        pstmt.close();
        con.closeConnection(conn);

           }catch(Exception ex){
     System.err.print(ex);
              }
        }

@WebMethod(operationName = "CheckOrderStatus")
public boolean checkOrderStatus(
        @WebParam(name = "OrderId") int orderId,
        @WebParam(name = "Records") int rows)
        {

        System.out.println("Check Order Status");
        int rowCount=0;
        boolean returnValue=false;

           try{

        conn = con.GetConnection("store01db");
            String query1="SELECT "
                + "COUNT(*) "
                + "FROM "
                + "products_to_order "
                + "WHERE "
                + "STATUS_ID=? "
                + "AND "
                + "ORDER_ID=? ";

        pstmt = conn.prepareStatement(query1); // create a statement
        pstmt.setInt(1, 3);
        pstmt.setInt(2, orderId);
        rs=pstmt.executeQuery();
```


Πανεπιστήμιο Πειραιώς

```
        rs.next();
        rowCount=rs.getInt(1);

        if(rowCount==rows){

       returnValue=true;

        System.out.println("All products of order "+orderId
              + " are fulfilled");
        }

        pstmt.close();
        con.closeConnection(conn);

            }catch(Exception ex){
     System.err.print(ex);
              }

            return returnValue;
        }


@WebMethod(operationName = "updateProductsToOrderStatus")
public void updateProductsToOrderStatus(
        @WebParam(name = "SupplierOrderId") String supplierOrderId)
        {



            try{

        conn = con.GetConnection("store01db");
             String query1="UPDATE "
                  + "products_to_order "
                  + "SET "
                  + "STATUS_ID=? "
                  + "WHERE "
                  + "SUPPLIER_ORDER_ID=? ";

        pstmt = conn.prepareStatement(query1); // create a statement
        pstmt.setString(2, supplierOrderId);
        pstmt.setInt(1, 3);
        pstmt.executeUpdate();

        pstmt.close();
```


Πανεπιστήμιο Πειραιώς

```
        con.closeConnection(conn);

            }catch(Exception ex){
      System.err.print(ex);
              }

            System.out.println("update Order With Supplier Order");
        }

}
```

Κώδικας 19:JdbcConnect.java

```
package com.jdbcquery.code;

import java.sql.*;

public class JdbcConnect {

public Connection  GetConnection(String DbName){

   Connection conn = null;

       try
       {
          String userName = "thesisTestDb";
          String password = "12345678";
          String url = "jdbc:mysql://176.34.182.139/"+DbName;
          Class.forName ("com.mysql.jdbc.Driver").newInstance ();
          conn = DriverManager.getConnection (url, userName, password);
       }
       catch (Exception e)
       {
          System.err.println (e);
       }
       return conn;
    }

    public void closeConnection(Connection conn){

        if (conn != null)
        {
           try
           {
              conn.close ();
             // System.out.println ("Database connection terminated");
           }
```


Πανεπιστήμιο Πειραιώς

```
            catch (Exception e) { /* ignore close errors */ }
        }
    }
}
```

Κώδικας 20:AddressInformation.java

```
package com.store.brandname.classes.repository;

public class AddressInformation {
   private int addressId;
   private String addressStreet;
   private String addressTown;
   private String addressState;
   private int addressPostcode;
   private String addressRecipientName;
   private int userId;

   public AddressInformation() {
   }

   public AddressInformation(int addressId, String addressStreet,
         String addressTown, String addressState, int addressPostcode,
         String addressRecipientName, int userId) {
      this.addressId = addressId;
      this.addressStreet = addressStreet;
      this.addressTown = addressTown;
      this.addressState = addressState;
      this.addressPostcode = addressPostcode;
      this.addressRecipientName = addressRecipientName;
      this.userId = userId;
   }

   public int getAddressId() {
      return addressId;
   }

   public void setAddressId(int addressId) {
      this.addressId = addressId;
   }

   public int getAddressPostcode() {
      return addressPostcode;
   }

   public void setAddressPostcode(int addressPostcode) {
```


Πανεπιστήμιο Πειραιώς

```
    this.addressPostcode = addressPostcode;
  }

  public String getAddressRecipientName() {
    return addressRecipientName;
  }

  public void setAddressRecipientName(String addressRecipientName) {
    this.addressRecipientName = addressRecipientName;
  }

  public String getAddressState() {
    return addressState;
  }

  public void setAddressState(String addressState) {
    this.addressState = addressState;
  }

  public String getAddressStreet() {
    return addressStreet;
  }

  public void setAddressStreet(String addressStreet) {
    this.addressStreet = addressStreet;
  }

  public String getAddressTown() {
    return addressTown;
  }

  public void setAddressTown(String addressTown) {
    this.addressTown = addressTown;
  }

  public int getUserId() {
    return userId;
  }

  public void setUserId(int userId) {
    this.userId = userId;
  }


}
```


Πανεπιστήμιο Πειραιώς

Κώδικας 21: InsertReceiptReturn.java

```
package com.store.brandname.classes.repository;

public class InsertReceiptRetrurn {
   private int receiptId;
   private boolean status;

   public InsertReceiptRetrurn() {
   }

   public InsertReceiptRetrurn(int receiptId, boolean status) {
      this.receiptId = receiptId;
      this.status = status;
   }

   public int getReceiptId() {
      return receiptId;
   }

   public void setReceiptId(int receiptId) {
      this.receiptId = receiptId;
   }

   public boolean isStatus() {
      return status;
   }

   public void setStatus(boolean status) {
      this.status = status;
   }


}
```

Κώδικας 22: OrderInformation.java

```
package com.store.brandname.classes.repository;

import java.util.Date;
```


Πανεπιστήμιο Πειραιώς

```
public class OrderInformation {
   private int orderId;
   private int userId;
   private int addressId;
   private int receiptId;
   private String orderDate;
   private String comment;
   private String shippingTrackingId;
   private String paymentMethodName;
   private String orderStatusName;

   public OrderInformation() {
   }

   public OrderInformation(int orderId, int userId, int addressId,
         int receiptId, String orderDate, String comment,
         String shippingTrackingId, String paymentMethodName,
         String orderStatusName) {
      this.orderId = orderId;
      this.userId = userId;
      this.addressId = addressId;
      this.receiptId = receiptId;
      this.orderDate = orderDate;
      this.comment = comment;
      this.shippingTrackingId = shippingTrackingId;
      this.paymentMethodName = paymentMethodName;
      this.orderStatusName = orderStatusName;
   }

   public int getAddressId() {
      return addressId;
   }

   public void setAddressId(int addressId) {
      this.addressId = addressId;
   }

   public String getComment() {
      return comment;
   }

   public void setComment(String comment) {
      this.comment = comment;
   }
```


Πανεπιστήμιο Πειραιώς

```
public String getOrderDate() {
   return orderDate;
}

public void setOrderDate(String orderDate) {
   this.orderDate = orderDate;
}

public int getOrderId() {
   return orderId;
}

public void setOrderId(int orderId) {
   this.orderId = orderId;
}

public String getOrderStatusName() {
   return orderStatusName;
}

public void setOrderStatusName(String orderStatusName) {
   this.orderStatusName = orderStatusName;
}

public String getPaymentMethodName() {
   return paymentMethodName;
}

public void setPaymentMethodName(String paymentMethodName) {
   this.paymentMethodName = paymentMethodName;
}

public int getReceiptId() {
   return receiptId;
}

public void setReceiptId(int receiptId) {
   this.receiptId = receiptId;
}

public String getShippingTrackingId() {
   return shippingTrackingId;
}

public void setShippingTrackingId(String shippingTrackingId) {
   this.shippingTrackingId = shippingTrackingId;
```

Πανεπιστήμιο Πειραιώς

```
   }

   public int getUserId() {
      return userId;
   }

   public void setUserId(int userId) {
      this.userId = userId;
   }


}
```

Κώδικας 23: OrderProductInformation.java

```
package com.store.brandname.classes.repository;

public class OrderProductInformation {

   private int productId;
   private String productName;
   private int productQuantity;
   private float productUnitPrice;

   public OrderProductInformation() {
   }

   public OrderProductInformation(int productId, String productName, int
productQuantity, float productUnitPrice) {
      this.productId = productId;
      this.productName = productName;
      this.productQuantity = productQuantity;
      this.productUnitPrice = productUnitPrice;
   }

   public int getProductId() {
      return productId;
   }

   public void setProductId(int productId) {
      this.productId = productId;
   }

   public String getProductName() {
      return productName;
   }
```


Πανεπιστήμιο Πειραιώς

```
  public void setProductName(String productName) {
    this.productName = productName;
  }

  public int getProductQuantity() {
    return productQuantity;
  }

  public void setProductQuantity(int productQuantity) {
    this.productQuantity = productQuantity;
  }

  public float getProductUnitPrice() {
    return productUnitPrice;
  }

  public void setProductUnitPrice(float productUnitPrice) {
    this.productUnitPrice = productUnitPrice;
  }
}
```

Κώδικας 24: ReceiptInformation.java

```
package com.store.brandname.classes.repository;

import java.sql.Timestamp;

public class ReceiptInformation {
  private int receiptId;
  private String receiptType;
  private String receiptDate;

  public ReceiptInformation() {
  }

  public ReceiptInformation(int receiptId, String receiptType,
      String receiptDate) {
    this.receiptId = receiptId;
    this.receiptType = receiptType;
    this.receiptDate = receiptDate;
  }

  public String getReceiptDate() {
    return receiptDate;
  }
```

Πανεπιστήμιο Πειραιώς

```
  public void setReceiptDate(String receiptDate) {
    this.receiptDate = receiptDate;
  }

  public int getReceiptId() {
    return receiptId;
  }

  public void setReceiptId(int receiptId) {
    this.receiptId = receiptId;
  }

  public String getReceiptType() {
    return receiptType;
  }

  public void setReceiptType(String receiptType) {
    this.receiptType = receiptType;
  }


}
```

Κώδικας 25:UserInformation.java

```
package com.store.brandname.classes.repository;

public class UserInformation {
  private int userId;
  private String userName;
  private String userSurname;
  private String userEmail;
  private String userPhone;
  private String userMobile;
  private int userTIN;
  private String userTaxOffice;
  private String userProfession;

  public UserInformation() {
  }

  public UserInformation(int userId, String userName, String userSurname,
      String userEmail, String userPhone, String userMobile, int userTIN,
      String userTaxOffice, String userProfession) {
```

Πανεπιστήμιο Πειραιώς

```
    this.userId = userId;
    this.userName = userName;
    this.userSurname = userSurname;
    this.userEmail = userEmail;
    this.userPhone = userPhone;
    this.userMobile = userMobile;
    this.userTIN = userTIN;
    this.userTaxOffice = userTaxOffice;
    this.userProfession = userProfession;
  }

  public String getUserEmail() {
    return userEmail;
  }

  public void setUserEmail(String userEmail) {
    this.userEmail = userEmail;
  }

  public int getUserId() {
    return userId;
  }

  public void setUserId(int userId) {
    this.userId = userId;
  }

  public String getUserMobile() {
    return userMobile;
  }

  public void setUserMobile(String userMobile) {
    this.userMobile = userMobile;
  }

  public String getUserName() {
    return userName;
  }

  public void setUserName(String userName) {
    this.userName = userName;
  }

  public String getUserPhone() {
    return userPhone;
  }
```

Πανεπιστήμιο Πειραιώς

```
  public void setUserPhone(String userPhone) {
    this.userPhone = userPhone;
  }

  public String getUserProfession() {
    return userProfession;
  }

  public void setUserProfession(String userProfession) {
    this.userProfession = userProfession;
  }

  public String getUserSurname() {
    return userSurname;
  }

  public void setUserSurname(String userSurname) {
    this.userSurname = userSurname;
  }

  public int getUserTIN() {
    return userTIN;
  }

  public void setUserTIN(int userTIN) {
    this.userTIN = userTIN;
  }

  public String getUserTaxOffice() {
    return userTaxOffice;
  }

  public void setUserTaxOffice(String userTaxOffice) {
    this.userTaxOffice = userTaxOffice;
  }


}
```

Κώδικας 26: Repository1WS.java

```
package com.store.brandname.webservice;

import com.jdbcquery.code.JdbcConnect;
import com.store.brandname.classes.repository.*;
```


Πανεπιστήμιο Πειραιώς

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Date;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

/**
 *
 * @author Nodas
 */
@WebService(serviceName = "Repository1WS")
public class Repository1WS {
private JdbcConnect con = new JdbcConnect();
private Connection conn;
private PreparedStatement pstmt;
private ResultSet rs;

   /*
    *
    *Web service operation
    */
  @WebMethod(operationName = "GetOrderProductDetails")
  public OrderProductInformation[] GetOrderProductDetails(@WebParam(name =
        "orderId") int orderId) {

     System.out.println("Get Order Details Invoked with"
          + "OrderId: "+orderId);

     int countRows;
     ArrayList<OrderProductInformation> queryResult =
       new ArrayList<OrderProductInformation>();

     OrderProductInformation[] returnValue = null;
       try{

       conn = con.GetConnection("store01db");

       String query1="SELECT "
            + "p.PRODUCT_ID, "
            + "p.PRODUCT_NAME, "
            + "pt.QUANTITY, "
            + "p.PRODUCT_PRICE "
```


Πανεπιστήμιο Πειραιώς

```
            + "FROM "
            + "products p "
            + "LEFT OUTER JOIN "
            + "products_to_order pt "
            + "ON "
            + "pt.PRODUCT_ID=p.PRODUCT_ID "
            + "WHERE ORDER_ID=?";

      pstmt = conn.prepareStatement(query1); // create a statement
      pstmt.setInt(1, orderId);
      rs=pstmt.executeQuery();

      int count=0;
      while(rs.next()){
      queryResult.add(count, new OrderProductInformation(rs.getInt(1),
            rs.getString(2), rs.getInt(3), rs.getFloat(4)));
      }

      pstmt.close();
      con.closeConnection(conn);

         }catch(Exception ex){
   System.err.print(ex);
          }

      countRows=queryResult.size();
      returnValue = new OrderProductInformation[countRows];

      for(int z=0; z<countRows; z++){
      returnValue[z]=queryResult.get(z);
      }

       System.out.println("Get Order Details Ended...");
   return returnValue;
}
   /*
    *
    *
    */
   @WebMethod(operationName = "GetUserInformation")
public UserInformation getUserInformation(@WebParam(name ="userId")
      int userId) {

   UserInformation returnValue = null;

   System.out.println("Get User Details with"
```


Πανεπιστήμιο Πειραιώς

```
            + "UserId: "+userId);


        try{

        conn = con.GetConnection("store01db");

        String query1="SELECT"
              + "* "
              + "FROM "
              + "users "
              + "WHERE "
              + "USER_ID=?";

        pstmt = conn.prepareStatement(query1); // create a statement
        pstmt.setInt(1, userId);
        rs=pstmt.executeQuery();

        while(rs.next()){
        returnValue = new UserInformation(userId, rs.getString("USER_NAME"),
              rs.getString("USER_SURNAME"), rs.getString("USER_EMAIL"),
              rs.getString("USER_PHONE"), rs.getString("USER_MOBILE"),
              rs.getInt("USER_TIN"), rs.getString("USER_TAX_OFFICE"),
              rs.getString("USER_PROFESSION"));
        }

        pstmt.close();
        con.closeConnection(conn);

            }catch(Exception ex){
    System.err.print(ex);
             }

         System.out.println("Get User Information Ended...");
    return returnValue;
}
     /*
     *
     *
     */
    @WebMethod(operationName = "GetAddressInformation")
  public AddressInformation getAddressInformation(@WebParam(name ="addressId")
        int addressId) {

    AddressInformation returnValue = null;
```


Πανεπιστήμιο Πειραιώς

```
    System.out.println("Get Address Information with"
        + "UserId: "+addressId);


    try{

    conn = con.GetConnection("store01db");

    String query1="SELECT"
          + "* "
          + "FROM "
          + "addresses "
          + "WHERE "
          + "ADDRESS_ID=?";

    pstmt = conn.prepareStatement(query1); // create a statement
    pstmt.setInt(1, addressId);
    rs=pstmt.executeQuery();

    while(rs.next()){
    returnValue = new AddressInformation(addressId,
          rs.getString("ADDRESS_STREET"),
          rs.getString("ADDRESS_TOWN"),
          rs.getString("ADDRESS_STATE"),
          rs.getInt("ADDRESS_POSTCODE"),
          rs.getString("ADDRESS_RECIPIENT_NAME"),
          rs.getInt("USER_ID"));
    }

    pstmt.close();
    con.closeConnection(conn);

        }catch(Exception ex){
    System.err.print(ex);
          }

     System.out.println("Get Address Information Ended...");
    return returnValue;
}
    /*
     *
     * Get order information web service
     */
    @WebMethod(operationName = "GetOrderInformation")
   public OrderInformation getOrderInformation(@WebParam(name ="orderId")
      int orderId) {
```


Πανεπιστήμιο Πειραιώς

```
OrderInformation returnValue = null;

System.out.println("Get Order Information with"
    + "OrderId: "+orderId);


  try{

  conn = con.GetConnection("store01db");

  String query1="SELECT "
      + "o.ORDER_ID, "
      + "o.USER_ID, "
      + "o.ADDRESS_ID, "
      + "o.RECEIPT_ID, "
      + "o.ORDER_DATE, "
      + "o.COMMENT, "
      + "o.SHIPPING_TRACKING_ID, "
      + "p.PAYMENT_METHOD_NAME, "
      + "ot.ORDER_STATUS_NAME "
      + "FROM "
      + "orders o "
      + "LEFT OUTER JOIN "
      + "payment_methods p "
      + "ON "
      + "o.PAYMENT_"
      + "METHOD_ID=p.PAYMENT_METHOD_ID "
      + "LEFT OUTER JOIN "
      + "order_status ot "
      + "ON "
      + "o.ORDER_STATUS_ID=ot.ORDER_STATUS_ID "
      + "WHERE ORDER_ID=?";


  pstmt = conn.prepareStatement(query1); // create a statement
  pstmt.setInt(1, orderId);
  rs=pstmt.executeQuery();

  while(rs.next()){
  returnValue = new OrderInformation(orderId,
      rs.getInt("USER_ID"),
      rs.getInt("ADDRESS_ID"),
      rs.getInt("RECEIPT_ID"),
      rs.getTimestamp("ORDER_DATE").toString(),
      rs.getString("COMMENT"),
```


Πανεπιστήμιο Πειραιώς

```
            rs.getString("SHIPPING_TRACKING_ID"),
            rs.getString("PAYMENT_METHOD_NAME"),
            rs.getString("ORDER_STATUS_NAME")
            );
      }

      pstmt.close();
      con.closeConnection(conn);

         }catch(Exception ex){
   System.err.print(ex);
          }

       System.out.println("Get Order Information Ended...");
   return returnValue;
}

        /*
   *
   *
   */
   @WebMethod(operationName = "GetReceiptInformation")
public ReceiptInformation getReceiptInformation(@WebParam(name ="receiptId")
      int receiptId) {

   ReceiptInformation returnValue = null;

   System.out.println("Get Receipt Information with"
        + "UserId: "+receiptId);


      try{

      conn = con.GetConnection("store01db");

      String query1="SELECT "
            + "* "
            + "FROM "
            + "receipt_type r "
            + "LEFT OUTER JOIN "
            + "receipts rs "
            + "ON "
            + "rs.RECEIPT_TYPE_ID=r.RECEIPT_TYPE_ID "
            + "WHERE "
            + "RECEIPT_ID=?";
```


Πανεπιστήμιο Πειραιώς

```
        pstmt = conn.prepareStatement(query1); // create a statement
        pstmt.setInt(1, receiptId);
        rs=pstmt.executeQuery();

        while(rs.next()){
        returnValue = new ReceiptInformation(receiptId,
              rs.getString("RECEIPT_TYPE_NAME"),
              rs.getTimestamp("RECEIPT_DATETIME").toString());
        }

        pstmt.close();
        con.closeConnection(conn);

           }catch(Exception ex){
     System.err.print(ex);
              }

         System.out.println("Get Receipt Information Ended...");
     return returnValue;
}
    /*
     * insertReceipt and OrderInformation
     */
     @WebMethod(operationName = "InsertReceipt")
public InsertReceiptRetrurn InsertReceipt(
        @WebParam(name = "receiptTypeId") int receiptTypeId )

{

int queryStatus1 = 0;
InsertReceiptRetrurn returnValue= new InsertReceiptRetrurn();

returnValue.setStatus(false);

Date today = new Date();
java.sql.Timestamp ts = new java.sql.Timestamp(today.getTime());

System.out.println(""+
        "InsertReceiptReceived:"+
        "\n UserID: "+receiptTypeId
        );

 try{
        conn = con.GetConnection("store01db");

        String query = "INSERT INTO receipts ("
```


Πανεπιστήμιο Πειραιώς

```
            + "RECEIPT_TYPE_ID,"
            + "RECEIPT_DATETIME"
            + ") VALUES (?,?)";

      pstmt = conn.prepareStatement(query, Statement.RETURN_GENERATED_KEYS);
      pstmt.setInt(1, receiptTypeId);
      pstmt.setTimestamp(2, ts);

      queryStatus1= pstmt.executeUpdate();
      rs=pstmt.getGeneratedKeys();

      while(rs.next())
      {
      returnValue.setReceiptId(rs.getInt(1));
      }
      if(queryStatus1==1){
      System.out.println("New row added to Receipts table");
      returnValue.setStatus(true);
      }
      pstmt.close();
     con.closeConnection(conn);

  }catch(Exception ex){
     System.err.print(ex);
  }

       return returnValue;
  }

  /*
   *
   *
   *
   */
      @WebMethod(operationName = "UpdateOrderReceiptId")
  public void UpdateOrderReceiptId(
      @WebParam(name = "orderId") int orderId,
      @WebParam(name = "receiptId") int receiptId)

  {

  System.out.println(""+
      "UpdateOrderReceiptId:"+
      "\n orderid: " +orderId+
      "\n receiptId: "+receiptId
      );
```


Πανεπιστήμιο Πειραιώς

```
 try{

     conn = con.GetConnection("store01db");
         String query1="UPDATE "
             + "orders "
             + "SET "
             + "RECEIPT_ID=? "
             + "WHERE "
             + "ORDER_ID=? ";

     pstmt = conn.prepareStatement(query1); // create a statement
     pstmt.setInt(1, receiptId);
     pstmt.setInt(2, orderId);
     pstmt.executeUpdate();

     pstmt.close();
     con.closeConnection(conn);

        }catch(Exception ex){
   System.err.print(ex);
           }

         System.out.println("update Order With Receipt Id");
     }

/*
 * Update Order  Shipping Id and Status Id
 *
 */
     @WebMethod(operationName = "UpdateOrderShippingAndStatusId")
public void UpdateOrderShippingId(
     @WebParam(name = "orderId") int orderId,
     @WebParam(name = "shippingTrackingId") String shippingTrackingId)

{

System.out.println(""+
     "Update Order Shipping And Status Id:"+
     "\n Order Id: " +orderId+
     "\n Shipping Tracking Id: "+shippingTrackingId
     );

 try{

     conn = con.GetConnection("store01db");
```


Πανεπιστήμιο Πειραιώς

```
            String query1="UPDATE "
                + "orders "
                + "SET "
                + "SHIPPING_TRACKING_ID=?, "
                + "ORDER_STATUS_ID=? "
                + "WHERE "
                + "ORDER_ID=? ";

        pstmt = conn.prepareStatement(query1); // create a statement
        pstmt.setString(1, shippingTrackingId);
        pstmt.setInt(2, 2);
        pstmt.setInt(3, orderId);
        pstmt.executeUpdate();

        pstmt.close();
        con.closeConnection(conn);

            }catch(Exception ex){
      System.err.print(ex);
              }

            System.out.println("Order updated with Shipping Tracking Id"
                + "And Status Id");
        }


}
```

Κώδικας 27: JdbcConnect.java

```
package com.jdbcquery.code;

import java.sql.*;

public class JdbcConnect {

public Connection  GetConnection(String DbName){

   Connection conn = null;

        try
        {
           String userName = "thesisTestDb";
           String password = "12345678";
           String url = "jdbc:mysql://176.34.182.139/"+DbName;
           Class.forName ("com.mysql.jdbc.Driver").newInstance ();
```


Πανεπιστήμιο Πειραιώς

```
            conn = DriverManager.getConnection (url, userName, password);
        }
        catch (Exception e)
        {
            System.err.println (e);
        }
        return conn;
    }

    public void closeConnection(Connection conn){

        if (conn != null)
        {
            try
            {
                conn.close ();
                // System.out.println ("Database connection terminated");
            }
            catch (Exception e) { /* ignore close errors */ }
        }
    }
}
```

6.5 Κώδικας Web service αποστολής e-mail

Κώδικας 28: OrderDetailsEmail.java

```
package com.store.brandname.classes.mail;

public class OrderDetailsEmail {
   private String productName;
   private int productQuantity;
   private float productPrice;

   public OrderDetailsEmail() {
   }

   public OrderDetailsEmail(String productName, int productQuantity, float
productPrice) {
      this.productName = productName;
      this.productQuantity = productQuantity;
      this.productPrice = productPrice;
   }
```

Πανεπιστήμιο Πειραιώς

```
    public String getProductName() {
        return productName;
    }

    public void setProductName(String productName) {
        this.productName = productName;
    }

    public float getProductPrice() {
        return productPrice;
    }

    public void setProductPrice(float productPrice) {
        this.productPrice = productPrice;
    }

    public int getProductQuantity() {
        return productQuantity;
    }

    public void setProductQuantity(int productQuantity) {
        this.productQuantity = productQuantity;
    }

    public void getProductName(String string) {
        throw new UnsupportedOperationException("Not yet implemented");
    }


}
```

Κώδικας 29: OrderDetailsEmailResponse.java

```
package com.store.brandname.classes.mail;

public class OrderDetailsEmailResponse {
    private String email;
    private String content;

    public OrderDetailsEmailResponse() {
    }

    public OrderDetailsEmailResponse(String email, String content) {
        this.email = email;
```


Πανεπιστήμιο Πειραιώς

```
        this.content = content;
    }

    public String getContent() {
        return content;
    }

    public void setContent(String content) {
        this.content = content;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }


}
```

Κώδικας 30: OrderEmailGeneratorWS.java

```
package com.store.brandname.webservice;

import com.jdbcquery.code.JdbcConnect;
import com.store.brandname.classes.mail.OrderDetailsEmail;
import com.store.brandname.classes.mail.OrderDetailsEmailResponse;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.HashMap;


import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

@WebService(serviceName = "OrderEmailGeneratorWS")
public class OrderEmailGeneratorWS {
JdbcConnect con = new JdbcConnect();
```

Πανεπιστήμιο Πειραιώς

```
/**
 * This is a sample web service operation
 */
@WebMethod(operationName = "CreateOrderDetails")
public OrderDetailsEmailResponse OrderDetailsContent(
      @WebParam(name = "OrderNumber") int orderId,
      @WebParam(name = "UserID") int userId,
      @WebParam(name = "MailOperation") String operation)
{

Connection conn;
PreparedStatement pstmt;
ResultSet rs;
OrderDetailsEmailResponse response = new OrderDetailsEmailResponse();

if(operation.equals("Order Details")){
   System.out.println("Order Details MAIL");
ArrayList<OrderDetailsEmail> mailContent =
      new ArrayList<OrderDetailsEmail>();

String htmlCode, htmlCode1, htmlCode2, htmlCodeFinal;
float price=0;
int i=0;

try{
      conn = con.GetConnection("store01db");


      String query = "SELECT "
            + "PRODUCT_NAME, "
            + "QUANTITY, "
            + "PRODUCT_PRICE "
            + "FROM "
            + "products p "
            + "LEFT OUTER JOIN "
            + "products_to_order pt "
            + "ON "
            + "pt.PRODUCT_ID=p.PRODUCT_ID "
            + "WHERE "
            + "ORDER_ID=? ";


      pstmt = conn.prepareStatement(query); // create a statement
      pstmt.setInt(1, orderId); // set input parameter 1
      rs=pstmt.executeQuery();
```


Πανεπιστήμιο Πειραιώς

```
        while(rs.next())
        {
        mailContent.add(i, new OrderDetailsEmail(rs.getString(1),
              rs.getInt(2), rs.getFloat(3)));
        i++;
        }



        query="SELECT "
              + "USER_EMAIL "
              + "FROM "
              + "users "
              + "WHERE "
              + "USER_ID=?";

        pstmt = conn.prepareStatement(query); // create a statement
        pstmt.setInt(1, userId); // set input parameter 1
        rs=pstmt.executeQuery();

        while(rs.next()){
         response.setEmail(rs.getString(1));
        }

        rs.close();
        pstmt.close();
        con.closeConnection(conn);

    }catch(Exception ex){
       System.err.print(ex);
    }

    htmlCode="<html>\n"
        + "<body>\n"
        + "\n"
        + "<h3>Your Order number is: "+orderId+"</h3>\n"
        + "\n"
        + "<p>\n"
        + "<table border='0' width='600'>\n"
        + "<tr>\n"
        + "<th align='left'>Product</th>\n"
        + "<th align='left'>Quantity</th>\n"
        + "<th align='left'>Price</th>\n"
        + "</tr>\n"
        + "<tr>\n";
```


Πανεπιστήμιο Πειραιώς

```
    for(int k=0; k<i; k++){

        htmlCode1= "<tr>\n"
              + "<td align='left'
width='80%'>"+mailContent.get(k).getProductName()+"</td>\n"
              + "<td align='center'>"+mailContent.get(k).getProductQuantity()+ "</td>\n"
              + "<td
align='left'>"+mailContent.get(k).getProductPrice()*mailContent.get(k).getProductQuan
tity()+"</td>\n"
              + "</tr>\n";

price=price+(mailContent.get(k).getProductPrice()*mailContent.get(k).getProductQuant
ity());
   htmlCode=htmlCode.concat(htmlCode1);
             }

htmlCode2=""
      +"<tr>\n"
      + "<td></td>\n"
      + "<td></td>\n"
      + "<td></td>\n"
      + "</tr>\n"
      + "<tr>\n"
      + "<td></td>\n"
      + "<td><b>Total:</b></td>\n"
      + "<td><b>"+price+"</b></td>\n"
      + "</tr>\n"
      + "</table>\n"
      + "</p>\n"
      + "</body>\n"
      + "</html>\n";

 htmlCode=htmlCode.concat(htmlCode2);
   response.setContent(htmlCode);
   }

    if(operation.equals("Bank Payment Verification True")){
     System.out.println("Bank Payment Verification True MAIL");
       try{
       conn = con.GetConnection("store01db");


         String query="SELECT "
               + "USER_EMAIL "
               + "FROM "
               + "users "
```


Πανεπιστήμιο Πειραιώς

```
            + "WHERE "
            + "USER_ID=?";

        pstmt = conn.prepareStatement(query); // create a statement
        pstmt.setInt(1, userId); // set input parameter 1
        rs=pstmt.executeQuery();

        while(rs.next()){
         response.setEmail(rs.getString(1));
        }

        rs.close();
        pstmt.close();
        con.closeConnection(conn);

      }catch(Exception ex){
      System.err.print(ex);
   }

       response.setContent(""
         + "<html>\n"
         + "<body>\n"
         + "\n"
         + "<h3>Your bank payment for order: "+orderId+" received</h3>\n"
         + "\n"
         + "<p>We continue your order processing.</p>\n"
         + "</body>\n"
         + "</html>\n");
    }
      return response;
   }
}
```

Κώδικας 31: JdbcConnect.java

```
package com.jdbcquery.code;

import java.sql.*;

public class JdbcConnect {

public Connection  GetConnection(String DbName){

   Connection conn = null;
```


Πανεπιστήμιο Πειραιώς

```
        try
        {
           String userName = "thesisTestDb";
           String password = "12345678";
           String url = "jdbc:mysql://176.34.182.139/"+DbName;
           Class.forName ("com.mysql.jdbc.Driver").newInstance ();
           conn = DriverManager.getConnection (url, userName, password);
        }
        catch (Exception e)
        {
           System.err.println (e);
        }
        return conn;
    }

    public void closeConnection(Connection conn){

        if (conn != null)
        {
           try
           {
              conn.close ();
             // System.out.println ("Database connection terminated");
           }
           catch (Exception e) { /* ignore close errors */ }
        }
    }
}
```

6.6 Κώδικας Web service δημιουργίας παραστατικού

Κώδικας 32: CustomerInformation.java

```
package com.store.brandname.classes.finance;

public class CustomerInformation {
   private int userId;
   private String streedAddress;
   private String town;
   private int postcode;
   private int tin;
```


Πανεπιστήμιο Πειραιώς

```
  private String profession;
  private String userName;
  private String taxOffice;
  private String phone;

  public CustomerInformation() {
  }

  public CustomerInformation(int userId, String streedAddress, String town,
      int postcode, int tin, String profession, String userName,
      String taxOffice, String phone) {
    this.userId = userId;
    this.streedAddress = streedAddress;
    this.town = town;
    this.postcode = postcode;
    this.tin = tin;
    this.profession = profession;
    this.userName = userName;
    this.taxOffice = taxOffice;
    this.phone = phone;
  }

  public String getPhone() {
    return phone;
  }

  public void setPhone(String phone) {
    this.phone = phone;
  }

  public int getPostcode() {
    return postcode;
  }

  public void setPostcode(int postcode) {
    this.postcode = postcode;
  }

  public String getProfession() {
    return profession;
  }

  public void setProfession(String profession) {
    this.profession = profession;
  }
```


Πανεπιστήμιο Πειραιώς

```
public String getStreedAddress() {
   return streedAddress;
}

public void setStreedAddress(String streedAddress) {
   this.streedAddress = streedAddress;
}

public String getTaxOffice() {
   return taxOffice;
}

public void setTaxOffice(String taxOffice) {
   this.taxOffice = taxOffice;
}

public int getTin() {
   return tin;
}

public void setTin(int tin) {
   this.tin = tin;
}

public String getTown() {
   return town;
}

public void setTown(String town) {
   this.town = town;
}

public int getUserId() {
   return userId;
}

public void setUserId(int userId) {
   this.userId = userId;
}

public String getUserName() {
   return userName;
}

public void setUserName(String userName) {
   this.userName = userName;
```

Πανεπιστήμιο Πειραιώς

```
  }



}
```

Κώδικας 33: DocumentInformation.java

```
package com.store.brandname.classes.finance;

public class DocumentInformation {
  private String documentType;
  private int documentNumber;
  private String date;
  private String PaymentMethod;
  private String time;

   public DocumentInformation() {
   }

   public DocumentInformation(String documentType, int documentNumber, String
date, String PaymentMethod, String time) {
      this.documentType = documentType;
      this.documentNumber = documentNumber;
      this.date = date;
      this.PaymentMethod = PaymentMethod;
      this.time = time;
   }

   public String getPaymentMethod() {
      return PaymentMethod;
   }

   public void setPaymentMethod(String PaymentMethod) {
      this.PaymentMethod = PaymentMethod;
   }

   public String getDate() {
      return date;
   }

   public void setDate(String date) {
      this.date = date;
   }
```


Πανεπιστήμιο Πειραιώς

```
  public int getDocumentNumber() {
    return documentNumber;
  }

  public void setDocumentNumber(int documentNumber) {
    this.documentNumber = documentNumber;
  }

  public String getDocumentType() {
    return documentType;
  }

  public void setDocumentType(String documentType) {
    this.documentType = documentType;
  }

  public String getTime() {
    return time;
  }

  public void setTime(String time) {
    this.time = time;
  }


}
```

Κώδικας 34: InvoiceProducts.java

```
package com.store.brandname.classes.finance;

public class InvoiceProducts {
  private int productCode;
  private String productName;
  private float unitPrice;
  private int productQuantity;

  public InvoiceProducts() {
  }

  public int getProductCode() {
    return productCode;
  }
```


Πανεπιστήμιο Πειραιώς

```
  public InvoiceProducts(int productCode, String productName, float unitPrice, int
productQuantity) {
    this.productCode = productCode;
    this.productName = productName;
    this.unitPrice = unitPrice;
    this.productQuantity = productQuantity;
  }



  public void setProductCode(int productCode) {
    this.productCode = productCode;
  }

  public String getProductName() {
    return productName;
  }

  public void setProductName(String productName) {
    this.productName = productName;
  }

  public int getProductQuantity() {
    return productQuantity;
  }

  public void setProductQuantity(int productQuantity) {
    this.productQuantity = productQuantity;
  }

  public float getUnitPrice() {
    return unitPrice;
  }

  public void setUnitPrice(float unitPrice) {
    this.unitPrice = unitPrice;
  }


}
```

Κώδικας 35: InvoiceRequest.java

Πανεπιστήμιο Πειραιώς

```
package com.store.brandname.classes.finance;

public class InvoiceRequest {
   private CustomerInformation customerInformation;
   private DocumentInformation documentInformation;
   private InvoiceProducts[] products;
   private String Comments;
   private String recipientsName;
   private int vat;
   private int orderId;

   public InvoiceRequest() {
   }

   public InvoiceRequest(CustomerInformation customerInformation,
DocumentInformation documentInformation, InvoiceProducts[] products, String
Comments, String recipientsName, int vat, int orderId) {
      this.customerInformation = customerInformation;
      this.documentInformation = documentInformation;
      this.products = products;
      this.Comments = Comments;
      this.recipientsName = recipientsName;
      this.vat = vat;
      this.orderId = orderId;
   }

   public String getComments() {
      return Comments;
   }

   public void setComments(String Comments) {
      this.Comments = Comments;
   }

   public CustomerInformation getCustomerInformation() {
      return customerInformation;
   }

   public void setCustomerInformation(CustomerInformation customerInformation) {
      this.customerInformation = customerInformation;
   }

   public DocumentInformation getDocumentInformation() {
      return documentInformation;
   }
```

Πανεπιστήμιο Πειραιώς

```
    public void setDocumentInformation(DocumentInformation documentInformation) {
        this.documentInformation = documentInformation;
    }

    public int getOrderId() {
        return orderId;
    }

    public void setOrderId(int orderId) {
        this.orderId = orderId;
    }

    public InvoiceProducts[] getProducts() {
        return products;
    }

    public void setProducts(InvoiceProducts[] products) {
        this.products = products;
    }

    public String getRecipientsName() {
        return recipientsName;
    }

    public void setRecipientsName(String recipientsName) {
        this.recipientsName = recipientsName;
    }

    public int getVat() {
        return vat;
    }

    public void setVat(int vat) {
        this.vat = vat;
    }


}
```

Κώδικας 36: InvoicePDFDocumentWS.java

```
package com.store.brandname.webservice;

import com.itextpdf.text.*;
import com.itextpdf.text.pdf.PdfPCell;
```


Πανεπιστήμιο Πειραιώς

```
import com.itextpdf.text.pdf.PdfPTable;
import com.itextpdf.text.pdf.PdfWriter;
import com.store.brandname.classes.finance.InvoiceProducts;
import com.store.brandname.classes.finance.InvoiceRequest;
import java.io.FileOutputStream;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

@WebService(serviceName = "InvoicePDFDocumentWS")
public class InvoicePDFDocumentWS {
private static Font headerFont =
      new Font(Font.FontFamily.HELVETICA, 10, Font.BOLD);
private static Font CompanyFont =
      new Font(Font.FontFamily.HELVETICA, 12, Font.BOLD);
private static Font subFont =
      new Font(Font.FontFamily.HELVETICA, 8,Font.NORMAL);
private static Font subFontBold =
      new Font(Font.FontFamily.HELVETICA, 8,Font.BOLD);
  /**
   * This is a sample web service operation
   */
  @WebMethod(operationName = "produceReciptPDF")
  public String produceReciptPDF(
        @WebParam(name = "receiptInput") InvoiceRequest invoice) {
  try {



        int tquantity=0;
        float tprice=0, totalPriceWithVat=0,  vat=0;
        String commentString="";
    Document document=new Document(PageSize.A4, 20,20,20,20);
    PdfWriter writer = PdfWriter.getInstance(document,
        new FileOutputStream("C:\\pdfs\\"+invoice.getOrderId()+".pdf"));
    document.open();

    Paragraph companyInfo = new Paragraph();
    companyInfo.add(new Chunk("Company Name\n", CompanyFont));
    companyInfo.add(new Chunk("Street Address\n", subFont));
    companyInfo.add(new Chunk("City, State, Postcode\n", subFont));
    companyInfo.add(new Chunk("Phone, Fax, etc\n", subFont));
    companyInfo.add(new Chunk("Website\n\n", subFont));
    document.add(companyInfo);
```


Πανεπιστήμιο Πειραιώς

```
/*
 * Customer Information
 */
Paragraph customerInfo = new Paragraph();
customerInfo.add(new Chunk("Customer Information\n", headerFont));
document.add(customerInfo);

PdfPTable  ctM = new PdfPTable(1);
ctM.setSpacingBefore(4);
ctM.setSpacingAfter(12);
ctM.setWidthPercentage(100);
PdfPTable  ct = new PdfPTable(8);

ct.setWidthPercentage(100);
ct.setWidths(new float[]{0.6f,1.3f,0.4f,1f,0.5f,1f,0.6f,1f});

PdfPCell cellct11 = new PdfPCell(new Phrase("Client Code", subFontBold));
PdfPCell cellct12 = new PdfPCell(new Phrase(": "+
    invoice.getCustomerInformation().getUserId(), subFont));
PdfPCell cellct13 = new PdfPCell(new Phrase("Street", subFontBold));
PdfPCell cellct14 = new PdfPCell(new Phrase(": "+
    invoice.getCustomerInformation().getStreedAddress(), subFont));
PdfPCell cellct15 = new PdfPCell(new Phrase("TIN", subFontBold));
PdfPCell cellct16 = new PdfPCell(new Phrase(": "
    +invoice.getCustomerInformation().getTin(), subFont));
PdfPCell cellct17 = new PdfPCell(new Phrase("Profession", subFontBold));
PdfPCell cellct18 = new PdfPCell(new Phrase(": "
    +invoice.getCustomerInformation().getProfession(), subFont));
PdfPCell cellct21 = new PdfPCell(new Phrase("Name", subFontBold));
PdfPCell cellct22 = new PdfPCell(new Phrase(": "
    +invoice.getCustomerInformation().getUserName(), subFont));
PdfPCell cellct23 = new PdfPCell(new Phrase("Town", subFontBold));
PdfPCell cellct24 = new PdfPCell(new Phrase(": "
    +invoice.getCustomerInformation().getPostcode()+", "
    +invoice.getCustomerInformation().getTown(), subFont));
PdfPCell cellct25 = new PdfPCell(new Phrase("Tax Office", subFontBold));
PdfPCell cellct26 = new PdfPCell(new Phrase(": "
    +invoice.getCustomerInformation().getTaxOffice(), subFont));
PdfPCell cellct27 = new PdfPCell(new Phrase("Phone", subFontBold));
PdfPCell cellct28 = new PdfPCell(new Phrase(": "
    +invoice.getCustomerInformation().getPhone(), subFont));

cellct11.setBorder(Rectangle.NO_BORDER);
cellct12.setBorder(Rectangle.NO_BORDER);
cellct13.setBorder(Rectangle.NO_BORDER);
```


Πανεπιστήμιο Πειραιώς

```
cellct14.setBorder(Rectangle.NO_BORDER);
cellct15.setBorder(Rectangle.NO_BORDER);
cellct16.setBorder(Rectangle.NO_BORDER);
cellct17.setBorder(Rectangle.NO_BORDER);
cellct18.setBorder(Rectangle.NO_BORDER);
cellct21.setBorder(Rectangle.NO_BORDER);
cellct22.setBorder(Rectangle.NO_BORDER);
cellct23.setBorder(Rectangle.NO_BORDER);
cellct24.setBorder(Rectangle.NO_BORDER);
cellct25.setBorder(Rectangle.NO_BORDER);
cellct26.setBorder(Rectangle.NO_BORDER);
cellct27.setBorder(Rectangle.NO_BORDER);
cellct28.setBorder(Rectangle.NO_BORDER);

ct.addCell(cellct11);
ct.addCell(cellct12);
ct.addCell(cellct13);
ct.addCell(cellct14);
ct.addCell(cellct15);
ct.addCell(cellct16);
ct.addCell(cellct17);
ct.addCell(cellct18);
ct.addCell(cellct21);
ct.addCell(cellct22);
ct.addCell(cellct23);
ct.addCell(cellct24);
ct.addCell(cellct25);
ct.addCell(cellct26);
ct.addCell(cellct27);
ct.addCell(cellct28);

ctM.addCell(new PdfPCell(ct));
document.add(ctM);

/*
 * DOCUMENT INFORMATION
 */

Paragraph documentInformation = new Paragraph();
documentInformation.add(new Chunk("Document Information\n", headerFont));
document.add(documentInformation);

PdfPTable  t1 = new PdfPTable(5);
t1.setSpacingBefore(4);
t1.setWidthPercentage(100);
t1.setWidths(new float[]{1f,1.4f,1f,1f,1f});
```


Πανεπιστήμιο Πειραιώς

```
PdfPCell cell11 = new PdfPCell(new Phrase("Document Type", subFontBold));
PdfPCell cell12 = new PdfPCell(new Phrase("Document Number ",
    subFontBold));
PdfPCell cell13 = new PdfPCell(new Phrase("Date", subFontBold));
PdfPCell cell14 = new PdfPCell(new Phrase("Payment Method", subFontBold));
PdfPCell cell15 = new PdfPCell(new Phrase("Time", subFontBold));


t1.addCell(cell11);
t1.addCell(cell12);
t1.addCell(cell13);
t1.addCell(cell14);
t1.addCell(cell15);

t1.addCell(new Phrase(invoice.getDocumentInformation().getDocumentType(),
    subFont));
t1.addCell(new Phrase(""
    +invoice.getDocumentInformation().getDocumentNumber(), subFont));
t1.addCell(new Phrase(invoice.getDocumentInformation().getDate(),
    subFont));
t1.addCell(new Phrase(invoice.getDocumentInformation().getPaymentMethod(),
    subFont));
t1.addCell(new Phrase(
  invoice.getDocumentInformation().getTime().subSequence(0, 8).toString(),
    subFont));


document.add(t1);
document.add(Chunk.NEWLINE);
 /*
 * PRODUCT-ORDER INFORMATION
 */

Paragraph orderInformation = new Paragraph();
orderInformation.add(new Chunk("Order Information\n", headerFont));
document.add(orderInformation);


PdfPTable  t2 = new PdfPTable(6);
t2.setSpacingBefore(4);
t2.setWidthPercentage(100);
t2.setWidths(new float[]{0.58f,3f,0.6f,0.5f,0.5f,0.4f});

PdfPCell cell21 = new PdfPCell(new Phrase("Product Code", subFontBold));
PdfPCell cell22 = new PdfPCell(new Phrase("Product Name", subFontBold));
```

Πανεπιστήμιο Πειραιώς

```
PdfPCell cell23 = new PdfPCell(new Phrase("Unit Price", subFontBold));
PdfPCell cell24 = new PdfPCell(new Phrase("Quantity", subFontBold));
PdfPCell cell25 = new PdfPCell(new Phrase("Price", subFontBold));
PdfPCell cell26 = new PdfPCell(new Phrase("Vat", subFontBold));

cell23.setHorizontalAlignment(Element.ALIGN_CENTER);
cell24.setHorizontalAlignment(Element.ALIGN_CENTER);
cell25.setHorizontalAlignment(Element.ALIGN_CENTER);
cell26.setHorizontalAlignment(Element.ALIGN_CENTER);

t2.addCell(cell21);
t2.addCell(cell22);
t2.addCell(cell23);
t2.addCell(cell24);
t2.addCell(cell25);
t2.addCell(cell26);

int arraySize=invoice.getProducts().length;
InvoiceProducts[] prod = new InvoiceProducts[arraySize];

prod=invoice.getProducts();
PdfPCell[][] pcell = new PdfPCell[arraySize][6];

for(int c=0; c<arraySize; c++){

tquantity=tquantity+prod[c].getProductQuantity();
tprice=tprice+(prod[c].getProductQuantity()*
    prod[c].getUnitPrice());


pcell[c][0]= new PdfPCell(new Phrase(""+prod[c].getProductCode(),
    subFont));
pcell[c][1]= new PdfPCell(new Phrase(prod[c].getProductName(), subFont));
pcell[c][2]= new PdfPCell(new Phrase(""+prod[c].getUnitPrice(), subFont));
pcell[c][3]= new PdfPCell(new Phrase(""+prod[c].getProductQuantity(),
    subFont));
pcell[c][4]= new PdfPCell(new Phrase(""+prod[c].getProductQuantity()*
    prod[c].getUnitPrice(), subFont));
pcell[c][5]= new PdfPCell(new Phrase(""+vat+"%", subFont));

pcell[c][0].setHorizontalAlignment(Element.ALIGN_RIGHT);
pcell[c][2].setHorizontalAlignment(Element.ALIGN_RIGHT);
pcell[c][3].setHorizontalAlignment(Element.ALIGN_RIGHT);
pcell[c][4].setHorizontalAlignment(Element.ALIGN_RIGHT);
pcell[c][5].setHorizontalAlignment(Element.ALIGN_CENTER);
```


Πανεπιστήμιο Πειραιώς

```
    t2.addCell(pcell[c][0]);
    t2.addCell(pcell[c][1]);
    t2.addCell(pcell[c][2]);
    t2.addCell(pcell[c][3]);
    t2.addCell(pcell[c][4]);
    t2.addCell(pcell[c][5]);
    }

    document.add(t2);

    /*
     *
     *
     *
     */

if(invoice.getCustomerInformation().getUserName().equals(invoice.getRecipientsName(
))){
     commentString=" ";
      }else{
        if(invoice.getRecipientsName().equals("null")){}
        else{
      commentString="Recipient's name: "+invoice.getRecipientsName();
        }
         }

System.out.println(invoice.getCustomerInformation().getUserName()+":"+invoice.getRe
cipientsName());
    PdfPTable comments = new PdfPTable(1);
    comments.getDefaultCell().setBorder(Rectangle.LEFT | Rectangle.RIGHT);
    PdfPCell cellc1 = new PdfPCell(new Phrase("Comments:",subFontBold));
    PdfPCell cellc2 = new PdfPCell(new Phrase(" "+invoice.getComments(),subFont));
    PdfPCell cellc3 = new PdfPCell(new Phrase(" "+commentString,subFont));
    PdfPCell cellc4 = new PdfPCell(new Phrase(" ",subFont));

    cellc1.setBorder(Rectangle.NO_BORDER);
    cellc2.setBorder(Rectangle.LEFT | Rectangle.RIGHT | Rectangle.TOP);
    cellc3.setBorder(Rectangle.LEFT | Rectangle.RIGHT);
    cellc4.setBorder(Rectangle.LEFT | Rectangle.RIGHT | Rectangle.BOTTOM);
    comments.addCell(cellc1);
    comments.addCell(cellc2);
    comments.addCell(cellc3);
    comments.addCell(cellc4);
    comments.setTotalWidth(400f);
    comments.writeSelectedRows(0, -1, 20, 100, writer.getDirectContent());
```

Πανεπιστήμιο Πειραιώς

```
    PdfPTable footer = new PdfPTable(2);

    vat=tprice*(invoice.getVat()/100);
    totalPriceWithVat=vat+tprice;
    PdfPCell cellf11 = new PdfPCell(new Phrase("Total Quantity:", subFontBold));
    PdfPCell cellf12 = new PdfPCell(new Phrase(""+tquantity, subFontBold));
    PdfPCell cellf13 = new PdfPCell(new Phrase("Subtotal:", subFontBold));
    PdfPCell cellf14 = new PdfPCell(new Phrase(""+tprice, subFontBold));
    PdfPCell cellf15 = new PdfPCell(new Phrase("VAT:", subFontBold));
    PdfPCell cellf16 = new PdfPCell(new Phrase(""+vat, subFontBold));
    PdfPCell cellf17 = new PdfPCell(new Phrase("Total:", subFontBold));
    PdfPCell cellf18 = new PdfPCell(new Phrase(""+totalPriceWithVat,
          subFontBold));

    //cellf11.setBorder(Rectangle.NO_BORDER);
    //cellf12.setBorder(Rectangle.TOP | Rectangle.LEFT | Rectangle.BOTTOM);
    footer.addCell(cellf11);
    footer.addCell(cellf12);
    footer.addCell(cellf13);
    footer.addCell(cellf14);
    footer.addCell(cellf15);
    footer.addCell(cellf16);
    footer.addCell(cellf17);
    footer.addCell(cellf18);



    footer.setTotalWidth(130f);
    footer.setWidths(new float[]{1.2f,0.8f});
    footer.writeSelectedRows(0, -1, 440, 100, writer.getDirectContent());


    /*
     * Order ID table
     */


    PdfPTable orderId = new PdfPTable(1);
    orderId.getDefaultCell().setBorder(Rectangle.NO_BORDER);
    PdfPCell orderIdCell = new PdfPCell(new Phrase("Order
Code:"+invoice.getOrderId(),
          subFont));
    orderIdCell.setBorder(Rectangle.NO_BORDER);
    orderIdCell.setHorizontalAlignment(Element.ALIGN_RIGHT);
    orderId.addCell(orderIdCell);
    orderId.setTotalWidth(100f);
```


Πανεπιστήμιο Πειραιώς

```
    orderId.writeSelectedRows(0, -1, 470, 830, writer.getDirectContent());

    document.close();
                } catch (Exception e) {
                }
  return null;
  }
}
```

6.7 Κώδικας κεντρικής διαδικασίας BPEL παραγγελίας προϊόντων

Κώδικας 37: OrderProcess.bpel

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<!--
////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
 Oracle JDeveloper BPEL Designer

 Created: Sun Feb 12 22:27:45 EET 2012
 Author:  Epameinondas Nikolopoulos
 Type: BPEL 2.0 Process
 Purpose: Synchronous BPEL Process
////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
-->
<process name="OrderProcess"

targetNamespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessProject1/OrderProcess"
         xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"

xmlns:client="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessProject1/OrderProcess"
         xmlns:ora="http://schemas.oracle.com/xpath/extension"
         xmlns:bpelx="http://schemas.oracle.com/bpel/extension"
      xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"

xmlns:ns1="http://xmlns.oracle.com/BankPaymentV02/Project1/BankVerificationProcess"

xmlns:ns2="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessProject1/OrderProcess/correlationset"
```


Πανεπιστήμιο Πειραιώς

```
xmlns:ns3="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessProj
ect1/BuyingProcessBpelV01"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20"
      xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
      xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
      xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
      xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
      xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions"
      xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
      xmlns:ns4="http://xmlns.oracle.com/ias/pcbpel/NotificationService"
      xmlns:ns5="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:ns6="http://webservice.brandname.store.com/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:ns7="http://webservice.gateway.creditcard.com/"
      xmlns:ns8="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway"

xmlns:ns9="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAndS
uppliers/ProcessOrderAndOrderFromSuppliers"
      xmlns:ns10="http://www.example.org"

xmlns:ns11="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAn
dSuppliers/ProcessOrderAndSupplier"

xmlns:ns12="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHuman
Task/OrderFulfillment"
      xmlns:ns13="http://services.web.shippingcompany.com/">

   <import
namespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessPr
oject1/OrderProcess" location="OrderProcess.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
   <!--

 ////////////////////////////////////////////////////////////////////
```


Πανεπιστήμιο Πειραιώς

```
////////////////////////////////////////////////////////////////////////////
     PARTNERLINKS
     List of services participating in this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
  -->
 <partnerLinks>
  <!--
   The 'client' role represents the requester of this service. It is
   used for callback. The location and correlation information associated
   with the client role are automatically set using WS-Addressing.
  -->
  <partnerLink name="orderprocess_client" partnerLinkType="client:OrderProcess"
myRole="OrderProcessProvider"/>
  <partnerLink name="NotificationService1"
          partnerLinkType="ns4:NotificationServiceLink"
          partnerRole="NotificationServiceProvider"/>
  <partnerLink name="Repository" partnerLinkType="ns6:Repository"
          partnerRole="RepositoryWS"/>
  <partnerLink name="CreditCardGateway"
          partnerLinkType="ns8:CreditCardGateway"
          partnerRole="CreditCardGatewayProvider"/>
  <partnerLink name="OrderMailContentGenerator"
          partnerLinkType="ns6:OrderMailContentGenerator"
          partnerRole="OrderEmailGeneratorWS"/>
  <partnerLink name="NotificationService2"
          partnerLinkType="ns4:NotificationServiceLink"
          partnerRole="NotificationServiceProvider"/>
  <partnerLink name="BankVerification"
          partnerLinkType="ns1:BankVerificationProcess"
          partnerRole="BankVerificationProcessProvider"
          myRole="BankVerificationProcessRequester"/>
  <partnerLink name="ProcessOrderAndSupply"
          partnerLinkType="ns11:ProcessOrderAndSupplier"
          partnerRole="ProcessOrderAndSupplierProvider"/>
  <partnerLink name="OrderFulfillmentHT"
          partnerLinkType="ns12:OrderFulfillment"
          partnerRole="OrderFulfillmentProvider"
          myRole="OrderFulfillmentRequester"/>
  <partnerLink name="Repository1" partnerLinkType="ns6:Repository1"
          partnerRole="Repository1WS"/>
  <partnerLink name="InvoiceGenerator" partnerLinkType="ns6:InvoiceGenerator"
          partnerRole="InvoicePDFDocumentWS"/>
  <partnerLink name="ShippingCompany" partnerLinkType="ns13:ShippingCompany"
          partnerRole="ShippingCompanyWS"/>
```


Πανεπιστήμιο Πειραιώς

```
  <partnerLink name="NotificationService4"
        partnerLinkType="ns4:NotificationServiceLink"
        partnerRole="NotificationServiceProvider"/>
 </partnerLinks>

 <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
   VARIABLES
   List of messages and XML documents used within this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
 -->
 <variables>
  <!-- Reference to the message passed as input during initiation -->
  <variable name="inputVariable"
messageType="client:OrderProcessRequestMessage"/>

  <!-- Reference to the message that will be returned to the requester-->
  <variable name="outputVariable"
messageType="client:OrderProcessResponseMessage"/>
  <variable name="Invoke1_process_InputVariable"
        messageType="ns1:BankVerificationProcessRequestMessage"/>
  <variable name="Receive1_processResponse_InputVariable"
        messageType="ns1:BankVerificationProcessResponseMessage"/>
  <variable name="Invoke2_InsertOrder_InputVariable"
        messageType="ns6:InsertOrder"/>
  <variable name="Invoke2_InsertOrder_OutputVariable"
        messageType="ns6:InsertOrderResponse"/>
  <variable name="Invoke3_InsertProductToOrder_InputVariable"
        messageType="ns6:InsertProductToOrder"/>
  <variable name="Invoke3_InsertProductToOrder_OutputVariable"
        messageType="ns6:InsertProductToOrderResponse"/>
  <variable name="InvokeCCG_CreditCardProcessing_InputVariable"
        messageType="ns8:CreditCardGatewayRequestMessage"/>
  <variable name="InvokeCCG_CreditCardProcessing_OutputVariable"
        messageType="ns8:CreditCardGatewayResponseMessage"/>
  <variable name="CCPaymentStatusBoolean" type="xsd:boolean"/>
  <variable name="InvokeOMCG_CreateOrderDetails_InputVariable"
        messageType="ns6:CreateOrderDetails"/>
  <variable name="InvokeOMCG_CreateOrderDetails_OutputVariable"
        messageType="ns6:CreateOrderDetailsResponse"/>
  <variable name="OrderID" type="xsd:unsignedInt"/>
  <variable name="UserId" type="xsd:unsignedInt"/>
```


Πανεπιστήμιο Πειραιώς

```
<variable name="Invoke5_processOrderAndSuppliers_InputVariable"
      messageType="ns11:ProcessOrderAndSupplierRequestMessage"/>
<variable name="Invoke5_processOrderAndSuppliers_OutputVariable"
      messageType="ns11:ProcessOrderAndSupplierResponseMessage"/>
<variable name="Invoke6_CheckOrderStatus_InputVariable"
      messageType="ns6:CheckOrderStatus"/>
<variable name="Invoke6_CheckOrderStatus_OutputVariable"
      messageType="ns6:CheckOrderStatusResponse"/>
<variable name="ProductsCount" type="xsd:int"/>
<variable name="Invoke7_OrderFulfillment_InputVariable"
      messageType="ns12:OrderFulfillmentRequestMessage"/>
<variable name="Invoke8_GetOrderProductDetails_InputVariable"
      messageType="ns6:GetOrderProductDetails"/>
<variable name="Invoke8_GetOrderProductDetails_OutputVariable"
      messageType="ns6:GetOrderProductDetailsResponse"/>
<variable name="Inkoke7_ReceiveOrderFulfillment_OutputVariable"
      messageType="ns12:OrderFulfillmentResponseMessage"/>
<variable name="Invoke9_GetUserInformation_InputVariable"
      messageType="ns6:GetUserInformation"/>
<variable name="Invoke9_GetUserInformation_OutputVariable"
      messageType="ns6:GetUserInformationResponse"/>
<variable name="Invoke11_GetAddressInformation_InputVariable"
      messageType="ns6:GetAddressInformation"/>
<variable name="Invoke11_GetAddressInformation_OutputVariable"
      messageType="ns6:GetAddressInformationResponse"/>
<variable name="Invoke12_GetOrderInformation_InputVariable"
      messageType="ns6:GetOrderInformation"/>
<variable name="Invoke12_GetOrderInformation_OutputVariable"
      messageType="ns6:GetOrderInformationResponse"/>
<variable name="Invoke13_InsertReceipt_InputVariable"
      messageType="ns6:InsertReceipt"/>
<variable name="Invoke13_InsertReceipt_OutputVariable"
      messageType="ns6:InsertReceiptResponse"/>
<variable name="Invoke14_UpdateOrderReceiptId_OutputVariable"
      messageType="ns6:UpdateOrderReceiptIdResponse"/>
<variable name="Invoke14_UpdateOrderReceiptId_InputVariable"
      messageType="ns6:UpdateOrderReceiptId"/>
<variable name="Invoke10_GetReceiptInformation_InputVariable"
      messageType="ns6:GetReceiptInformation"/>
<variable name="Invoke10_GetReceiptInformation_OutputVariable"
      messageType="ns6:GetReceiptInformationResponse"/>
<variable name="Invoke15_produceReciptPDF_InputVariable"
      messageType="ns6:produceReciptPDF"/>
<variable name="Invoke15_produceReciptPDF_OutputVariable"
      messageType="ns6:produceReciptPDFResponse"/>
<variable name="Invoke17_UpdateOrderShippingAndStatusId_InputVariable"
```


Πανεπιστήμιο Πειραιώς

```
        messageType="ns6:UpdateOrderShippingAndStatusId"/>
  <variable name="Invoke17_UpdateOrderShippingAndStatusId_OutputVariable"
        messageType="ns6:UpdateOrderShippingAndStatusIdResponse"/>
  <variable name="Invoke18_GoodsReceptionRequest_InputVariable"
        messageType="ns13:GoodsReceptionRequest"/>
  <variable name="Invoke18_GoodsReceptionRequest_OutputVariable"
        messageType="ns13:GoodsReceptionRequestResponse"/>
  <variable name="ShippingPartnerId" type="xsd:string">
   <from>string("1hj45fdjn78n")</from>
  </variable>
 </variables>
 <correlationSets>
  <correlationSet name="CorrelationSet1"
            properties="ns2:NameCorr"/>
 </correlationSets>
 <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
   ORCHESTRATION LOGIC
   Set of activities coordinating the flow of messages across the
   services integrated within this business process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
 -->
 <sequence name="main">

  <!-- Receive input from requestor. (Note: This maps to operation defined in
OrderProcess.wsdl) -->
  <receive name="receiveInput" partnerLink="orderprocess_client"
portType="client:OrderProcess" operation="OrderProcess" variable="inputVariable"
createInstance="yes"/>

  <!-- Generate reply to synchronous request -->
  <scope name="Scope4">
   <bpelx:annotation>
    <bpelx:general>
     <bpelx:property name="userLabel">IfCreditCardBillIt</bpelx:property>
    </bpelx:general>
   </bpelx:annotation>
   <if name="BillCreditCard">
    <documentation>if Credit Card Selected</documentation>
    <condition>$inputVariable.payload/ns3:paymentMethodId=2</condition>
    <sequence name="Sequence7">
     <assign name="InputForCreditCardGateway">
```


Πανεπιστήμιο Πειραιώς

```
        <copy>
         <from>string($inputVariable.payload/ns3:creditCard/ns3:cardNumber)</from>

<to>$InvokeCCG_CreditCardProcessing_InputVariable.payload/ns8:CardNumber</to>
        </copy>
        <copy>

<from>$inputVariable.payload/ns3:creditCard/ns3:cardExpirationMonth</from>

<to>$InvokeCCG_CreditCardProcessing_InputVariable.payload/ns8:CardExpirationMont
h</to>
        </copy>
        <copy>
         <from>$inputVariable.payload/ns3:creditCard/ns3:cardExpirationYear</from>

<to>$InvokeCCG_CreditCardProcessing_InputVariable.payload/ns8:CardExpirationYear
</to>
        </copy>
        <copy>
         <from>$inputVariable.payload/ns3:creditCard/ns3:cardType</from>

<to>$InvokeCCG_CreditCardProcessing_InputVariable.payload/ns8:CardType</to>
        </copy>
        <copy>
         <from>$inputVariable.payload/ns3:creditCard/ns3:cardUserName</from>

<to>$InvokeCCG_CreditCardProcessing_InputVariable.payload/ns8:CardHoldersName<
/to>
        </copy>
        <copy>

<from>$inputVariable.payload/ns3:creditCard/ns3:cardVerificationCode</from>

<to>$InvokeCCG_CreditCardProcessing_InputVariable.payload/ns8:CardVerificationCod
e</to>
        </copy>
        <copy>
         <from>$inputVariable.payload/ns3:orderPrice</from>

<to>$InvokeCCG_CreditCardProcessing_InputVariable.payload/ns8:CardBillingAmount<
/to>
        </copy>
      </assign>
      <invoke name="InvokeGreditCardGateway"
          partnerLink="CreditCardGateway"
          portType="ns8:CreditCardGateway"
```

Πανεπιστήμιο Πειραιώς

```
            operation="CreditCardProcessing"
            inputVariable="InvokeCCG_CreditCardProcessing_InputVariable"
            outputVariable="InvokeCCG_CreditCardProcessing_OutputVariable"
            bpelx:invokeAsDetail="no"/>
        <assign name="OutputOfCreditCardGateway">
         <copy>

<from>$InvokeCCG_CreditCardProcessing_OutputVariable.payload/ns8:result</from>
          <to>$CCPaymentStatusBoolean</to>
         </copy>
        </assign>
        <if name="CreditCardTransaction">
         <documentation>valid Transaction</documentation>
         <condition>$CCPaymentStatusBoolean</condition>
         <sequence name="Sequence9">
          <empty name="DoNothing"/>
         </sequence>
         <else>
          <documentation>invalid Transaction</documentation>
          <sequence name="Sequence8">
           <assign name="AssignErrorValues">
            <copy>
             <from>false()</from>
             <to>$outputVariable.payload/ns3:OrderStatus</to>
            </copy>
            <copy>
             <from>0</from>
             <to>$outputVariable.payload/ns3:OrderId</to>
            </copy>
           </assign>
           <reply name="InvalidTransactionReply"
                variable="outputVariable"
                partnerLink="orderprocess_client"
                portType="client:OrderProcess" operation="OrderProcess"/>
           <exit name="Exit2"/>
          </sequence>
         </else>
        </if>
       </sequence>
       <else>
        <documentation>If other method selected</documentation>
        <empty name="NoCreditCardSelected"/>
       </else>
      </if>
     </scope>
     <scope name="Scope1">
```


Πανεπιστήμιο Πειραιώς

```
    <bpelx:annotation>
     <bpelx:general>
      <bpelx:property name="userLabel">InsertOrder</bpelx:property>
     </bpelx:general>
    </bpelx:annotation>
    <variables>
     <variable name="RepeatUntil_Invoke_ProductToOrder" type="xsd:boolean"/>
    </variables>
    <sequence>
     <assign name="InputVariablesForSubmitingOrder">
      <copy>

<from>$inputVariable.payload/ns3:UserInformation/ns3:DeliveryAddressId</from>

<to>$Invoke2_InsertOrder_InputVariable.parameters/order/deliveryAddressId</to>
      </copy>
      <copy>
       <from>$inputVariable.payload/ns3:UserInformation/ns3:UserId</from>
       <to>$Invoke2_InsertOrder_InputVariable.parameters/order/userId</to>
      </copy>
      <copy>
       <from>$inputVariable.payload/ns3:paymentMethodId</from>

<to>$Invoke2_InsertOrder_InputVariable.parameters/order/paymentMethodId</to>
      </copy>
      <copy>
       <from>$inputVariable.payload/ns3:orderPrice</from>
       <to>$Invoke2_InsertOrder_InputVariable.parameters/order/orderPrice</to>
      </copy>
     </assign>
     <invoke name="SubmitOrderAndGetOrderId" partnerLink="Repository"
          portType="ns6:RepositoryWS" operation="InsertOrder"
          inputVariable="Invoke2_InsertOrder_InputVariable"
          outputVariable="Invoke2_InsertOrder_OutputVariable"
          bpelx:invokeAsDetail="no"/>
     <forEach parallel="no" counterName="ForEach1Counter"
          name="ForEachProductInsertationToOrder">
      <startCounterValue>1</startCounterValue>

<finalCounterValue>count($inputVariable.payload/ns3:products)</finalCounterValue>
      <scope name="Scope2">
       <sequence name="Sequence2">
        <assign name="AssignVariablesToInvokeAndInitializeWhile">
         <copy>

<from>$inputVariable.payload/ns3:products[position()=$ForEach1Counter]/ns3:produ
```


Πανεπιστήμιο Πειραιώς

```
ctId</from>

<to>$Invoke3_InsertProductToOrder_InputVariable.parameters/productToOrder/prod
uctId</to>
          </copy>
          <copy>

<from>$inputVariable.payload/ns3:products[position()=$ForEach1Counter]/ns3:produ
ctQuantity</from>

<to>$Invoke3_InsertProductToOrder_InputVariable.parameters/productToOrder/prod
uctQuantity</to>
          </copy>
          <copy>

<from>$Invoke2_InsertOrder_OutputVariable.parameters/return/orderId</from>

<to>$Invoke3_InsertProductToOrder_InputVariable.parameters/productToOrder/order
Id</to>
          </copy>
          <copy>
           <from>true()</from>
           <to>$RepeatUntil_Invoke_ProductToOrder</to>
          </copy>
         </assign>
         <while name="WhileThereAreProducts">
          <condition>$RepeatUntil_Invoke_ProductToOrder</condition>
          <sequence name="Sequence3">
           <invoke name="InsertProductsToOrder" partnerLink="Repository"
                 portType="ns6:RepositoryWS"
                 operation="InsertProductToOrder"
                 inputVariable="Invoke3_InsertProductToOrder_InputVariable"
                 outputVariable="Invoke3_InsertProductToOrder_OutputVariable"
                 bpelx:invokeAsDetail="no"/>
           <assign name="StopWhile">
            <copy>

<from>($Invoke3_InsertProductToOrder_OutputVariable.parameters/return!=true())</
from>
             <to>$RepeatUntil_Invoke_ProductToOrder</to>
            </copy>
           </assign>
          </sequence>
         </while>
         <assign name="AssignCustomerReturnValues">
          <copy>
```


Πανεπιστήμιο Πειραιώς

```
<from>$Invoke2_InsertOrder_OutputVariable.parameters/return/orderId</from>
            <to>$outputVariable.payload/ns3:OrderId</to>
           </copy>
           <copy>
            <from>$ForEach1Counter</from>
            <to>$ProductsCount</to>
           </copy>
           <copy>

<from>$Invoke2_InsertOrder_OutputVariable.parameters/return/status</from>
            <to>$outputVariable.payload/ns3:OrderStatus</to>
           </copy>
          </assign>
         </sequence>
        </scope>
       </forEach>
      </sequence>
     </scope>
     <assign name="AssignGlobalVariables">
      <copy>
       <from>$Invoke2_InsertOrder_OutputVariable.parameters/return/orderId</from>
       <to>$OrderID</to>
      </copy>
      <copy>
       <from>$inputVariable.payload/ns3:UserInformation/ns3:UserId</from>
       <to>$UserId</to>
      </copy>
     </assign>
     <reply name="ReturnOrderId" partnerLink="orderprocess_client"
          portType="client:OrderProcess" operation="OrderProcess"
          variable="outputVariable"/>
     <scope name="Scope3">
      <bpelx:annotation>
       <bpelx:general>
        <bpelx:property name="userLabel">Payment</bpelx:property>
       </bpelx:general>
      </bpelx:annotation>
      <variables>
       <variable name="BankPaymentBooleanforIF" type="xsd:boolean"/>
      </variables>
      <sequence name="Sequence6">
       <flow name="Flow1">
        <sequence name="Sequence">
         <if name="SelectPaymentMethod">
          <documentation>Bank Deposit</documentation>
```


Πανεπιστήμιο Πειραιώς

```
        <condition>($inputVariable.payload/ns3:paymentMethodId=1)</condition>
        <sequence>
         <assign name="BankVerificationInput">
          <copy>

<from>$Invoke2_InsertOrder_OutputVariable.parameters/return/orderId</from>
            <to>$Invoke1_process_InputVariable.payload/ns1:OrderNumber</to>
          </copy>
          <copy>
           <from>$inputVariable.payload/ns3:orderPrice</from>
           <to>$Invoke1_process_InputVariable.payload/ns1:OrderPrice</to>
          </copy>
         </assign>
         <invoke name="SendDataForBankVerification" partnerLink="BankVerification"
              portType="ns1:BankVerificationProcess"
              operation="process"
              inputVariable="Invoke1_process_InputVariable"
              bpelx:invokeAsDetail="no"/>
        <receive name="GetBankVerificationOutput" createInstance="no"
              partnerLink="BankVerification"
              portType="ns1:BankVerificationProcessCallback"
              operation="processResponse"
              variable="Receive1_processResponse_InputVariable"/>
        <assign name="AssignReceive">
         <copy>

<from>$Receive1_processResponse_InputVariable.payload/ns1:result</from>
            <to>$BankPaymentBooleanforIF</to>
          </copy>
         </assign>
         <if name="CheckIfBankDepositReceived">
          <documentation>Received</documentation>
          <condition>$BankPaymentBooleanforIF</condition>
          <sequence>
           <assign name="InputForMailContentGenerator">
            <copy>
             <from>$OrderID</from>

<to>$InvokeOMCG_CreateOrderDetails_InputVariable.parameters/OrderNumber</to>
            </copy>
            <copy>
             <from>string("Bank Payment Verification True")</from>

<to>$InvokeOMCG_CreateOrderDetails_InputVariable.parameters/MailOperation</to>
            </copy>
            <copy>
```


Πανεπιστήμιο Πειραιώς

```
            <from>$UserId</from>

<to>$InvokeOMCG_CreateOrderDetails_InputVariable.parameters/UserID</to>
           </copy>
          </assign>
          <invoke name="GetMailContent1"
               partnerLink="OrderMailContentGenerator"
               portType="ns6:OrderEmailGeneratorWS"
               operation="CreateOrderDetails"
               inputVariable="InvokeOMCG_CreateOrderDetails_InputVariable"
               outputVariable="InvokeOMCG_CreateOrderDetails_OutputVariable"
               bpelx:invokeAsDetail="no"/>
          <scope name="Email2">
           <bpelx:annotation>
            <bpelx:pattern patternName="bpelx:email"/>
            <bpelx:general>
             <bpelx:property
name="userLabel">BankPaymentVerification</bpelx:property>
            </bpelx:general>
           </bpelx:annotation>
           <variables>
            <variable name="varNotificationReq"
                  messageType="ns4:EmailNotificationRequest"/>
            <variable name="varNotificationResponse"
                  messageType="ns4:ArrayOfResponse"/>
            <variable name="NotificationServiceFaultVariable"
                  messageType="ns4:NotificationServiceErrorMessage"/>
           </variables>
           <sequence name="Sequence11">
            <assign name="EmailParamsAssign">
             <copy>
              <from>string('Default')</from>
              <to variable="varNotificationReq"
                part="EmailPayload"><query>ns4:FromAccountName</query></to>
             </copy>
             <copy>
              <from>string('')</from>
              <to variable="varNotificationReq"
                part="EmailPayload"><query>ns4:Bcc</query></to>
             </copy>
             <copy>
              <from>string('')</from>
              <to variable="varNotificationReq"
                part="EmailPayload"><query>ns4:Cc</query></to>
             </copy>
             <copy>
```


Πανεπιστήμιο Πειραιώς

```
                <from>string('')</from>
                <to variable="varNotificationReq"
                   part="EmailPayload"><query>ns4:ReplyToAddress</query></to>
               </copy>
               <copy>
                <from>string('Bank Payment Received')</from>
                <to variable="varNotificationReq"
                   part="EmailPayload"><query>ns4:Subject</query></to>
               </copy>
               <copy>

<from>$InvokeOMCG_CreateOrderDetails_OutputVariable.parameters/return/email</f
rom>
                <to variable="varNotificationReq"
                   part="EmailPayload"><query>ns4:To</query></to>
               </copy>
               <copy>

<from>$InvokeOMCG_CreateOrderDetails_OutputVariable.parameters/return/content<
/from>
                <to variable="varNotificationReq"

part="EmailPayload"><query>ns4:Content/ns4:ContentBody</query></to>
               </copy>
               <copy>
                <from>string('text/html; charset=UTF-8')</from>
                <to variable="varNotificationReq"

part="EmailPayload"><query>ns4:Content/ns4:MimeType</query></to>
               </copy>
              </assign>
              <invoke name="InvokeNotificationService"
                   portType="ns4:NotificationService"
                   partnerLink="NotificationService2"
                   inputVariable="varNotificationReq"
                   outputVariable="varNotificationResponse"
                   operation="sendEmailNotification"/>
             </sequence>
            </scope>
           </sequence>
           <else>
            <documentation>NotReceived</documentation>
            <empty name="DoNothing2"/>
           </else>
          </if>
         </sequence>
```


Πανεπιστήμιο Πειραιώς

```
          <else>
           <documentation>Cash</documentation>
           <sequence name="Sequence5">
            <empty name="CashSelectedDoNothing"/>
           </sequence>
          </else>
         </if>
        </sequence>
        <sequence name="Sequence6">
         <assign name="AssignOrderId">
          <copy>

<from>$Invoke2_InsertOrder_OutputVariable.parameters/return/orderId</from>

<to>$InvokeOMCG_CreateOrderDetails_InputVariable.parameters/OrderNumber</to>
          </copy>
          <copy>
           <from>string("Order Details")</from>

<to>$InvokeOMCG_CreateOrderDetails_InputVariable.parameters/MailOperation</to>
          </copy>
          <copy>
           <from>$UserId</from>
           <to>$InvokeOMCG_CreateOrderDetails_InputVariable.parameters/UserID</to>
          </copy>
         </assign>
         <invoke name="GetMailContent2"
              partnerLink="OrderMailContentGenerator"
              portType="ns6:OrderEmailGeneratorWS"
              operation="CreateOrderDetails"
              inputVariable="InvokeOMCG_CreateOrderDetails_InputVariable"
              outputVariable="InvokeOMCG_CreateOrderDetails_OutputVariable"
              bpelx:invokeAsDetail="no"/>
         <scope name="Email1">
          <bpelx:annotation>
           <bpelx:pattern patternName="bpelx:email"/>
           <bpelx:general>
            <bpelx:property name="userLabel">OrderDetails</bpelx:property>
           </bpelx:general>
          </bpelx:annotation>
          <variables>
           <variable name="varNotificationReq"
                messageType="ns4:EmailNotificationRequest"/>
           <variable name="varNotificationResponse"
                messageType="ns4:ArrayOfResponse"/>
           <variable name="NotificationServiceFaultVariable"
```


Πανεπιστήμιο Πειραιώς

```
                messageType="ns4:NotificationServiceErrorMessage"/>
        </variables>
        <sequence name="Sequence1">
         <assign name="EmailParamsAssign">
          <copy>
           <from>string('Default')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:FromAccountName</query></to>
          </copy>
          <copy>
           <from>string('')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Bcc</query></to>
          </copy>
          <copy>
           <from>string('')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Cc</query></to>
          </copy>
          <copy>
           <from>string('')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:ReplyToAddress</query></to>
          </copy>
          <copy>
           <from>string('Thank you for your order')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Subject</query></to>
          </copy>
          <copy>

<from>$InvokeOMCG_CreateOrderDetails_OutputVariable.parameters/return/email</f
rom>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:To</query></to>
          </copy>
          <copy>

<from>$InvokeOMCG_CreateOrderDetails_OutputVariable.parameters/return/content<
/from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Content/ns4:ContentBody</query></to>
          </copy>
          <copy>
           <from>string('text/html; charset=UTF-8')</from>
           <to variable="varNotificationReq"
```


Πανεπιστήμιο Πειραιώς

```
part="EmailPayload"><query>ns4:Content/ns4:MimeType</query></to>
          </copy>
         </assign>
         <invoke name="InvokeNotificationService"
              portType="ns4:NotificationService"
              partnerLink="NotificationService1"
              inputVariable="varNotificationReq"
              outputVariable="varNotificationResponse"
              operation="sendEmailNotification"/>
        </sequence>
       </scope>
      </sequence>
     </flow>
    </sequence>
   </scope>
   <scope name="Scope5">
    <bpelx:annotation>
     <bpelx:general>
      <bpelx:property name="userLabel">ProcessOrderAndSupply</bpelx:property>
     </bpelx:general>
    </bpelx:annotation>
    <sequence name="Sequence13">
     <assign name="TransformInputForFurtherProcessing">
      <bpelx:annotation>
       <bpelx:pattern patternName="bpelx:transformation"/>
      </bpelx:annotation>
      <copy>
       <from>ora:doXSLTransformForDoc("xsl/Transformation_2.xsl",
$inputVariable.payload, "Invoke2_InsertOrder_OutputVariable.parameters",
$Invoke2_InsertOrder_OutputVariable.parameters)</from>
        <to variable="Invoke5_processOrderAndSuppliers_InputVariable"
          part="payload"/>
      </copy>
     </assign>
     <invoke name="SendOrderForFurtherProcessing"
          partnerLink="ProcessOrderAndSupply"
          portType="ns11:ProcessOrderAndSupplier"
          operation="processOrderAndSuppliers"
          inputVariable="Invoke5_processOrderAndSuppliers_InputVariable"
          outputVariable="Invoke5_processOrderAndSuppliers_OutputVariable"
          bpelx:invokeAsDetail="no"/>
    </sequence>
   </scope>
   <scope name="Scope6">
    <bpelx:annotation>
     <bpelx:general>
```


Πανεπιστήμιο Πειραιώς

```
      <bpelx:property
name="userLabel">CheckIfProuctsOfOrderFulfilled</bpelx:property>
    </bpelx:general>
   </bpelx:annotation>
   <variables>
    <variable name="WhileVariable" type="xsd:boolean"/>
    <variable name="ifVariable" type="xsd:boolean"/>
   </variables>
   <sequence name="Sequence14">
    <assign name="InitializeRepeatFulfillmentCheckVariable">
     <copy>
      <from>true()</from>
      <to>$WhileVariable</to>
     </copy>
    </assign>
    <while name="RepeatFulfillmentCheck">
     <condition>$WhileVariable</condition>
     <sequence>
      <assign name="InputForFulfillmentCheck">
       <copy>
        <from>$OrderID</from>
        <to>$Invoke6_CheckOrderStatus_InputVariable.parameters/OrderId</to>
       </copy>
       <copy>
        <from>$ProductsCount</from>
        <to>$Invoke6_CheckOrderStatus_InputVariable.parameters/Records</to>
       </copy>
      </assign>
      <invoke name="CheckIfFulfilled" partnerLink="Repository"
           portType="ns6:RepositoryWS" operation="CheckOrderStatus"
           inputVariable="Invoke6_CheckOrderStatus_InputVariable"
           outputVariable="Invoke6_CheckOrderStatus_OutputVariable"
           bpelx:invokeAsDetail="no"/>
      <assign name="OutputOfFulfillmentCheck">
       <copy>

<from>$Invoke6_CheckOrderStatus_OutputVariable.parameters/return</from>
        <to>$ifVariable</to>
       </copy>
      </assign>
      <if name="CheckFulfillment">
       <documentation>Products of order fulfilled</documentation>
       <condition>$ifVariable</condition>
       <assign name="TerminateRepeats">
        <copy>
         <from>false()</from>
```


Πανεπιστήμιο Πειραιώς

```
            <to>$WhileVariable</to>
           </copy>
          </assign>
          <else>
           <documentation>Products of order not fulfilled</documentation>
           <wait name="WaitAndCheckAgain">
            <for>'PT1M'</for>
           </wait>
          </else>
         </if>
        </sequence>
       </while>
      </sequence>
     </scope>
     <scope name="Scope7">
      <bpelx:annotation>
       <bpelx:general>
        <bpelx:property name="userLabel">FulfillOrder</bpelx:property>
       </bpelx:general>
      </bpelx:annotation>
      <variables>
       <variable name="OrderFulfillmentOutputBooleanCheck" type="xsd:boolean"/>
      </variables>
      <sequence name="Sequence15">
       <assign name="InsertReceipeInput">
        <copy>
         <from>$inputVariable.payload/ns3:OrderReceiptTypeId</from>
         <to>$Invoke13_InsertReceipt_InputVariable.parameters/receiptTypeId</to>
        </copy>
       </assign>
       <invoke name="InsertReceipt"
            partnerLink="Repository1" portType="ns6:Repository1WS"
            operation="InsertReceipt"
            inputVariable="Invoke13_InsertReceipt_InputVariable"
            outputVariable="Invoke13_InsertReceipt_OutputVariable"
            bpelx:invokeAsDetail="no"/>
       <assign name="UpdateOrderInput">
        <copy>

<from>$Invoke13_InsertReceipt_OutputVariable.parameters/return/receiptId</from>
         <to>$Invoke14_UpdateOrderReceiptId_InputVariable.parameters/receiptId</to>
        </copy>
        <copy>
         <from>$OrderID</from>
         <to>$Invoke14_UpdateOrderReceiptId_InputVariable.parameters/orderId</to>
        </copy>
```


Πανεπιστήμιο Πειραιώς

```
    </assign>
    <invoke name="UpdateOrderInsertReceiptId"
        partnerLink="Repository1" portType="ns6:Repository1WS"
        operation="UpdateOrderReceiptId"
        outputVariable="Invoke14_UpdateOrderReceiptId_OutputVariable"
        inputVariable="Invoke14_UpdateOrderReceiptId_InputVariable"
        bpelx:invokeAsDetail="no"/>
    <assign name="InputForInvokesToGetReceipOrInvoiceInformaton">
     <copy>
      <from>$inputVariable.payload/ns3:UserInformation/ns3:UserId</from>
      <to>$Invoke9_GetUserInformation_InputVariable.parameters/userId</to>
     </copy>
     <copy>

<from>$inputVariable.payload/ns3:UserInformation/ns3:DeliveryAddressId</from>

<to>$Invoke11_GetAddressInformation_InputVariable.parameters/addressId</to>
     </copy>
     <copy>
      <from>$OrderID</from>
      <to>$Invoke12_GetOrderInformation_InputVariable.parameters/orderId</to>
     </copy>
     <copy>

<from>$Invoke13_InsertReceipt_OutputVariable.parameters/return/receiptId</from>
      <to>$Invoke10_GetReceiptInformation_InputVariable.parameters/receiptId</to>
     </copy>
     <copy>
      <from>$OrderID</from>
      <to>$Invoke8_GetOrderProductDetails_InputVariable.parameters/orderId</to>
     </copy>
    </assign>
    <flow name="Flow2">
     <sequence name="Sequence19">
      <invoke name="GetOrderProductsDetails" partnerLink="Repository1"
          portType="ns6:Repository1WS"
          operation="GetOrderProductDetails"
          inputVariable="Invoke8_GetOrderProductDetails_InputVariable"
          outputVariable="Invoke8_GetOrderProductDetails_OutputVariable"
          bpelx:invokeAsDetail="no"/>
     </sequence>
     <sequence name="Sequence18">
      <invoke name="GetOrderDetails"
          partnerLink="Repository1" portType="ns6:Repository1WS"
          operation="GetOrderInformation"
          inputVariable="Invoke12_GetOrderInformation_InputVariable"
```


Πανεπιστήμιο Πειραιώς

```
            outputVariable="Invoke12_GetOrderInformation_OutputVariable"
            bpelx:invokeAsDetail="no"/>
      </sequence>
      <sequence name="Sequence17">
       <invoke name="GetReceiptDetails"
            partnerLink="Repository1" portType="ns6:Repository1WS"
            operation="GetReceiptInformation"
            inputVariable="Invoke10_GetReceiptInformation_InputVariable"
            outputVariable="Invoke10_GetReceiptInformation_OutputVariable"
            bpelx:invokeAsDetail="no"/>
      </sequence>
      <sequence name="Sequence">
       <invoke name="GetUserDetails"
            partnerLink="Repository1" portType="ns6:Repository1WS"
            operation="GetUserInformation"
            inputVariable="Invoke9_GetUserInformation_InputVariable"
            outputVariable="Invoke9_GetUserInformation_OutputVariable"
            bpelx:invokeAsDetail="no"/>
      </sequence>
      <sequence name="Sequence16">
       <invoke name="GetAddressDetails"
            partnerLink="Repository1" portType="ns6:Repository1WS"
            operation="GetAddressInformation"
            inputVariable="Invoke11_GetAddressInformation_InputVariable"
            outputVariable="Invoke11_GetAddressInformation_OutputVariable"
            bpelx:invokeAsDetail="no"/>
      </sequence>
     </flow>
     <assign name="GetOutPutAndTransformItForInputToPdfGenerator">
      <bpelx:annotation>
       <bpelx:pattern patternName="bpelx:transformation"/>
      </bpelx:annotation>
      <copy>
       <from>ora:doXSLTransformForDoc("xsl/invoicepdf.xsl",
$Invoke9_GetUserInformation_OutputVariable.parameters,
"Invoke10_GetReceiptInformation_OutputVariable.parameters",
$Invoke10_GetReceiptInformation_OutputVariable.parameters,
"Invoke11_GetAddressInformation_OutputVariable.parameters",
$Invoke11_GetAddressInformation_OutputVariable.parameters,
"Invoke12_GetOrderInformation_OutputVariable.parameters",
$Invoke12_GetOrderInformation_OutputVariable.parameters,
"Invoke8_GetOrderProductDetails_OutputVariable.parameters",
$Invoke8_GetOrderProductDetails_OutputVariable.parameters,
"Invoke2_InsertOrder_OutputVariable.parameters",
$Invoke2_InsertOrder_OutputVariable.parameters)</from>
       <to variable="Invoke15_produceReciptPDF_InputVariable"
```


Πανεπιστήμιο Πειραιώς

```
          part="parameters"/>
       </copy>
      </assign>
      <invoke name="GenerateInvoiceOrReceiptPDF" bpelx:invokeAsDetail="no"
          partnerLink="InvoiceGenerator"
          portType="ns6:InvoicePDFDocumentWS"
          operation="produceReciptPDF"
          inputVariable="Invoke15_produceReciptPDF_InputVariable"
          outputVariable="Invoke15_produceReciptPDF_OutputVariable"/>
      <assign name="GetOutputAndTransformItForInput">
       <bpelx:annotation>
        <bpelx:pattern patternName="bpelx:transformation"/>
       </bpelx:annotation>
       <copy>
        <from>ora:doXSLTransformForDoc("xsl/getOrderDetails.xsl",
$Invoke8_GetOrderProductDetails_OutputVariable.parameters,
"Invoke2_InsertOrder_OutputVariable.parameters",
$Invoke2_InsertOrder_OutputVariable.parameters)</from>
         <to variable="Invoke7_OrderFulfillment_InputVariable"
          part="payload"/>
       </copy>
      </assign>
      <invoke name="OrderFulfillmentInvoke"
          partnerLink="OrderFulfillmentHT"
          portType="ns12:OrderFulfillment" operation="process"
          inputVariable="Invoke7_OrderFulfillment_InputVariable"
          bpelx:invokeAsDetail="no"/>
      <receive name="ReceiveOrderFulfillment" createInstance="no"
           partnerLink="OrderFulfillmentHT"
           portType="ns12:OrderFulfillmentCallback"
           operation="processResponse"
           variable="Inkoke7_ReceiveOrderFulfillment_OutputVariable"/>
      <assign name="AssignValuesForChecking">
       <copy>

<from>$Inkoke7_ReceiveOrderFulfillment_OutputVariable.payload/ns12:result</from>
         <to>$OrderFulfillmentOutputBooleanCheck</to>
       </copy>
      </assign>
      <if name="checkReturnOfOrderFulfillment">
       <documentation>Order fulfilled</documentation>
       <condition>$OrderFulfillmentOutputBooleanCheck</condition>
       <empty name="DoNothing3"/>
       <else>
        <documentation>Order not fulfilled</documentation>
        <exit name="Exit3"/>
```


Πανεπιστήμιο Πειραιώς

```
      </else>
     </if>
    </sequence>
   </scope>
   <scope name="Scope8">
    <bpelx:annotation>
     <bpelx:general>
      <bpelx:property name="userLabel">DispatchOrder</bpelx:property>
     </bpelx:general>
    </bpelx:annotation>
    <sequence name="Sequence20">
     <assign name="InputForShippingService">
      <copy>
       <from>number(1)</from>

<to>$Invoke18_GoodsReceptionRequest_InputVariable.parameters/PartnerLocationId<
/to>
      </copy>
      <copy>
       <from>$ShippingPartnerId</from>

<to>$Invoke18_GoodsReceptionRequest_InputVariable.parameters/PartnerId</to>
      </copy>
     </assign>
     <invoke name="RequestShippingService"
          inputVariable="Invoke18_GoodsReceptionRequest_InputVariable"
          outputVariable="Invoke18_GoodsReceptionRequest_OutputVariable"
          partnerLink="ShippingCompany" portType="ns13:ShippingCompanyWS"
          operation="GoodsReceptionRequest" bpelx:invokeAsDetail="no"/>
     <flow name="Flow3">
      <sequence name="Sequence">
       <assign name="UpdateOrderInput">
        <copy>

<from>$Invoke18_GoodsReceptionRequest_OutputVariable.parameters/return/orderId
</from>

<to>$Invoke17_UpdateOrderShippingAndStatusId_InputVariable.parameters/shippingT
rackingId</to>
        </copy>
        <copy>
         <from>$OrderID</from>

<to>$Invoke17_UpdateOrderShippingAndStatusId_InputVariable.parameters/orderId</
to>
        </copy>
```


Πανεπιστήμιο Πειραιώς

```
        </assign>
        <invoke name="UpdateOrderStatusAndShippingId"
              partnerLink="Repository1" portType="ns6:Repository1WS"
              operation="UpdateOrderShippingAndStatusId"
              inputVariable="Invoke17_UpdateOrderShippingAndStatusId_InputVariable"

outputVariable="Invoke17_UpdateOrderShippingAndStatusId_OutputVariable"
              bpelx:invokeAsDetail="no"/>
       </sequence>
       <sequence name="Sequence21">
        <scope name="Email4">
         <bpelx:annotation>
          <bpelx:pattern patternName="bpelx:email"/>
          <bpelx:general>
           <bpelx:property
name="userLabel">FinalEmailWithTrackingId</bpelx:property>
          </bpelx:general>
         </bpelx:annotation>
         <variables>
          <variable name="varNotificationReq"
                messageType="ns4:EmailNotificationRequest"/>
          <variable name="varNotificationResponse"
                messageType="ns4:ArrayOfResponse"/>
          <variable name="NotificationServiceFaultVariable"
                messageType="ns4:NotificationServiceErrorMessage"/>
         </variables>
         <sequence name="Sequence22">
          <assign name="EmailParamsAssign">
           <copy>
            <from>string('Default')</from>
            <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:FromAccountName</query></to>
           </copy>
           <copy>
            <from>string('')</from>
            <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Bcc</query></to>
           </copy>
           <copy>
            <from>string('')</from>
            <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Cc</query></to>
           </copy>
           <copy>
            <from>string('')</from>
            <to variable="varNotificationReq"
```


Πανεπιστήμιο Πειραιώς

```
part="EmailPayload"><query>ns4:ReplyToAddress</query></to>
            </copy>
            <copy>
              <from>string('Your order has been dispatched')</from>
              <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Subject</query></to>
            </copy>
            <copy>

<from>$Invoke9_GetUserInformation_OutputVariable.parameters/return/userEmail</f
rom>
              <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:To</query></to>
            </copy>
            <copy>
              <from>concat(string('Your order with id: '), $OrderID, string(' has been
dispatched, you can view your order status from now on  through the following link:
http://www.shippingcompany.com/tracking.php?trackingId='),
$Invoke18_GoodsReceptionRequest_OutputVariable.parameters/return/orderId,
string('
Thank you for choosing us!'))</from>
              <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Content/ns4:ContentBody</query></to>
            </copy>
            <copy>
              <from>string('text/html; charset=UTF-8')</from>
              <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Content/ns4:MimeType</query></to>
            </copy>
          </assign>
          <invoke name="InvokeNotificationService"
                portType="ns4:NotificationService"
                partnerLink="NotificationService4"
                inputVariable="varNotificationReq"
                outputVariable="varNotificationResponse"
                operation="sendEmailNotification"/>
        </sequence>
       </scope>
      </sequence>
    </flow>
   </sequence>
  </scope>
 </sequence>
</process>
```

Κώδικας 38: OrderProcessv01.xsd

Πανεπιστήμιο Πειραιώς

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema attributeFormDefault="unqualified"
          elementFormDefault="qualified"
          targetNamespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/Buy
ingProcessProject1/BuyingProcessBpelV01"
          xmlns:xs="http://www.w3.org/2001/XMLSchema"
     >
 <xs:element name="Order">
 <xs:complexType>
<xs:sequence maxOccurs="1">
   <xs:element name="UserInformation">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="UserId" type="xs:unsignedInt"/>
      <xs:element name="DeliveryAddressId" type="xs:unsignedInt"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="paymentMethodId" type="xs:unsignedInt" maxOccurs="1"/>
   <xs:element name="OrderReceiptTypeId" type="xs:unsignedInt"/>
   <xs:element name="orderPrice" type="xs:float" maxOccurs="1"/>
   <xs:element name="products" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="productId" type="xs:unsignedInt" maxOccurs="1"/>
      <xs:element name="productQuantity" type="xs:unsignedInt" maxOccurs="1"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="creditCard" maxOccurs="1" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="cardExpirationMonth" type="xs:unsignedInt"
            minOccurs="1" maxOccurs="1"/>
      <xs:element name="cardExpirationYear" type="xs:unsignedInt" minOccurs="1"
            maxOccurs="1"/>
      <xs:element name="cardNumber" type="xs:double" minOccurs="1"
            maxOccurs="1"/>
      <xs:element name="cardType" type="xs:string" minOccurs="1"
            maxOccurs="1"/>
      <xs:element name="cardUserName" type="xs:string" minOccurs="1"
            maxOccurs="1"/>
      <xs:element name="cardVerificationCode" type="xs:unsignedInt"
            minOccurs="1" maxOccurs="1"/>
     </xs:sequence>
    </xs:complexType>
```


Πανεπιστήμιο Πειραιώς

```
   </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="OrderResponse">
<xs:complexType >
<xs:sequence>
<xs:element name="OrderId" type="xs:unsignedInt" minOccurs="0"/>
   <xs:element name="OrderStatus" type="xs:boolean"/>
  </xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
```

Κώδικας 39: Transformation_2.xsl

```
<?xml version="1.0" encoding="UTF-8" ?>
<?oracle-xsl-mapper
 <!-- SPECIFICATION OF MAP SOURCES AND TARGETS, DO NOT MODIFY. -->
 <mapSources>
  <source type="XSD">
   <schema location="../xsd/OrderProcessv01.xsd"/>
   <rootElement name="Order"
namespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessPr
oject1/BuyingProcessBpelV01"/>
  </source>
  <source type="WSDL">
   <schema
location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"/>
   <rootElement name="InsertOrderResponse"
namespace="http://webservice.brandname.store.com/"/>
   <param name="Invoke2_InsertOrder_OutputVariable.parameters" />
  </source>
 </mapSources>
 <mapTargets>
  <target type="WSDL">
   <schema location="http://10.19.149.3:8001/soa-
infra/services/thesis/ProcessOrderAndSuppliers/ProcessOrderAndSupplier.wsdl"/>
   <rootElement name="process"
namespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAnd
Suppliers/ProcessOrderAndSupplier"/>
  </target>
 </mapTargets>
 <!-- GENERATED BY ORACLE XSL MAPPER 11.1.1.5.0(build 110418.1550.0174) AT
[THU FEB 16 17:58:14 EET 2012]. -->
?>
<xsl:stylesheet version="1.0"
```


Πανεπιστήμιο Πειραιώς

```
xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20"
        xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
        xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns:client="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAn
dSuppliers/ProcessOrderAndSupplier"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
        xmlns:ora="http://schemas.oracle.com/xpath/extension"

xmlns:socket="http://www.oracle.com/XSL/Transform/java/oracle.tip.adapter.socket.
ProtocolTranslator"
        xmlns:wsp="http://www.w3.org/ns/ws-policy"

xmlns:mhdr="http://www.oracle.com/XSL/Transform/java/oracle.tip.mediator.service
.common.functions.MediatorExtnFunction"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
        xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
        xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
        xmlns:med="http://schemas.oracle.com/mediator/xpath"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
        xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions"
        xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:ns0="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessProj
ect1/BuyingProcessBpelV01"
        xmlns:bpmn="http://schemas.oracle.com/bpm/xpath"
        xmlns:tns="http://webservice.brandname.store.com/"
        xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
        xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
```


Πανεπιστήμιο Πειραιώς

```
            exclude-result-prefixes="xsi xsl soap wsp wsp1_2 wsu xsd ns0 tns wsam client
wsdl plnk xp20 bpws bpel bpm ora socket mhdr oraext dvm hwf med ids xdk xref bpmn
ldap">
 <xsl:param name="Invoke2_InsertOrder_OutputVariable.parameters"/>
 <xsl:template match="/">
  <client:process>
   <xsl:for-each select="/ns0:Order/ns0:products">
    <client:Products>
     <client:ProductId>
      <xsl:value-of select="ns0:productId"/>
     </client:ProductId>
     <client:Quantity>
      <xsl:value-of select="ns0:productQuantity"/>
     </client:Quantity>
    </client:Products>
   </xsl:for-each>
   <client:OrderId>
    <xsl:value-of
select="$Invoke2_InsertOrder_OutputVariable.parameters/tns:InsertOrderResponse/re
turn/orderId"/>
   </client:OrderId>
  </client:process>
 </xsl:template>
</xsl:stylesheet>
```

Κώδικας 40: invoicepdf.xsl

```
<?xml version="1.0" encoding="UTF-8" ?>
<?oracle-xsl-mapper
 <!-- SPECIFICATION OF MAP SOURCES AND TARGETS, DO NOT MODIFY. -->
 <mapSources>
  <source type="WSDL">
   <schema
location="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?WSDL"/
>
   <rootElement name="GetUserInformationResponse"
namespace="http://webservice.brandname.store.com/"/>
  </source>
  <source type="WSDL">
   <schema
location="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?WSDL"/
>
   <rootElement name="GetReceiptInformationResponse"
namespace="http://webservice.brandname.store.com/"/>
   <param name="Invoke10_GetReceiptInformation_OutputVariable.parameters" />
  </source>
  <source type="WSDL">
```


Πανεπιστήμιο Πειραιώς

```
    <schema
location="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?WSDL"/
>
    <rootElement name="GetAddressInformationResponse"
namespace="http://webservice.brandname.store.com/"/>
    <param name="Invoke11_GetAddressInformation_OutputVariable.parameters" />
   </source>
   <source type="WSDL">
    <schema
location="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?WSDL"/
>
    <rootElement name="GetOrderInformationResponse"
namespace="http://webservice.brandname.store.com/"/>
    <param name="Invoke12_GetOrderInformation_OutputVariable.parameters" />
   </source>
   <source type="WSDL">
    <schema
location="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?WSDL"/
>
    <rootElement name="GetOrderProductDetailsResponse"
namespace="http://webservice.brandname.store.com/"/>
    <param name="Invoke8_GetOrderProductDetails_OutputVariable.parameters" />
   </source>
   <source type="WSDL">
    <schema
location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"/>
    <rootElement name="InsertOrderResponse"
namespace="http://webservice.brandname.store.com/"/>
    <param name="Invoke2_InsertOrder_OutputVariable.parameters" />
   </source>
  </mapSources>
  <mapTargets>
   <target type="WSDL">
    <schema
location="http://176.34.182.139:8888/ThesisInvoceService/InvoicePDFDocumentWS?
WSDL"/>
    <rootElement name="produceReciptPDF"
namespace="http://webservice.brandname.store.com/"/>
   </target>
  </mapTargets>
  <!-- GENERATED BY ORACLE XSL MAPPER 11.1.1.5.0(build 110418.1550.0174) AT
[MON FEB 20 22:15:51 EET 2012]. -->
?>
<xsl:stylesheet version="1.0"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
```


Πανεπιστήμιο Πειραιώς

```
ons.Xpath20"
        xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
        xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
        xmlns:ora="http://schemas.oracle.com/xpath/extension"

xmlns:socket="http://www.oracle.com/XSL/Transform/java/oracle.tip.adapter.socket.
ProtocolTranslator"
        xmlns:wsp="http://www.w3.org/ns/ws-policy"

xmlns:mhdr="http://www.oracle.com/XSL/Transform/java/oracle.tip.mediator.service
.common.functions.MediatorExtnFunction"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
        xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
        xmlns:med="http://schemas.oracle.com/mediator/xpath"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
        xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd"
        xmlns:bpmn="http://schemas.oracle.com/bpm/xpath"
        xmlns:tns="http://webservice.brandname.store.com/"
        xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
        xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
        exclude-result-prefixes="xsi xsl soap wsp wsp1_2 xsd wsu tns wsam xp20 bpws
bpel bpm ora socket mhdr oraext dvm hwf med ids xdk xref bpmn ldap">
 <xsl:param name="Invoke10_GetReceiptInformation_OutputVariable.parameters"/>
 <xsl:param name="Invoke11_GetAddressInformation_OutputVariable.parameters"/>
 <xsl:param name="Invoke12_GetOrderInformation_OutputVariable.parameters"/>
 <xsl:param name="Invoke8_GetOrderProductDetails_OutputVariable.parameters"/>
 <xsl:param name="Invoke2_InsertOrder_OutputVariable.parameters"/>
 <xsl:template match="/">
  <tns:produceReciptPDF>
   <receiptInput>
```


Πανεπιστήμιο Πειραιώς

```
      <comments>
       <xsl:value-of
select="$Invoke12_GetOrderInformation_OutputVariable.parameters/tns:GetOrderInfor
mationResponse/return/comment"/>
      </comments>
      <xsl:if test="/tns:GetUserInformationResponse/return">
       <customerInformation>
        <xsl:if test="/tns:GetUserInformationResponse/return/userPhone">
         <phone>
          <xsl:value-of select="/tns:GetUserInformationResponse/return/userPhone"/>
         </phone>
        </xsl:if>
        <postcode>
         <xsl:value-of
select="$Invoke11_GetAddressInformation_OutputVariable.parameters/tns:GetAddress
InformationResponse/return/addressPostcode"/>
        </postcode>
        <xsl:if test="/tns:GetUserInformationResponse/return/userProfession">
         <profession>
          <xsl:value-of
select="/tns:GetUserInformationResponse/return/userProfession"/>
         </profession>
        </xsl:if>
        <streedAddress>
         <xsl:value-of
select="$Invoke11_GetAddressInformation_OutputVariable.parameters/tns:GetAddress
InformationResponse/return/addressStreet"/>
        </streedAddress>
        <xsl:if test="/tns:GetUserInformationResponse/return/userTaxOffice">
         <taxOffice>
          <xsl:value-of
select="/tns:GetUserInformationResponse/return/userTaxOffice"/>
         </taxOffice>
        </xsl:if>
        <tin>
         <xsl:value-of select="/tns:GetUserInformationResponse/return/userTIN"/>
        </tin>
        <town>
         <xsl:value-of
select="$Invoke11_GetAddressInformation_OutputVariable.parameters/tns:GetAddress
InformationResponse/return/addressTown"/>
        </town>
        <userId>
         <xsl:value-of select="/tns:GetUserInformationResponse/return/userId"/>
        </userId>
        <xsl:if test="/tns:GetUserInformationResponse/return/userName">
```


Πανεπιστήμιο Πειραιώς

```
          <userName>
           <xsl:value-of
select='concat(/tns:GetUserInformationResponse/return/userName,"
",/tns:GetUserInformationResponse/return/userSurname)'/>
          </userName>
         </xsl:if>
        </customerInformation>
       </xsl:if>
       <documentInformation>
        <date>
         <xsl:value-of select='substring-
before($Invoke10_GetReceiptInformation_OutputVariable.parameters/tns:GetReceiptIn
formationResponse/return/receiptDate," ")'/>
        </date>
        <documentNumber>
         <xsl:value-of
select="$Invoke10_GetReceiptInformation_OutputVariable.parameters/tns:GetReceiptI
nformationResponse/return/receiptId"/>
        </documentNumber>
        <documentType>
         <xsl:value-of
select="$Invoke10_GetReceiptInformation_OutputVariable.parameters/tns:GetReceiptI
nformationResponse/return/receiptType"/>
        </documentType>
        <paymentMethod>
         <xsl:value-of
select="$Invoke12_GetOrderInformation_OutputVariable.parameters/tns:GetOrderInfor
mationResponse/return/paymentMethodName"/>
        </paymentMethod>
        <time>
         <xsl:value-of select='substring-
after($Invoke10_GetReceiptInformation_OutputVariable.parameters/tns:GetReceiptInfo
rmationResponse/return/receiptDate," ")'/>
        </time>
       </documentInformation>
       <orderId>
        <xsl:value-of
select="$Invoke2_InsertOrder_OutputVariable.parameters/tns:InsertOrderResponse/re
turn/orderId"/>
       </orderId>
       <xsl:for-each
select="$Invoke8_GetOrderProductDetails_OutputVariable.parameters/tns:GetOrderPro
ductDetailsResponse/return">
        <products>
         <xsl:if test="@xsi:nil">
          <xsl:attribute name="xsi:nil">
```


Πανεπιστήμιο Πειραιώς

```
            <xsl:value-of select="@xsi:nil"/>
           </xsl:attribute>
          </xsl:if>
          <productCode>
           <xsl:value-of select="productId"/>
          </productCode>
          <xsl:if test="productName">
           <productName>
            <xsl:value-of select="productName"/>
           </productName>
          </xsl:if>
          <productQuantity>
           <xsl:value-of select="productQuantity"/>
          </productQuantity>
          <unitPrice>
           <xsl:value-of select="productUnitPrice"/>
          </unitPrice>
         </products>
        </xsl:for-each>
        <recipientsName>
         <xsl:value-of
select="$Invoke11_GetAddressInformation_OutputVariable.parameters/tns:GetAddress
InformationResponse/return/addressRecipientName"/>
        </recipientsName>
        <vat>
         <xsl:value-of select="number(0.0)"/>
        </vat>
       </receiptInput>
    </tns:produceReciptPDF>
  </xsl:template>
</xsl:stylesheet>
```

Κώδικας 41: getOrderDetails.xsl

```
<?xml version="1.0" encoding="UTF-8" ?>
<?oracle-xsl-mapper
 <!-- SPECIFICATION OF MAP SOURCES AND TARGETS, DO NOT MODIFY. -->
 <mapSources>
  <source type="WSDL">
   <schema
location="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?WSDL"/
>
   <rootElement name="GetOrderProductDetailsResponse"
namespace="http://webservice.brandname.store.com/"/>
  </source>
  <source type="WSDL">
   <schema
```


Πανεπιστήμιο Πειραιώς

```
location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"/>
    <rootElement name="InsertOrderResponse"
namespace="http://webservice.brandname.store.com/"/>
    <param name="Invoke2_InsertOrder_OutputVariable.parameters" />
   </source>
 </mapSources>
 <mapTargets>
  <target type="WSDL">
    <schema location="http://10.19.149.3:8001/soa-
infra/services/thesis/OrderFulfillmentHumanTask/OrderFulfillment.wsdl"/>
    <rootElement name="payload" namespace=""/>
    <rootElementDatatype name="OrderProducts2"
namespace="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHuman
Task/OrderFulfillment"/>
   </target>
 </mapTargets>
 <!-- GENERATED BY ORACLE XSL MAPPER 11.1.1.5.0(build 110418.1550.0174) AT
[SAT FEB 18 12:13:18 EET 2012]. -->
?>
<xsl:stylesheet version="1.0"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20"
          xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
          xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
          xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
          xmlns:ora="http://schemas.oracle.com/xpath/extension"

xmlns:socket="http://www.oracle.com/XSL/Transform/java/oracle.tip.adapter.socket.
ProtocolTranslator"
          xmlns:wsp="http://www.w3.org/ns/ws-policy"

xmlns:ns0="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHumanT
ask/OrderFulfillment"

xmlns:mhdr="http://www.oracle.com/XSL/Transform/java/oracle.tip.mediator.service
.common.functions.MediatorExtnFunction"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
          xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
```


Πανεπιστήμιο Πειραιώς

```
        xmlns:med="http://schemas.oracle.com/mediator/xpath"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
        xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
        xmlns:bpmn="http://schemas.oracle.com/bpm/xpath"
        xmlns:tns="http://webservice.brandname.store.com/"
        xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
        xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
        exclude-result-prefixes="xsi xsl soap wsp wsp1_2 xsd wsu tns wsam ns0 xp20 bpws bpel bpm ora socket mhdr oraext dvm hwf med ids xdk xref bpmn ldap">
 <xsl:param name="Invoke2_InsertOrder_OutputVariable.parameters"/>
 <xsl:template match="/">
  <payload>
   <xsl:for-each select="/tns:GetOrderProductDetailsResponse/return">
    <ns0:Products>
     <ns0:ProductId>
      <xsl:value-of select="productId"/>
     </ns0:ProductId>
     <ns0:ProductName>
      <xsl:value-of select="productName"/>
     </ns0:ProductName>
     <ns0:ProductQuantity>
      <xsl:value-of select="productQuantity"/>
     </ns0:ProductQuantity>
     <ns0:UnitPrice>
      <xsl:value-of select="productUnitPrice"/>
     </ns0:UnitPrice>
     <ns0:TotalPrice>
      <xsl:value-of select="productQuantity * productUnitPrice"/>
     </ns0:TotalPrice>
    </ns0:Products>
   </xsl:for-each>
   <ns0:OrderId>
    <xsl:value-of
select="$Invoke2_InsertOrder_OutputVariable.parameters/tns:InsertOrderResponse/return/orderId"/>
   </ns0:OrderId>
  </payload>
 </xsl:template>
```


Πανεπιστήμιο Πειραιώς

```
</xsl:stylesheet>
```

Κώδικας 42: NotificationService.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns="http://xmlns.oracle.com/ias/pcbpel/NotificationService"
        targetNamespace="http://xmlns.oracle.com/ias/pcbpel/NotificationService"
        elementFormDefault="qualified" >

 <xsd:complexType name="ContentType">
  <xsd:sequence>
   <xsd:element name="MimeType" default="text/plain" type="xsd:string"
minOccurs="0"/>
   <xsd:element name="ContentBody" nillable="true" type="xsd:anyType"/>
   <xsd:element name="ContentEncoding" nillable="true" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="NotificationContextType">
  <xsd:sequence>
   <xsd:element name="compositeInstanceId" type="xsd:string" minOccurs="0"/>
   <xsd:element name="compositeDN" type="xsd:string" minOccurs="0"/>
   <xsd:element name="taskId" type="xsd:string" minOccurs="0"/>
   <xsd:element name="componentName" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="BodyPartType">
  <xsd:sequence>
   <xsd:element name="MimeType" default="text/plain" type="xsd:string"
minOccurs="0"/>
   <xsd:element name="ContentBody" nillable="true" type="xsd:anyType"/>
   <xsd:element name="BodyPartName" type="xsd:string"/>
   <xsd:element name="Disposition" default="inline" type="dispositionEnum"
minOccurs="0"/>
   <xsd:element name="ContentId" type="xsd:string" minOccurs="0"/>
   <xsd:element name="ContentEncoding" nillable="true"
type="contentEncodingEnum"/>
   <xsd:element name="AttachmentContentEnclosed" default="true"
type="xsd:boolean" minOccurs="0"/>
  </xsd:sequence>
 </xsd:complexType>

 <!-- Restricted to subset of section 6.1 of RFC 2045 -->
 <xsd:simpleType name="contentEncodingEnum">
   <xsd:restriction base="xsd:string">
```


Πανεπιστήμιο Πειραιώς

```
      <xsd:enumeration value="7bit"/>
      <xsd:enumeration value="base64"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- Restricted to subset of section 2 of RFC 2183 -->
  <xsd:simpleType name="dispositionEnum">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="inline"/>
      <xsd:enumeration value="attachment"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:element name="MultiPart" type="MultiPartType"/>

  <xsd:complexType name="MultiPartType">
   <xsd:sequence>
    <xsd:element name="BodyPart" type="BodyPartType" maxOccurs="unbounded"/>
   </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="IMPayloadType" >
   <xsd:sequence>
    <xsd:element name="To" type="xsd:string" minOccurs="0"/>
    <xsd:element name="Content" type="ContentType"/>
    <xsd:element name="NotificationContext" type="NotificationContextType"/>
   </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="FaxPayloadType">
   <xsd:sequence>
    <xsd:element name="To" type="xsd:string" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="Content" type="ContentType"/>
    <xsd:element name="CoverPageName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="NotificationContext" type="NotificationContextType"/>
   </xsd:sequence>
  </xsd:complexType>

  <xsd:element name="FaxPayload" type="FaxPayloadType"/>

  <xsd:complexType name="VoicePayloadType" >
   <xsd:sequence>
    <xsd:element name="To" type="xsd:string" minOccurs="0"/>
    <xsd:element name="Content" type="ContentType"/>
    <xsd:element name="NotificationContext" type="NotificationContextType"/>
   </xsd:sequence>
```


Πανεπιστήμιο Πειραιώς

```
 </xsd:complexType>

 <xsd:element name="VoicePayload" type="VoicePayloadType"/>

 <xsd:complexType name="SMSPayloadType">
  <xsd:sequence>
   <xsd:element name="From" type="xsd:string" minOccurs="0"/>
   <xsd:element name="To" type="xsd:string" minOccurs="0"/>
   <xsd:element name="Subject" type="xsd:string"/>
   <xsd:element name="Content" type="ContentType"/>
   <xsd:element name="NotificationContext" type="NotificationContextType"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:element name="SMSPayload" type="SMSPayloadType"/>

 <xsd:complexType name="PagerPayloadType">
  <xsd:sequence>
   <xsd:element name="From" type="xsd:string" minOccurs="0"/>
   <xsd:element name="To" type="xsd:string" minOccurs="0"/>
   <xsd:element name="Content" type="ContentType"/>
   <xsd:element name="NotificationContext" type="NotificationContextType"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:element name="PagerPayload" type="PagerPayloadType"/>

 <xsd:complexType name="URIPayloadType">
  <xsd:sequence>
   <xsd:element name="To" type="xsd:string" minOccurs="1"/>
   <xsd:element name="Subject" type="xsd:string"/>
   <xsd:element name="UserId" type="xsd:string" minOccurs="1"/>
   <xsd:element name="DisplayText" type="xsd:string" minOccurs="1"/>
   <xsd:element name="URILocation" type="xsd:string" minOccurs="1"/>
   <xsd:element name="CreatedDate" type="xsd:dateTime" minOccurs="1"/>
   <xsd:element name="Content" type="ContentType"/>
   <xsd:element name="NotificationContext" type="NotificationContextType"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:element name="URIPayload" type="URIPayloadType"/>

 <xsd:complexType name="EmailPayloadType">
  <xsd:sequence>
   <xsd:element name="FromAccountName" minOccurs="0" type="xsd:string"/>
   <xsd:element name="To" type="xsd:string" minOccurs="0"/>
```


Πανεπιστήμιο Πειραιώς

```
    <xsd:element name="ReplyToAddress" minOccurs="0" type="xsd:string"/>
    <xsd:element name="Subject" type="xsd:string"/>
    <xsd:element name="Content" type="ContentType"/>
    <xsd:element name="EmailHeaders" type="ArrayOfEmailHeaderType"
minOccurs="0"/>
    <xsd:element name="Cc" minOccurs="0" type="xsd:string"/>
    <xsd:element name="Bcc" minOccurs="0" type="xsd:string"/>
    <xsd:element name="NotificationContext" type="NotificationContextType"/>
   </xsd:sequence>
 </xsd:complexType>

 <xsd:element name="EmailPayload" type="EmailPayloadType"/>

 <xsd:complexType name="UserNotificationPayloadType">
  <xsd:sequence>
   <xsd:element name="UserId" type="xsd:string"/>
   <xsd:element type="GenericPayloadType" name="GenericPayload"/>
   <xsd:element name="PreferenceProperties"
          type="UserPropertiesType"
          minOccurs="0" maxOccurs="1"/>
   <xsd:element name="NotificationContext" type="NotificationContextType"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="GroupNotificationPayloadType">
  <xsd:sequence>
   <xsd:element name="GroupId" type="xsd:string"/>
   <xsd:element name="GenericPayload" type="GenericPayloadType"/>
   <xsd:element name="NotificationContext" type="NotificationContextType"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="EmailHeaderType">
  <xsd:sequence>
   <xsd:element name="HeaderName" type="xsd:string"/>
   <xsd:element name="HeaderValue" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="ArrayOfEmailHeaderType">
  <xsd:sequence maxOccurs="1">
   <xsd:element name="EmailHeader" maxOccurs="unbounded"
type="EmailHeaderType"/>
  </xsd:sequence>
 </xsd:complexType>
```


Πανεπιστήμιο Πειραιώς

```
<xsd:complexType name="GenericPayloadType">
 <xsd:sequence>
  <xsd:element ref="FaxPayload" minOccurs="0"/>
  <xsd:element ref="VoicePayload" minOccurs="0"/>
  <xsd:element ref="SMSPayload" minOccurs="0"/>
  <xsd:element ref="PagerPayload" minOccurs="0"/>
  <xsd:element ref="URIPayload" minOccurs="0"/>
  <xsd:element ref="EmailPayload" minOccurs="0"/>
  <xsd:element ref="CommonPayload" minOccurs="0"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:element name="GenericPayload" type="GenericPayloadType"/>

<xsd:complexType name="ResponseType">
 <xsd:sequence>
  <xsd:element name="MessageId" type="xsd:string"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ArrayOfResponseType">
 <xsd:sequence>
  <xsd:element name="Response" type="ResponseType" maxOccurs="unbounded"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CommonPayloadType">
 <xsd:sequence>
  <xsd:element name="To" type="xsd:string" minOccurs="0"/>
  <xsd:element name="From" type="xsd:string" minOccurs="0"/>
  <xsd:element name="Subject" type="xsd:string"/>
  <xsd:element name="Content" type="ContentType"/>
  <xsd:element name="NotificationContext" type="NotificationContextType"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:element name="CommonPayload" type="CommonPayloadType"/>

<xsd:complexType name="UserPropertiesType">
 <xsd:sequence>
  <xsd:element name="Parameter" type="UserPropertiesParameterType"
minOccurs="1" maxOccurs="unbounded"/>
 </xsd:sequence>
</xsd:complexType>

 <xsd:complexType name="UserPropertiesParameterType">
```


Πανεπιστήμιο Πειραιώς

```
  <xsd:sequence>
   <xsd:element name="Name" type="xsd:string"/>
   <xsd:element name="Value" type="xsd:anyType"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="TaskNotificationPayloadType">
  <xsd:sequence>
   <xsd:element name="TaskId" type="xsd:string" minOccurs="1"/>
   <xsd:element name="TaskVersion" type="xsd:integer" minOccurs="1"/>
   <xsd:element name="TaskAction" type="xsd:string" minOccurs="1"/>
   <xsd:element name="NotificationRecipientType" type="xsd:string" minOccurs="1"/>
   <xsd:element name="IdForActionableLink" type="xsd:string" minOccurs="1"/>
   <xsd:element name="ResponseAction" type="xsd:string" minOccurs="1"/>
   <xsd:element name="ActionMessage" type="xsd:string" minOccurs="0"/>
   <xsd:element name="Actionable" type="xsd:boolean" minOccurs="1"/>
   <xsd:element name="IncludeTaskAttachments" type="xsd:boolean"
minOccurs="1"/>
   <xsd:element name="SecureNotifications" type="xsd:boolean" minOccurs="1"/>
   <xsd:element name="HideWorklistUrlInEmail" type="xsd:boolean" minOccurs="1"/>
   <xsd:element name="WordMLXSLT" type="xsd:string" minOccurs="1"/>
   <xsd:element name="Xslt" type="xsd:string" minOccurs="1"/>
   <xsd:element name="PreferencePropertyNames" type="xsd:string"
minOccurs="0"/>
   <xsd:element name="PreferencePropertyValues" type="xsd:string" minOccurs="0"/>
   <xsd:element name="NotificationContext" type="NotificationContextType"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:element name="TaskNotificationPayload" type="TaskNotificationPayloadType"/>

</xsd:schema>
```

Κώδικας 43: composite.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 1.0 at [2/12/12 11:36 AM]. -->
<composite name="BuyingProcessProject1"
        revision="1.0"
        label="2012-02-12_11-36-23_077"
        mode="active"
        state="on"
        xmlns="http://xmlns.oracle.com/sca/1.0"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
```


Πανεπιστήμιο Πειραιώς

```
        xmlns:ui="http://xmlns.oracle.com/soa/designer/">
 <import
namespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessPr
oject1/OrderProcess"
       location="OrderProcess.wsdl" importType="wsdl"/>
 <import namespace="http://xmlns.oracle.com/ias/pcbpel/NotificationService"
       location="NotificationService.wsdl" importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"

location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"
       importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"
       location="RepositoryWSWrapper.wsdl" importType="wsdl"/>
 <import
namespace="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway"
       location="http://10.19.149.3:8001/soa-
infra/services/thesis/CreditCardGateway/CreditCardGateway.wsdl"
       importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"

location="http://176.34.182.139:8888/ThesisMailServices/OrderEmailGeneratorWS?
WSDL"
       importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"
       location="OrderEmailGeneratorWSWrapper.wsdl" importType="wsdl"/>
 <import
namespace="http://xmlns.oracle.com/BankPaymentV02/Project1/BankVerificationPro
cess"
       location="http://10.19.149.3:8001/soa-
infra/services/thesis/Project1/BankVerificationProcess.wsdl"
       importType="wsdl"/>
 <import
namespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAnd
Suppliers/ProcessOrderAndSupplier"
       location="http://10.19.149.3:8001/soa-
infra/services/thesis/ProcessOrderAndSuppliers/ProcessOrderAndSupplier.wsdl"
       importType="wsdl"/>
 <import
namespace="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHuman
Task/OrderFulfillment"
       location="http://10.19.149.3:8001/soa-
infra/services/thesis/OrderFulfillmentHumanTask/OrderFulfillment.wsdl"
       importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"

location="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?WSDL"
```


Πανεπιστήμιο Πειραιώς

```
      importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"
      location="Repository1WSWrapper.wsdl" importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"

location="http://176.34.182.139:8888/ThesisInvoceService/InvoicePDFDocumentWS?
WSDL"
      importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"
      location="InvoicePDFDocumentWSWrapper.wsdl" importType="wsdl"/>
 <import namespace="http://services.web.shippingcompany.com/"

location="http://176.34.182.139:8888/ThesisShippingServices/ShippingCompanyWS?
WSDL"
      importType="wsdl"/>
 <import namespace="http://services.web.shippingcompany.com/"
      location="ShippingCompanyWSWrapper.wsdl" importType="wsdl"/>
 <service name="orderprocess_client_ep" ui:wsdlLocation="OrderProcess.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessProje
ct1/OrderProcess#wsdl.interface(OrderProcess)"/>
  <binding.ws
port="http://xmlns.oracle.com/BuyingProcessApplicationV01/BuyingProcessProject1/
OrderProcess#wsdl.endpoint(orderprocess_client_ep/OrderProcess_pt)"/>
 </service>
 <component name="OrderProcess" version="2.0">
  <implementation.bpel src="OrderProcess.bpel"/>
 </component>
 <reference name="NotificationService1"
       ui:wsdlLocation="NotificationService.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.interface(Noti
ficationService)"/>
  <binding.wsif
port="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.endpoint(Notificat
ionService/JavaPort)"
          location="NotificationService.wsdl"/>
 </reference>
 <reference name="Repository"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WS
DL">
  <interface.wsdl
interface="http://webservice.brandname.store.com/#wsdl.interface(RepositoryWS)"/>
  <binding.ws
port="http://webservice.brandname.store.com/#wsdl.endpoint(RepositoryWS/Reposit
```


Πανεπιστήμιο Πειραιώς

```
oryWSPort)"

location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="CreditCardGateway"
        ui:wsdlLocation="http://10.19.149.3:8001/soa-
infra/services/thesis/CreditCardGateway/CreditCardGateway.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway#wsdl.inte
rface(CreditCardGateway)"/>
  <binding.ws
port="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway#wsdl.endpoin
t(creditcardgateway_client_ep/CreditCardGateway_pt)"
          location="http://thesisnoip.no-ip.org:8001/soa-
infra/services/thesis/CreditCardGateway/creditcardgateway_client_ep?WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="OrderMailContentGenerator"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisMailServices/OrderEmailGenerat
orWS?WSDL">
  <interface.wsdl
interface="http://webservice.brandname.store.com/#wsdl.interface(OrderEmailGenera
torWS)"/>
  <binding.ws
port="http://webservice.brandname.store.com/#wsdl.endpoint(OrderEmailGenerator
WS/OrderEmailGeneratorWSPort)"

location="http://176.34.182.139:8888/ThesisMailServices/OrderEmailGeneratorWS?
WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="NotificationService2"
        ui:wsdlLocation="NotificationService.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.interface(Noti
```


Πανεπιστήμιο Πειραιώς

```
ficationService)"/>
   <binding.wsif
port="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.endpoint(Notificat
ionService/JavaPort)"
            location="NotificationService.wsdl"/>
 </reference>
 <reference name="BankVerification"
         ui:wsdlLocation="http://10.19.149.3:8001/soa-
infra/services/thesis/Project1/BankVerificationProcess.wsdl">
   <interface.wsdl
interface="http://xmlns.oracle.com/BankPaymentV02/Project1/BankVerificationProce
ss#wsdl.interface(BankVerificationProcess)"

callbackInterface="http://xmlns.oracle.com/BankPaymentV02/Project1/BankVerificati
onProcess#wsdl.interface(BankVerificationProcessCallback)"/>
   <binding.ws
port="http://xmlns.oracle.com/BankPaymentV02/Project1/BankVerificationProcess#
wsdl.endpoint(bankverificationprocess_client_ep/BankVerificationProcess_pt)"
           location="http://thesisnoip.no-ip.org:8001/soa-
infra/services/thesis/Project1/bankverificationprocess_client_ep?WSDL"
           soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
           type="xs:string" many="false">WSDLDriven</property>
   </binding.ws>
   <callback>
    <binding.ws
port="http://xmlns.oracle.com/BankPaymentV02/Project1/BankVerificationProcess#
wsdl.endpoint(bankverificationprocess_client_ep/BankVerificationProcessCallback_pt)"
/>
   </callback>
 </reference>
 <reference name="ProcessOrderAndSupply"
         ui:wsdlLocation="http://10.19.149.3:8001/soa-
infra/services/thesis/ProcessOrderAndSuppliers/ProcessOrderAndSupplier.wsdl">
   <interface.wsdl
interface="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAndSu
ppliers/ProcessOrderAndSupplier#wsdl.interface(ProcessOrderAndSupplier)"/>
   <binding.ws
port="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAndSuppli
ers/ProcessOrderAndSupplier#wsdl.endpoint(processorderandsupplier_client_ep/Proc
essOrderAndSupplier_pt)"
           location="http://thesisnoip.no-ip.org:8001/soa-
infra/services/thesis/ProcessOrderAndSuppliers/processorderandsupplier_client_ep?
WSDL"
           soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
```


Πανεπιστήμιο Πειραιώς

```
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="OrderFulfillmentHT"
        ui:wsdlLocation="http://10.19.149.3:8001/soa-
infra/services/thesis/OrderFulfillmentHumanTask/OrderFulfillment.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHumanTa
sk/OrderFulfillment#wsdl.interface(OrderFulfillment)"

callbackInterface="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentH
umanTask/OrderFulfillment#wsdl.interface(OrderFulfillmentCallback)"/>
  <binding.ws
port="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHumanTask/O
rderFulfillment#wsdl.endpoint(orderfulfillment_client_ep/OrderFulfillment_pt)"
          location="http://thesisnoip.no-ip.org:8001/soa-
infra/services/thesis/OrderFulfillmentHumanTask/orderfulfillment_client_ep?WSDL"
          soapVersion="1.1">
   <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
  <callback>
   <binding.ws
port="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHumanTask/O
rderFulfillment#wsdl.endpoint(orderfulfillment_client_ep/OrderFulfillmentCallback_pt)
"/>
  </callback>
 </reference>
 <reference name="Repository1"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?
WSDL">
  <interface.wsdl
interface="http://webservice.brandname.store.com/#wsdl.interface(Repository1WS)"/
>
  <binding.ws
port="http://webservice.brandname.store.com/#wsdl.endpoint(Repository1WS/Repos
itory1WSPort)"

location="http://176.34.182.139:8888/ThesisOrderServices1/Repository1WS?WSDL"
          soapVersion="1.1">
   <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="InvoiceGenerator"
```


Πανεπιστήμιο Πειραιώς

```
ui:wsdlLocation="http://176.34.182.139:8888/ThesisInvoceService/InvoicePDFDocum
entWS?WSDL">
  <interface.wsdl
interface="http://webservice.brandname.store.com/#wsdl.interface(InvoicePDFDocum
entWS)"/>
  <binding.ws
port="http://webservice.brandname.store.com/#wsdl.endpoint(InvoicePDFDocument
WS/InvoicePDFDocumentWSPort)"

location="http://176.34.182.139:8888/ThesisInvoceService/InvoicePDFDocumentWS?
WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="ShippingCompany"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisShippingServices/ShippingCompa
nyWS?WSDL">
  <interface.wsdl
interface="http://services.web.shippingcompany.com/#wsdl.interface(ShippingCompa
nyWS)"/>
  <binding.ws
port="http://services.web.shippingcompany.com/#wsdl.endpoint(ShippingCompanyW
S/ShippingCompanyWSPort)"

location="http://176.34.182.139:8888/ThesisShippingServices/ShippingCompanyWS?
WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="NotificationService4"
        ui:wsdlLocation="NotificationService.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.interface(Noti
ficationService)"/>
  <binding.wsif
port="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.endpoint(Notificat
ionService/JavaPort)"
           location="NotificationService.wsdl"/>
 </reference>
 <wire>
```


Πανεπιστήμιο Πειραιώς

```
  <source.uri>orderprocess_client_ep</source.uri>
  <target.uri>OrderProcess/orderprocess_client</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/NotificationService1</source.uri>
  <target.uri>NotificationService1</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/Repository</source.uri>
  <target.uri>Repository</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/CreditCardGateway</source.uri>
  <target.uri>CreditCardGateway</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/OrderMailContentGenerator</source.uri>
  <target.uri>OrderMailContentGenerator</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/NotificationService2</source.uri>
  <target.uri>NotificationService2</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/BankVerification</source.uri>
  <target.uri>BankVerification</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/ProcessOrderAndSupply</source.uri>
  <target.uri>ProcessOrderAndSupply</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/OrderFulfillmentHT</source.uri>
  <target.uri>OrderFulfillmentHT</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/Repository1</source.uri>
  <target.uri>Repository1</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/InvoiceGenerator</source.uri>
  <target.uri>InvoiceGenerator</target.uri>
 </wire>
 <wire>
  <source.uri>OrderProcess/ShippingCompany</source.uri>
  <target.uri>ShippingCompany</target.uri>
```


Πανεπιστήμιο Πειραιώς

```
 </wire>
 <wire>
  <source.uri>OrderProcess/NotificationService4</source.uri>
  <target.uri>NotificationService4</target.uri>
 </wire>
</composite>
```

6.8 Κώδικας διαδικασίας BPEL επεξεργασίας παραγγελίας και παραγγελίας προϊόντων από προμηθευτές

Κώδικας 44: ProcessOrderAndSupplier.bpel

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<!--
////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
 Oracle JDeveloper BPEL Designer

 Created: Thu Feb 16 12:48:59 EET 2012
 Author:  Epameinondas Nikolopoulos
 Type: BPEL 2.0 Process
 Purpose: Synchronous BPEL Process
////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
-->
<process name="ProcessOrderAndSupplier"

targetNamespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAndSuppliers/ProcessOrderAndSupplier"
         xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"

xmlns:client="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAndSuppliers/ProcessOrderAndSupplier"
         xmlns:ora="http://schemas.oracle.com/xpath/extension"
         xmlns:bpelx="http://schemas.oracle.com/bpel/extension"
      xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
      xmlns:ns1="http://webservice.brandname.store.com/"
      xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.Xpath20"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.ExtFunc"
```


Πανεπιστήμιο Πειραιώς

```
xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
      xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
      xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
      xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
      xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions"
      xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:ns2="http://webservice.supplier1.com/"
      xmlns:ns3="http://webservice.supplier2.com/"

xmlns:ns4="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHuma
nTask/InsertSupplierOrderWhenIsReceived">

  <import
namespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAnd
Suppliers/ProcessOrderAndSupplier" location="ProcessOrderAndSupplier.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
  <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
     PARTNERLINKS
     List of services participating in this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
  -->
 <partnerLinks>
  <!--
    The 'client' role represents the requester of this service. It is
    used for callback. The location and correlation information associated
    with the client role are automatically set using WS-Addressing.
  -->
  <partnerLink name="processorderandsupplier_client"
partnerLinkType="client:ProcessOrderAndSupplier"
myRole="ProcessOrderAndSupplierProvider"/>
  <partnerLink name="Repository" partnerLinkType="ns1:Repository"
          partnerRole="RepositoryWS"/>
  <partnerLink name="SupplierA" partnerLinkType="ns2:SupplierA"
        partnerRole="Supplier1WS"/>
  <partnerLink name="SupplierB" partnerLinkType="ns3:SupplierB"
```


Πανεπιστήμιο Πειραιώς

```
            partnerRole="Supplier2WS"/>
   <partnerLink name="SupplierOrderSubmitHumanTask"
            partnerLinkType="ns4:InsertSupplierOrderWhenIsReceived"
            partnerRole="InsertSupplierOrderWhenIsReceivedProvider"/>
  </partnerLinks>

  <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
     VARIABLES
     List of messages and XML documents used within this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
  -->
  <variables>
   <!-- Reference to the message passed as input during initiation -->
   <variable name="inputVariable"
messageType="client:ProcessOrderAndSupplierRequestMessage"/>

   <!-- Reference to the message that will be returned to the requester-->
   <variable name="outputVariable"
messageType="client:ProcessOrderAndSupplierResponseMessage"/>
   <variable name="Invoke1_AvailabilityAndCommitment_InputVariable"
         messageType="ns1:AvailabilityAndCommitment"/>
   <variable name="Invoke1_AvailabilityAndCommitment_OutputVariable"
         messageType="ns1:AvailabilityAndCommitmentResponse"/>
   <variable name="SupplierA_OrderSubmit_InputVariable"
         messageType="ns2:OrderSubmit"/>
   <variable name="SupplierA_OrderSubmit_OutputVariable"
         messageType="ns2:OrderSubmitResponse"/>
   <variable name="SupplierB_OrderSubmit_InputVariable"
         messageType="ns3:OrderSubmit"/>
   <variable name="SupplierB_OrderSubmit_OutputVariable"
         messageType="ns3:OrderSubmitResponse"/>
   <variable name="updateOrderWithSupplierOrderID_InputVariable"
         messageType="ns1:updateOrderWithSupplierOrder"/>
   <variable name="updateOrderWithSupplierOrderID_OutputVariable"
         messageType="ns1:updateOrderWithSupplierOrderResponse"/>
   <variable name="SendSupplierOrder_InputVariable"
         messageType="ns4:InsertSupplierOrderWhenIsReceivedRequestMessage"/>
  </variables>

  <!--
```


Πανεπιστήμιο Πειραιώς

```
//////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////
   ORCHESTRATION LOGIC
   Set of activities coordinating the flow of messages across the
   services integrated within this business process

//////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////
 -->
 <sequence name="main">

  <!-- Receive input from requestor. (Note: This maps to operation defined in
ProcessOrderAndSupplier.wsdl) -->
  <receive name="receiveInput" partnerLink="processorderandsupplier_client"
portType="client:ProcessOrderAndSupplier"
        operation="processOrderAndSuppliers" variable="inputVariable"
createInstance="yes"/>

  <!-- Generate reply to synchronous request -->
  <forEach parallel="no" counterName="Counter" name="ForEachProduct">
   <startCounterValue>1</startCounterValue>

<finalCounterValue>count($inputVariable.payload/client:Products)</finalCounterValue
>
    <scope name="Scope1">
     <variables>
      <variable name="BooleanVariable1" type="xsd:boolean"/>
     </variables>
     <sequence>
      <assign name="RepositoryCheckInput">
       <copy>

<from>$inputVariable.payload/client:Products[position()=$Counter]/client:ProductId<
/from>

<to>$Invoke1_AvailabilityAndCommitment_InputVariable.parameters/productToOrder
/productId</to>
       </copy>
       <copy>

<from>$inputVariable.payload/client:Products[position()=$Counter]/client:Quantity</f
rom>

<to>$Invoke1_AvailabilityAndCommitment_InputVariable.parameters/productToOrder
/productQuantity</to>
       </copy>
```


Πανεπιστήμιο Πειραιώς

```
        <copy>
         <from>$inputVariable.payload/client:OrderId</from>

<to>$Invoke1_AvailabilityAndCommitment_InputVariable.parameters/productToOrder
/orderId</to>
        </copy>
       </assign>
       <invoke name="CheckRepository" partnerLink="Repository"
            portType="ns1:RepositoryWS"
            operation="AvailabilityAndCommitment"
            inputVariable="Invoke1_AvailabilityAndCommitment_InputVariable"
            outputVariable="Invoke1_AvailabilityAndCommitment_OutputVariable"
            bpelx:invokeAsDetail="no"/>
       <assign name="RepositoryCheckOutput">
        <copy>

<from>$Invoke1_AvailabilityAndCommitment_OutputVariable.parameters/return/need
SupplierStatus</from>
          <to>$BooleanVariable1</to>
        </copy>
       </assign>
       <if name="IsThereANeedForSupplier">
        <documentation>NeedsSupplier</documentation>
        <condition>$BooleanVariable1</condition>
        <sequence name="Sequence1">
         <if name="ChooseSupplier">
          <documentation>SupplierA</documentation>

<condition>($Invoke1_AvailabilityAndCommitment_OutputVariable.parameters/return
/supplierId=1)</condition>
           <sequence name="Sequence2">
            <assign name="SubmitOrderInput1">
             <copy>

<from>$Invoke1_AvailabilityAndCommitment_OutputVariable.parameters/return/supp
lierProductId</from>

<to>$SupplierA_OrderSubmit_InputVariable.parameters/order/productId</to>
             </copy>
             <copy>

<from>$Invoke1_AvailabilityAndCommitment_OutputVariable.parameters/return/prod
uctQuantity</from>

<to>$SupplierA_OrderSubmit_InputVariable.parameters/order/productQuantity</to>
             </copy>
```


Πανεπιστήμιο Πειραιώς

```
            <copy>
             <from>number(768989)</from>

<to>$SupplierA_OrderSubmit_InputVariable.parameters/order/partnerId</to>
            </copy>
           </assign>
           <invoke name="SubmitOrderToSupplierA"
                partnerLink="SupplierA" portType="ns2:Supplier1WS"
                operation="OrderSubmit"
                inputVariable="SupplierA_OrderSubmit_InputVariable"
                outputVariable="SupplierA_OrderSubmit_OutputVariable"
                bpelx:invokeAsDetail="no"/>
           <flow name="Flow1">
            <sequence name="Sequence">
             <assign name="SupplierOrderIdAndPriceInput1">
              <copy>

<from>$SupplierA_OrderSubmit_OutputVariable.parameters/return/orderId</from>

<to>$SendSupplierOrder_InputVariable.payload/ns4:SupplierOrderId</to>
              </copy>
              <copy>

<from>$SupplierA_OrderSubmit_OutputVariable.parameters/return/orderPrice</from
>

<to>$SendSupplierOrder_InputVariable.payload/ns4:SupplierOrderPrice</to>
              </copy>
             </assign>
             <invoke name="OneWayVerificationOfPhysicalSupplierOrderReceival1"
                  partnerLink="SupplierOrderSubmitHumanTask"
                  portType="ns4:InsertSupplierOrderWhenIsReceived"
                  operation="SendSupplierOrder"
                  inputVariable="SendSupplierOrder_InputVariable"
                  bpelx:invokeAsDetail="no"/>
           </sequence>
           <sequence name="Sequence4">
            <assign name="SubmitOrderOutpoutAndUpdateInput1">
             <copy>

<from>$SupplierA_OrderSubmit_OutputVariable.parameters/return/orderId</from>

<to>$updateOrderWithSupplierOrderID_InputVariable.parameters/SupplierOrderId</t
o>
              </copy>
              <copy>
```


Πανεπιστήμιο Πειραιώς

```
<from>$inputVariable.payload/client:Products[position()=$Counter]/client:ProductId<
/from>

<to>$updateOrderWithSupplierOrderID_InputVariable.parameters/ProductToOrder/pr
oductId</to>
              </copy>
              <copy>
               <from>$inputVariable.payload/client:OrderId</from>

<to>$updateOrderWithSupplierOrderID_InputVariable.parameters/ProductToOrder/or
derId</to>
              </copy>
             </assign>
             <invoke name="UpdateProductsToOrder1"
                  partnerLink="Repository"
                  portType="ns1:RepositoryWS"
                  operation="updateOrderWithSupplierOrder"
                  inputVariable="updateOrderWithSupplierOrderID_InputVariable"
                  outputVariable="updateOrderWithSupplierOrderID_OutputVariable"
                  bpelx:invokeAsDetail="no"/>
            </sequence>
           </flow>
          </sequence>
          <else>
           <documentation>SupplierB</documentation>
           <sequence name="Sequence3">
            <assign name="SubmitOrderInput1">
             <copy>

<from>$Invoke1_AvailabilityAndCommitment_OutputVariable.parameters/return/supp
lierProductId</from>

<to>$SupplierB_OrderSubmit_InputVariable.parameters/order/productId</to>
              </copy>
              <copy>

<from>$Invoke1_AvailabilityAndCommitment_OutputVariable.parameters/return/prod
uctQuantity</from>

<to>$SupplierB_OrderSubmit_InputVariable.parameters/order/productQuantity</to>
              </copy>
              <copy>
               <from>number(890754)</from>

<to>$SupplierB_OrderSubmit_InputVariable.parameters/order/partnerId</to>
```


Πανεπιστήμιο Πειραιώς

```
            </copy>
           </assign>
           <invoke name="SubmitOrderToSupplierB"
                partnerLink="SupplierB" portType="ns3:Supplier2WS"
                operation="OrderSubmit"
                inputVariable="SupplierB_OrderSubmit_InputVariable"
                outputVariable="SupplierB_OrderSubmit_OutputVariable"
                bpelx:invokeAsDetail="no"/>
           <flow name="Flow2">
            <sequence name="Sequence">
             <assign name="SupplierOrderIdAndPriceInput2">
              <copy>

<from>$SupplierB_OrderSubmit_OutputVariable.parameters/return/orderId</from>

<to>$SendSupplierOrder_InputVariable.payload/ns4:SupplierOrderId</to>
              </copy>
              <copy>

<from>$SupplierB_OrderSubmit_OutputVariable.parameters/return/orderPrice</from
>

<to>$SendSupplierOrder_InputVariable.payload/ns4:SupplierOrderPrice</to>
              </copy>
             </assign>
             <invoke name="OneWayVerificationOfPhysicalSupplierOrderReceival2"
                  partnerLink="SupplierOrderSubmitHumanTask"
                  portType="ns4:InsertSupplierOrderWhenIsReceived"
                  operation="SendSupplierOrder"
                  inputVariable="SendSupplierOrder_InputVariable"
                  bpelx:invokeAsDetail="no"/>
            </sequence>
            <sequence name="Sequence5">
             <assign name="SubmitOrderOutpoutAndUpdateInput2">
              <copy>

<from>$inputVariable.payload/client:Products[position()=$Counter]/client:ProductId<
/from>

<to>$updateOrderWithSupplierOrderID_InputVariable.parameters/ProductToOrder/pr
oductId</to>
              </copy>
              <copy>

<from>$SupplierB_OrderSubmit_OutputVariable.parameters/return/orderId</from>
```


Πανεπιστήμιο Πειραιώς

```
<to>$updateOrderWithSupplierOrderID_InputVariable.parameters/SupplierOrderId</t
o>
                </copy>
                <copy>
                 <from>$inputVariable.payload/client:OrderId</from>

<to>$updateOrderWithSupplierOrderID_InputVariable.parameters/ProductToOrder/or
derId</to>
                </copy>
               </assign>
               <invoke name="UpdateProductsToOrder2"
                    partnerLink="Repository"
                    portType="ns1:RepositoryWS"
                    operation="updateOrderWithSupplierOrder"
                    inputVariable="updateOrderWithSupplierOrderID_InputVariable"
                    outputVariable="updateOrderWithSupplierOrderID_OutputVariable"
                    bpelx:invokeAsDetail="no"/>
             </sequence>
            </flow>
           </sequence>
          </else>
         </if>
        </sequence>
        <else>
         <documentation>Do Nothing</documentation>
         <empty name="DoNothing"/>
        </else>
       </if>
     </sequence>
    </scope>
  </forEach>
  <reply name="replyOutputNull" partnerLink="processorderandsupplier_client"
portType="client:ProcessOrderAndSupplier"
       operation="processOrderAndSuppliers" variable="outputVariable"/>
 </sequence>
</process>
```

Κώδικας 45: Composite.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 1.0 at [2/16/12 11:42 AM]. -->
<composite name="ProcessOrderAndSuppliers"
       revision="1.0"
       label="2012-02-16_11-42-59_514"
```


Πανεπιστήμιο Πειραιώς

```
        mode="active"
        state="on"
        xmlns="http://xmlns.oracle.com/sca/1.0"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
        xmlns:ui="http://xmlns.oracle.com/soa/designer/">
 <import
namespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAnd
Suppliers/ProcessOrderAndSupplier"
       location="ProcessOrderAndSupplier.wsdl" importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"

location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"
       importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"
       location="RepositoryWSWrapper.wsdl" importType="wsdl"/>
 <import namespace="http://webservice.supplier1.com/"

location="http://176.34.182.139:8888/ThesisSuppliersServices/Supplier1WS?WSDL"
       importType="wsdl"/>
 <import namespace="http://webservice.supplier2.com/"

location="http://176.34.182.139:8888/ThesisSuppliersServices/Supplier2WS?WSDL"
       importType="wsdl"/>
 <import namespace="http://webservice.supplier1.com/"
       location="Supplier1WSWrapper.wsdl" importType="wsdl"/>
 <import namespace="http://webservice.supplier2.com/"
       location="Supplier2WSWrapper.wsdl" importType="wsdl"/>
 <import
namespace="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHum
anTask/InsertSupplierOrderWhenIsReceived"
       location="http://10.19.149.3:8001/soa-
infra/services/thesis/SupplierOrderHumanTask/InsertSupplierOrderWhenIsReceived.
wsdl"
       importType="wsdl"/>
 <service name="processorderandsupplier_client_ep"
        ui:wsdlLocation="ProcessOrderAndSupplier.wsdl">
   <interface.wsdl
interface="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAndSu
ppliers/ProcessOrderAndSupplier#wsdl.interface(ProcessOrderAndSupplier)"/>
    <binding.ws
port="http://xmlns.oracle.com/BuyingProcessApplicationV01/ProcessOrderAndSuppli
ers/ProcessOrderAndSupplier#wsdl.endpoint(processorderandsupplier_client_ep/Proc
essOrderAndSupplier_pt)"/>
 </service>
```


Πανεπιστήμιο Πειραιώς

```
 <component name="ProcessOrderAndSupplier" version="2.0">
  <implementation.bpel src="ProcessOrderAndSupplier.bpel"/>
 </component>
 <reference name="Repository"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WS
DL">
  <interface.wsdl
interface="http://webservice.brandname.store.com/#wsdl.interface(RepositoryWS)"/>
  <binding.ws
port="http://webservice.brandname.store.com/#wsdl.endpoint(RepositoryWS/Reposit
oryWSPort)"

location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="SupplierA"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisSuppliersServices/Supplier1WS?
WSDL">
  <interface.wsdl
interface="http://webservice.supplier1.com/#wsdl.interface(Supplier1WS)"/>
  <binding.ws
port="http://webservice.supplier1.com/#wsdl.endpoint(Supplier1WS/Supplier1WSPor
t)"

location="http://176.34.182.139:8888/ThesisSuppliersServices/Supplier1WS?WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="SupplierB"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisSuppliersServices/Supplier2WS?
WSDL">
  <interface.wsdl
interface="http://webservice.supplier2.com/#wsdl.interface(Supplier2WS)"/>
  <binding.ws
port="http://webservice.supplier2.com/#wsdl.endpoint(Supplier2WS/Supplier2WSPor
t)"

location="http://176.34.182.139:8888/ThesisSuppliersServices/Supplier2WS?WSDL"
```


Πανεπιστήμιο Πειραιώς

```
            soapVersion="1.1">
     <property name="weblogic.wsee.wsat.transaction.flowOption"
            type="xs:string" many="false">WSDLDriven</property>
   </binding.ws>
  </reference>
  <reference name="SupplierOrderSubmitHumanTask"
          ui:wsdlLocation="http://10.19.149.3:8001/soa-
infra/services/thesis/SupplierOrderHumanTask/InsertSupplierOrderWhenIsReceived.
wsdl">
   <interface.wsdl
interface="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHuman
Task/InsertSupplierOrderWhenIsReceived#wsdl.interface(InsertSupplierOrderWhenIs
Received)"/>
   <binding.ws
port="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHumanTask
/InsertSupplierOrderWhenIsReceived#wsdl.endpoint(insertsupplierorderwhenisreceiv
ed_client_ep/InsertSupplierOrderWhenIsReceived_pt)"
            location="http://thesisnoip.no-ip.org:8001/soa-
infra/services/thesis/SupplierOrderHumanTask/insertsupplierorderwhenisreceived_cl
ient_ep?WSDL"
            soapVersion="1.1">
     <property name="weblogic.wsee.wsat.transaction.flowOption"
            type="xs:string" many="false">WSDLDriven</property>
   </binding.ws>
  </reference>
  <wire>
   <source.uri>processorderandsupplier_client_ep</source.uri>
   <target.uri>ProcessOrderAndSupplier/processorderandsupplier_client</target.uri>
  </wire>
  <wire>
   <source.uri>ProcessOrderAndSupplier/Repository</source.uri>
   <target.uri>Repository</target.uri>
  </wire>
  <wire>
   <source.uri>ProcessOrderAndSupplier/SupplierA</source.uri>
   <target.uri>SupplierA</target.uri>
  </wire>
  <wire>
   <source.uri>ProcessOrderAndSupplier/SupplierB</source.uri>
   <target.uri>SupplierB</target.uri>
  </wire>
  <wire>

<source.uri>ProcessOrderAndSupplier/SupplierOrderSubmitHumanTask</source.uri>
   <target.uri>SupplierOrderSubmitHumanTask</target.uri>
  </wire>
```


Πανεπιστήμιο Πειραιώς

```
</composite>
```

Κώδικας 46: ProcessOrderAndSupplier.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema attributeFormDefault="unqualified"
        elementFormDefault="qualified"
        targetNamespace="http://xmlns.oracle.com/BuyingProcessApplicationV01/Pro
cessOrderAndSuppliers/ProcessOrderAndSupplier"
        xmlns="http://www.w3.org/2001/XMLSchema">
 <element name="process">
                <complexType>
                        <sequence>
                                <element name="Products" maxOccurs="unbounded">
   <complexType>
    <sequence>
     <element name="ProductId" type="unsignedInt" maxOccurs="1"/>
     <element name="Quantity" type="unsignedInt" maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <element name="OrderId" type="unsignedInt" maxOccurs="1"/>
 </sequence>
                </complexType>
        </element>
        <element name="processResponse">
                <complexType>
                        <sequence>
                                <element name="result" type="string"/>
                        </sequence>
                </complexType>
        </element>
</schema>
```

6.9 Κώδικας διακασίας BPEL συναλλαγής με πιστωτική κάρτα

Κώδικας 47: CreditCardGateway.bpel

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<!--
//////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////
 Oracle JDeveloper BPEL Designer
```


Πανεπιστήμιο Πειραιώς

```
 Created: Thu Feb 09 12:05:02 EET 2012
 Author:  Epameinondas Nikolopoulos
 Type: BPEL 2.0
 Purpose: Synchronous BPEL Process
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
-->
<process name="CreditCardGateway"

targetNamespace="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway"
          xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"

xmlns:client="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway"
          xmlns:ora="http://schemas.oracle.com/xpath/extension"
          xmlns:bpelx="http://schemas.oracle.com/bpel/extension"
      xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
      xmlns:ns1="http://ws.cdyne.com/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
      xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
      xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
      xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
      xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions"
      xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
      xmlns:ns2="http://webservice.gateway.creditcard.com/"
      xmlns:ns3="http://webservice.ethnikigr.com/"
      xmlns:ns4="http://xmlns.oracle.com/ias/pcbpel/NotificationService"
      xmlns:ns5="http://xmlns.oracle.com/bpel/workflow/taskService"
      xmlns:ns6="http://webservice.alphabankgr.com/">

   <import
namespace="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway"
location="CreditCardGateway.wsdl"
```


Πανεπιστήμιο Πειραιώς

```
importType="http://schemas.xmlsoap.org/wsdl/"/>
  <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
     PARTNERLINKS
     List of services participating in this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
  -->
 <partnerLinks>
  <!--
    The 'client' role represents the requester of this service. It is
    used for callback. The location and correlation information associated
    with the client role are automatically set using WS-Addressing.
  -->
  <partnerLink name="creditcardgateway_client"
partnerLinkType="client:CreditCardGateway" myRole="CreditCardGatewayProvider"/>
  <partnerLink name="CrediCardValidator"
           partnerLinkType="ns1:CrediCardValidator"
           partnerRole="LuhnCheckerSoap" myRole="LuhnCheckerSoap"/>
  <partnerLink name="FindBank" partnerLinkType="ns2:FindBank"
           partnerRole="CreditCardGatewayWS"/>
  <partnerLink name="BankA"
           partnerLinkType="ns3:InsertCrediCardTransaction"
           partnerRole="EthnikiGrWS"/>
  <partnerLink name="BankB" partnerLinkType="ns6:BankB"
           partnerRole="AlphaBankGrWS"/>
 </partnerLinks>

 <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
    VARIABLES
    List of messages and XML documents used within this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
 -->
 <variables>
  <!-- Reference to the message passed as input during initiation -->
  <variable name="inputVariable"
messageType="client:CreditCardGatewayRequestMessage"/>
```


Πανεπιστήμιο Πειραιώς

```
  <!-- Reference to the message that will be returned to the requester-->
  <variable name="outputVariable"
messageType="client:CreditCardGatewayResponseMessage"/>
  <variable name="Invoke1_CheckCC_InputVariable"
        messageType="ns1:CheckCCSoapIn"/>
  <variable name="Invoke1_CheckCC_OutputVariable"
        messageType="ns1:CheckCCSoapOut"/>
  <variable name="CCVCheckCreditCardType" type="xsd:boolean"/>
  <variable name="InvokeGateway_FindBank_OutputVariable"
        messageType="ns2:FindBankResponse"/>
  <variable name="InvokeGateway_FindBank_InputVariable"
        messageType="ns2:FindBank"/>
  <variable name="Invoke2_InsertCardTransaction_InputVariable"
        messageType="ns3:InsertCardTransaction"/>
  <variable name="Invoke2_InsertCardTransaction_OutputVariable"
        messageType="ns3:InsertCardTransactionResponse"/>
  <variable name="Invoke3_InsertCardTransaction_InputVariable"
        messageType="ns6:InsertCardTransaction"/>
  <variable name="Invoke3_InsertCardTransaction_OutputVariable"
        messageType="ns6:InsertCardTransactionResponse"/>
 </variables>

 <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
   ORCHESTRATION LOGIC
   Set of activities coordinating the flow of messages across the
   services integrated within this business process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
 -->
 <sequence name="main">

  <!-- Receive input from requestor. (Note: This maps to operation defined in
CreditCardGateway.wsdl) -->
  <receive name="receiveInput" partnerLink="creditcardgateway_client"
portType="client:CreditCardGateway" operation="CreditCardProcessing"
variable="inputVariable" createInstance="yes"/>

  <!-- Generate reply to synchronous request -->
  <assign name="InputForCardNumberAndTypeValidation">
   <copy>
    <from>$inputVariable.payload/client:CardNumber</from>
    <to>$Invoke1_CheckCC_InputVariable.parameters/ns1:CardNumber</to>
```


Πανεπιστήμιο Πειραιώς

```
    </copy>
   </assign>
   <invoke name="CardNumberAndTypeValidation"
         partnerLink="CrediCardValidator" portType="ns1:LuhnCheckerSoap"
         operation="CheckCC" inputVariable="Invoke1_CheckCC_InputVariable"
         bpelx:invokeAsDetail="no"
         outputVariable="Invoke1_CheckCC_OutputVariable"/>
   <assign name="CheckOutput">
    <copy>

<from>($inputVariable.payload/client:CardType=$Invoke1_CheckCC_OutputVariable.pa
rameters/ns1:CheckCCResult/ns1:CardType)</from>
      <to>$CCVCheckCreditCardType</to>
    </copy>
   </assign>
   <if name="CheckIfCreditCardIsValid">
    <documentation>Valid</documentation>

<condition>$Invoke1_CheckCC_OutputVariable.parameters/ns1:CheckCCResult/ns1:Car
dValid and $CCVCheckCreditCardType</condition>
    <empty name="Continue"/>
    <else>
     <documentation>Not Valid</documentation>
     <sequence name="Sequence3">
      <assign name="AssignError1">
       <copy>
        <from>false()</from>
        <to>$outputVariable.payload/client:result</to>
       </copy>
      </assign>
      <reply name="SendError1" variable="outputVariable"
           partnerLink="creditcardgateway_client"
           portType="client:CreditCardGateway"
           operation="CreditCardProcessing"/>
      <exit name="Exit1"/>
     </sequence>
    </else>
   </if>
   <assign name="InputForGetaway">
    <copy>
     <from>$inputVariable.payload/client:CardNumber</from>
     <to>$InvokeGateway_FindBank_InputVariable.parameters/cardNumber</to>
    </copy>
   </assign>
   <invoke name="InvokeGateway"
         partnerLink="FindBank" portType="ns2:CreditCardGatewayWS"
```


Πανεπιστήμιο Πειραιώς

```
        operation="FindBank"
        inputVariable="InvokeGateway_FindBank_InputVariable"
        outputVariable="InvokeGateway_FindBank_OutputVariable"
        bpelx:invokeAsDetail="no"/>
   <if name="CheckIfBankFound">
    <documentation>Bank Found</documentation>

<condition>$InvokeGateway_FindBank_OutputVariable.parameters/return/status</con
dition>
    <if name="SelectBank">
     <documentation>BankA</documentation>
     <condition>($InvokeGateway_FindBank_OutputVariable.parameters/return/bank =
"EthnikiGr")</condition>
      <sequence name="Sequence1">
       <assign name="AssignInputForTransaction1">
        <copy>
         <from>$inputVariable.payload/client:CardNumber</from>

<to>$Invoke2_InsertCardTransaction_InputVariable.parameters/cardNumber</to>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardHoldersName</from>

<to>$Invoke2_InsertCardTransaction_InputVariable.parameters/cardHoldersName</to
>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardExpirationMonth</from>

<to>$Invoke2_InsertCardTransaction_InputVariable.parameters/cardExpirationMonth<
/to>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardExpirationYear</from>

<to>$Invoke2_InsertCardTransaction_InputVariable.parameters/cardExpirationYear</t
o>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardVerificationCode</from>

<to>$Invoke2_InsertCardTransaction_InputVariable.parameters/cardVerificationCode<
/to>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardBillingAmount</from>
```


Πανεπιστήμιο Πειραιώς

```
<to>$Invoke2_InsertCardTransaction_InputVariable.parameters/cardBillingAmount</t
o>
       </copy>
       <copy>
        <from>number("321890")</from>

<to>$Invoke2_InsertCardTransaction_InputVariable.parameters/businessPartner</to>
       </copy>
      </assign>
      <invoke name="InsertTransaction1"
           inputVariable="Invoke2_InsertCardTransaction_InputVariable"
           outputVariable="Invoke2_InsertCardTransaction_OutputVariable"
           bpelx:invokeAsDetail="no"
           partnerLink="BankA"
           portType="ns3:EthnikiGrWS" operation="InsertCardTransaction"/>
      <assign name="TransactionOutput1">
       <copy>

<from>$Invoke2_InsertCardTransaction_OutputVariable.parameters/return</from>
        <to>$outputVariable.payload/client:result</to>
       </copy>
      </assign>
     </sequence>
     <elseif>
      <documentation>BankB</documentation>
      <condition>($InvokeGateway_FindBank_OutputVariable.parameters/return/bank
="AlphaBankGr")</condition>
      <sequence name="Sequence4">
       <assign name="AssignInputForTransaction2">
        <copy>
         <from>$inputVariable.payload/client:CardNumber</from>

<to>$Invoke3_InsertCardTransaction_InputVariable.parameters/cardNumber</to>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardHoldersName</from>

<to>$Invoke3_InsertCardTransaction_InputVariable.parameters/cardHoldersName</to
>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardExpirationMonth</from>

<to>$Invoke3_InsertCardTransaction_InputVariable.parameters/cardExpirationMonth<
/to>
```


Πανεπιστήμιο Πειραιώς

```
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardExpirationYear</from>

<to>$Invoke3_InsertCardTransaction_InputVariable.parameters/cardExpirationYear</t
o>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardVerificationCode</from>

<to>$Invoke3_InsertCardTransaction_InputVariable.parameters/cardVerificationCode<
/to>
        </copy>
        <copy>
         <from>$inputVariable.payload/client:CardBillingAmount</from>

<to>$Invoke3_InsertCardTransaction_InputVariable.parameters/cardBillingAmount</t
o>
        </copy>
        <copy>
         <from>number(23131)</from>

<to>$Invoke3_InsertCardTransaction_InputVariable.parameters/businessPartner</to>
        </copy>
       </assign>
       <invoke name="InsertTransaction2"
            partnerLink="BankB" portType="ns6:AlphaBankGrWS"
            operation="InsertCardTransaction"
            inputVariable="Invoke3_InsertCardTransaction_InputVariable"
            outputVariable="Invoke3_InsertCardTransaction_OutputVariable"
            bpelx:invokeAsDetail="no"/>
       <assign name="TransactionOutput2">
        <copy>

<from>$Invoke3_InsertCardTransaction_OutputVariable.parameters/return</from>
          <to>$outputVariable.payload/client:result</to>
        </copy>
       </assign>
      </sequence>
    </elseif>
   </if>
   <else>
    <documentation>Bank Not Found</documentation>
    <sequence name="Sequence2">
     <assign name="AssignError2">
      <copy>
```


Πανεπιστήμιο Πειραιώς

```
        <from>false()</from>
        <to>$outputVariable.payload/client:result</to>
       </copy>
      </assign>
      <reply name="SendError2" variable="outputVariable"
          partnerLink="creditcardgateway_client"
          portType="client:CreditCardGateway"
          operation="CreditCardProcessing"/>
      <exit name="Exit2"/>
     </sequence>
    </else>
   </if>
   <reply name="replyOutput" partnerLink="creditcardgateway_client"
portType="client:CreditCardGateway" operation="CreditCardProcessing"
variable="outputVariable"/>
 </sequence>
</process>
```

Κώδικας 48: CreditCardGateway.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema attributeFormDefault="unqualified"
        elementFormDefault="qualified"
        targetNamespace="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGat
eway"
        xmlns="http://www.w3.org/2001/XMLSchema">
        <element name="process">
                <complexType>
                        <sequence>
                                <element name="CardNumber" type="string"/>
                 <element name="CardHoldersName" type="string"/>
                 <element name="CardExpirationMonth" type="integer"/>
                 <element name="CardExpirationYear" type="integer"/>
                 <element name="CardVerificationCode" type="integer"/>
                 <element name="CardType" type="string"/>
                 <element name="CardBillingAmount" type="float"/>
               </sequence>
                </complexType>
        </element>
        <element name="processResponse">
                <complexType>
                        <sequence>
                                <element name="result" type="boolean"/>
                        </sequence>
                </complexType>
        </element>
</schema>
```

Πανεπιστήμιο Πειραιώς

Κώδικας 49: composite.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 1.0 at [2/9/12 12:03 PM]. -->
<composite name="CreditCardGateway"
        revision="1.0"
        label="2012-02-09_12-03-58_117"
        mode="active"
        state="on"
        xmlns="http://xmlns.oracle.com/sca/1.0"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
        xmlns:ui="http://xmlns.oracle.com/soa/designer/">
 <import
namespace="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway"
       location="CreditCardGateway.wsdl" importType="wsdl"/>
 <import namespace="http://ws.cdyne.com/"
       location="http://ws.cdyne.com/creditcardverify/luhnchecker.asmx?wsdl"
       importType="wsdl"/>
 <import namespace="http://ws.cdyne.com/"
       location="CrediCardValidatorWrapper.wsdl" importType="wsdl"/>
 <import namespace="http://webservice.gateway.creditcard.com/"
       location="CreditCardGatewayWSWrapper.wsdl" importType="wsdl"/>
 <import namespace="http://webservice.gateway.creditcard.com/"

location="http://176.34.182.139:8888/ThesisWebServicesA/CreditCardGatewayWS?W
SDL"
       importType="wsdl"/>
 <import namespace="http://webservice.ethnikigr.com/"
       location="http://176.34.182.139:8888/ThesisWebServicesA/EthnikiGrWS?WSDL"
       importType="wsdl"/>
 <import namespace="http://webservice.ethnikigr.com/"
       location="EthnikiGrWSWrapper.wsdl" importType="wsdl"/>
 <import namespace="http://xmlns.oracle.com/bpel/workflow/taskService"
       location="oramds:/soa/shared/workflow/TaskServiceInterface.wsdl"
       importType="wsdl"/>
 <import namespace="http://webservice.alphabankgr.com/"

location="http://176.34.182.139:8888/ThesisWebServicesA/AlphaBankGrWS?WSDL"
       importType="wsdl"/>
 <import namespace="http://webservice.alphabankgr.com/"
```


Πανεπιστήμιο Πειραιώς

```
      location="AlphaBankGrWSWrapper.wsdl" importType="wsdl"/>
 <service name="creditcardgateway_client_ep"
       ui:wsdlLocation="CreditCardGateway.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway#wsdl.inte
rface(CreditCardGateway)"/>
  <binding.ws
port="http://xmlns.oracle.com/ThesisV01/Project1/CreditCardGateway#wsdl.endpoin
t(creditcardgateway_client_ep/CreditCardGateway_pt)"/>
 </service>
 <component name="CreditCardGateway" version="2.0">
  <implementation.bpel src="CreditCardGateway.bpel"/>
 </component>
 <reference name="CrediCardValidator"

ui:wsdlLocation="http://ws.cdyne.com/creditcardverify/luhnchecker.asmx?wsdl">
  <interface.wsdl interface="http://ws.cdyne.com/#wsdl.interface(LuhnCheckerSoap)"

callbackInterface="http://ws.cdyne.com/#wsdl.interface(LuhnCheckerSoap)"/>
  <binding.ws
port="http://ws.cdyne.com/#wsdl.endpoint(LuhnChecker/LuhnCheckerSoap12)"
          location="http://ws.cdyne.com/creditcardverify/luhnchecker.asmx?wsdl"
          soapVersion="1.2">
   <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
  <binding.ws
port="http://ws.cdyne.com/#wsdl.endpoint(LuhnChecker/LuhnCheckerSoap)"
          location="http://ws.cdyne.com/creditcardverify/luhnchecker.asmx?wsdl"
          soapVersion="1.1">
   <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
  <callback>
   <binding.ws
port="http://ws.cdyne.com/#wsdl.endpoint(LuhnChecker/LuhnCheckerSoap_pt)"/>
  </callback>
  <callback>
   <binding.ws
port="http://ws.cdyne.com/#wsdl.endpoint(LuhnChecker/LuhnCheckerSoap_pt)"/>
  </callback>
 </reference>
 <reference name="FindBank"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisWebServicesA/CreditCardGatewa
yWS?WSDL">
```


Πανεπιστήμιο Πειραιώς

```
  <interface.wsdl
interface="http://webservice.gateway.creditcard.com/#wsdl.interface(CreditCardGate
wayWS)"/>
  <binding.ws
port="http://webservice.gateway.creditcard.com/#wsdl.endpoint(CreditCardGateway
WS/CreditCardGatewayWSPort)"

location="http://176.34.182.139:8888/ThesisWebServicesA/CreditCardGatewayWS?W
SDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="BankA"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisWebServicesA/EthnikiGrWS?WS
DL">
  <interface.wsdl
interface="http://webservice.ethnikigr.com/#wsdl.interface(EthnikiGrWS)"/>
  <binding.ws
port="http://webservice.ethnikigr.com/#wsdl.endpoint(EthnikiGrWS/EthnikiGrWSPor
t)"

location="http://176.34.182.139:8888/ThesisWebServicesA/EthnikiGrWS?WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="BankB"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisWebServicesA/AlphaBankGrWS?
WSDL">
  <interface.wsdl
interface="http://webservice.alphabankgr.com/#wsdl.interface(AlphaBankGrWS)"/>
  <binding.ws
port="http://webservice.alphabankgr.com/#wsdl.endpoint(AlphaBankGrWS/AlphaBan
kGrWSPort)"

location="http://176.34.182.139:8888/ThesisWebServicesA/AlphaBankGrWS?WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
```


Πανεπιστήμιο Πειραιώς

```
  <wire>
   <source.uri>creditcardgateway_client_ep</source.uri>
   <target.uri>CreditCardGateway/creditcardgateway_client</target.uri>
  </wire>
  <wire>
   <source.uri>CreditCardGateway/CrediCardValidator</source.uri>
   <target.uri>CrediCardValidator</target.uri>
  </wire>
  <wire>
   <source.uri>CreditCardGateway/FindBank</source.uri>
   <target.uri>FindBank</target.uri>
  </wire>
  <wire>
   <source.uri>CreditCardGateway/BankA</source.uri>
   <target.uri>BankA</target.uri>
  </wire>
  <wire>
   <source.uri>CreditCardGateway/BankB</source.uri>
   <target.uri>BankB</target.uri>
  </wire>
</composite>
```

6.10 Κώδικας Human Task διαδικασίας BPEL επιβεβαίωσης συναλλαγής με κατάθεση στην τράπεζα

Κώδικας 50: BankPaymentProcess.bpel

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<!--
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
 Oracle JDeveloper BPEL Designer

 Created: Sat Feb 11 16:40:44 EET 2012
 Author:  Epameinondas Nikolopoulos
 Type: BPEL 2.0 Process
 Purpose: Asynchronous BPEL Process
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
-->
<process name="BankPaymentProcess"

targetNamespace="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankP
```


Πανεπιστήμιο Πειραιώς

```
aymentProcess"
      xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"

xmlns:client="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankPaym
entProcess"
      xmlns:ora="http://schemas.oracle.com/xpath/extension"
      xmlns:bpelx="http://schemas.oracle.com/bpel/extension"
      xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
      xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
      xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
      xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
      xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
      xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions"
      xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
      xmlns:task="http://xmlns.oracle.com/bpel/workflow/task"
      xmlns:taskservice="http://xmlns.oracle.com/bpel/workflow/taskService"
      xmlns:wfcommon="http://xmlns.oracle.com/bpel/workflow/common"
      xmlns:wf="http://schemas.oracle.com/bpel/extensions/workflow"
      xmlns:ns1="http://xmlns.oracle.com/bpel/workflow/TaskEvidenceService"
      xmlns:ns2="http://xmlns.oracle.com/bpel/workflow/routingSlip">

   <import
namespace="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankPayme
ntProcess" location="BankPaymentProcess.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
   <!--

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
      PARTNERLINKS
      List of services participating in this BPEL process

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
```


Πανεπιστήμιο Πειραιώς

```
  -->
  <partnerLinks>
    <!--
   The 'client' role represents the requester of this service. It is
   used for callback. The location and correlation information associated
   with the client role are automatically set using WS-Addressing.
  -->
     <partnerLink name="bankpaymentprocess_client"
partnerLinkType="client:BankPaymentProcess"
myRole="BankPaymentProcessProvider"
partnerRole="BankPaymentProcessRequester"/>
     <partnerLink name="BankPaymentVerification.TaskService_1"
            partnerLinkType="taskservice:TaskService"
            partnerRole="TaskService"
            myRole="TaskServiceCallbackListener"/>
  </partnerLinks>

  <!--

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
     VARIABLES
     List of messages and XML documents used within this BPEL process

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
  -->
  <variables>
    <!-- Reference to the message passed as input during initiation -->
    <variable name="inputVariable"
messageType="client:BankPaymentProcessRequestMessage"/>

    <!-- Reference to the message that will be sent back to the requester during callback
-->
    <variable name="outputVariable"
messageType="client:BankPaymentProcessResponseMessage"/>
    <variable name="BankPaymentVerification1_globalVariable"
          messageType="taskservice:taskMessage"/>
  </variables>

  <!--

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
     ORCHESTRATION LOGIC
     Set of activities coordinating the flow of messages across the
```


Πανεπιστήμιο Πειραιώς

```
    services integrated within this business process

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
  -->
  <sequence name="main">
    <!-- Receive input from requestor. (Note: This maps to operation defined in
BankPaymentProcess.wsdl) -->
    <receive name="receiveInput" partnerLink="bankpaymentprocess_client"
portType="client:BankPaymentProcess" operation="BankPaymentProcess"
variable="inputVariable" createInstance="yes"/>

    <!--
     Asynchronous callback to the requester. (Note: the callback location and
correlation id is transparently handled using WS-addressing.)
    -->
    <scope name="BankPaymentVerification1"
        xmlns:wf="http://schemas.oracle.com/bpel/extensions/workflow"
        wf:key="BankPaymentVerification1_globalVariable">
      <bpelx:annotation>
        <bpelx:pattern patternName="bpelx:workflow"/>
      </bpelx:annotation>
      <variables>
        <variable name="initiateTaskInput"
              messageType="taskservice:initiateTaskMessage"/>
        <variable name="initiateTaskResponseMessage"
              messageType="taskservice:initiateTaskResponseMessage"/>
      </variables>
      <sequence>
        <assign name="BankPaymentVerification1_AssignTaskAttributes">
          <copy>
            <from>$inputVariable.payload/client:OrderNumber</from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:title</query></to>
          </copy>
          <copy>
            <from>number(3)</from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:priority</query></to>
          </copy>
          <copy>
            <from>$inputVariable.payload/client:OrderNumber</from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:identityContext</query></to>
          </copy>
          <copy>
```


Πανεπιστήμιο Πειραιώς

```
            <from>$inputVariable.payload/client:OrderNumber</from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:title</query></to>
          </copy>
          <copy>
            <from>number(3)</from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:priority</query></to>
          </copy>
          <copy>
            <from>$inputVariable.payload/client:OrderNumber</from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:identityContext</query></to>
          </copy>
          <copy>
            <from><literal><payload
xmlns="http://xmlns.oracle.com/bpel/workflow/task">
 <BPOrderNumber xmlns="http://xmlns.oracle.com/bpel/workflow/task"/>
 <BPOrderPrice xmlns="http://xmlns.oracle.com/bpel/workflow/task"/>
</payload></literal></from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload</query></to>
          </copy>
          <copy>
            <from variable="inputVariable"
part="payload"><query>client:OrderNumber</query></from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload/task:BPOrderNumber</query></to>
          </copy>
          <copy>
            <from variable="inputVariable"
part="payload"><query>client:OrderPrice</query></from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload/task:BPOrderPrice</query></to>
          </copy>
        </assign>
        <invoke name="initiateTask_BankPaymentVerification1"
          partnerLink="BankPaymentVerification.TaskService_1"
          operation="initiateTask"
          portType="taskservice:TaskService"
          inputVariable="initiateTaskInput"
          outputVariable="initiateTaskResponseMessage"/>
        <receive name="receiveCompletedTask_BankPaymentVerification1"
           partnerLink="BankPaymentVerification.TaskService_1"
           portType="taskservice:TaskServiceCallback"
           operation="onTaskCompleted"
```


Πανεπιστήμιο Πειραιώς

```
            variable="BankPaymentVerification1_globalVariable"
            createInstance="no"/>
      </sequence>
    </scope>
    <if name="taskIf">
      <documentation>Task outcome is APPROVE</documentation>
      <bpelx:annotation>
        <bpelx:pattern patternName="Task outcome is APPROVE"/>
      </bpelx:annotation>

<condition>$BankPaymentVerification1_globalVariable.payload/task:systemAttributes/
task:outcome = 'APPROVE'</condition>
      <sequence>
        <assign name="True">
          <copy>
            <from>"true"</from>
            <to>$outputVariable.payload/client:result</to>
          </copy>
        </assign>
      </sequence>
      <elseif>
        <documentation>Task outcome is REJECT</documentation>
        <bpelx:annotation>
          <bpelx:pattern patternName="Task outcome is REJECT"/>
        </bpelx:annotation>

<condition>$BankPaymentVerification1_globalVariable.payload/task:systemAttributes/
task:outcome = 'REJECT'</condition>
        <sequence>
          <assign name="False">
            <copy>
              <from>("false")</from>
              <to>$outputVariable.payload/client:result</to>
            </copy>
          </assign>
        </sequence>
      </elseif>
      <else>
        <sequence>
          <assign name="False">
            <copy>
              <from>("false")</from>
              <to>$outputVariable.payload/client:result</to>
            </copy>
          </assign>
        </sequence>
```


Πανεπιστήμιο Πειραιώς

```
        </else>
      </if>
      <invoke name="callbackClient" partnerLink="bankpaymentprocess_client"
portType="client:BankPaymentProcessCallback"
operation="BankPaymentProcessResponse" inputVariable="outputVariable"/>
   </sequence>
</process>
```

Κώδικας 51: BankPaymentProcess.xsd

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<!--
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
 Oracle JDeveloper BPEL Designer

 Created: Sat Feb 11 16:40:44 EET 2012
 Author:  Nodas
 Type: BPEL 2.0 Process
 Purpose: Asynchronous BPEL Process
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
-->
<process name="BankPaymentProcess"

targetNamespace="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankP
aymentProcess"
      xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"

xmlns:client="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankPaym
entProcess"
      xmlns:ora="http://schemas.oracle.com/xpath/extension"
      xmlns:bpelx="http://schemas.oracle.com/bpel/extension"
      xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
      xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
      xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
      xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
      xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
```


Πανεπιστήμιο Πειραιώς

```
      xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions"
      xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
      xmlns:task="http://xmlns.oracle.com/bpel/workflow/task"
      xmlns:taskservice="http://xmlns.oracle.com/bpel/workflow/taskService"
      xmlns:wfcommon="http://xmlns.oracle.com/bpel/workflow/common"
      xmlns:wf="http://schemas.oracle.com/bpel/extensions/workflow"
      xmlns:ns1="http://xmlns.oracle.com/bpel/workflow/TaskEvidenceService"
      xmlns:ns2="http://xmlns.oracle.com/bpel/workflow/routingSlip">

  <import
namespace="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankPayme
ntProcess" location="BankPaymentProcess.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
  <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
     PARTNERLINKS
     List of services participating in this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
  -->
  <partnerLinks>
    <!--
   The 'client' role represents the requester of this service. It is
   used for callback. The location and correlation information associated
   with the client role are automatically set using WS-Addressing.
  -->
    <partnerLink name="bankpaymentprocess_client"
partnerLinkType="client:BankPaymentProcess"
myRole="BankPaymentProcessProvider"
partnerRole="BankPaymentProcessRequester"/>
    <partnerLink name="BankPaymentVerification.TaskService_1"
          partnerLinkType="taskservice:TaskService"
          partnerRole="TaskService"
          myRole="TaskServiceCallbackListener"/>
  </partnerLinks>

  <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
```


Πανεπιστήμιο Πειραιώς

```
    VARIABLES
    List of messages and XML documents used within this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
  -->
  <variables>
    <!-- Reference to the message passed as input during initiation -->
    <variable name="inputVariable"
messageType="client:BankPaymentProcessRequestMessage"/>

    <!-- Reference to the message that will be sent back to the requester during callback
-->
    <variable name="outputVariable"
messageType="client:BankPaymentProcessResponseMessage"/>
    <variable name="BankPaymentVerification1_globalVariable"
          messageType="taskservice:taskMessage"/>
  </variables>

  <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
    ORCHESTRATION LOGIC
    Set of activities coordinating the flow of messages across the
    services integrated within this business process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
  -->
  <sequence name="main">
    <!-- Receive input from requestor. (Note: This maps to operation defined in
BankPaymentProcess.wsdl) -->
    <receive name="receiveInput" partnerLink="bankpaymentprocess_client"
portType="client:BankPaymentProcess" operation="BankPaymentProcess"
variable="inputVariable" createInstance="yes"/>

    <!--
      Asynchronous callback to the requester. (Note: the callback location and
correlation id is transparently handled using WS-addressing.)
    -->
    <scope name="BankPaymentVerification1"
        xmlns:wf="http://schemas.oracle.com/bpel/extensions/workflow"
        wf:key="BankPaymentVerification1_globalVariable">
      <bpelx:annotation>
        <bpelx:pattern patternName="bpelx:workflow"/>
```


Πανεπιστήμιο Πειραιώς

```
        </bpelx:annotation>
        <variables>
          <variable name="initiateTaskInput"
                messageType="taskservice:initiateTaskMessage"/>
          <variable name="initiateTaskResponseMessage"
                messageType="taskservice:initiateTaskResponseMessage"/>
        </variables>
        <sequence>
          <assign name="BankPaymentVerification1_AssignTaskAttributes">
            <copy>
              <from>$inputVariable.payload/client:OrderNumber</from>
              <to variable="initiateTaskInput"
part="payload"><query>task:task/task:title</query></to>
            </copy>
            <copy>
              <from>number(3)</from>
              <to variable="initiateTaskInput"
part="payload"><query>task:task/task:priority</query></to>
            </copy>
            <copy>
              <from>$inputVariable.payload/client:OrderNumber</from>
              <to variable="initiateTaskInput"
part="payload"><query>task:task/task:identityContext</query></to>
            </copy>
            <copy>
              <from>$inputVariable.payload/client:OrderNumber</from>
              <to variable="initiateTaskInput"
part="payload"><query>task:task/task:title</query></to>
            </copy>
            <copy>
              <from>number(3)</from>
              <to variable="initiateTaskInput"
part="payload"><query>task:task/task:priority</query></to>
            </copy>
            <copy>
              <from>$inputVariable.payload/client:OrderNumber</from>
              <to variable="initiateTaskInput"
part="payload"><query>task:task/task:identityContext</query></to>
            </copy>
            <copy>
              <from><literal><payload
xmlns="http://xmlns.oracle.com/bpel/workflow/task">
 <BPOrderNumber xmlns="http://xmlns.oracle.com/bpel/workflow/task"/>
 <BPOrderPrice xmlns="http://xmlns.oracle.com/bpel/workflow/task"/>
</payload></literal></from>
              <to variable="initiateTaskInput"
```


Πανεπιστήμιο Πειραιώς

```
part="payload"><query>task:task/task:payload</query></to>
            </copy>
            <copy>
              <from variable="inputVariable"
part="payload"><query>client:OrderNumber</query></from>
              <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload/task:BPOrderNumber</query></to>
            </copy>
            <copy>
              <from variable="inputVariable"
part="payload"><query>client:OrderPrice</query></from>
              <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload/task:BPOrderPrice</query></to>
            </copy>
          </assign>
          <invoke name="initiateTask_BankPaymentVerification1"
               partnerLink="BankPaymentVerification.TaskService_1"
               operation="initiateTask"
               portType="taskservice:TaskService"
               inputVariable="initiateTaskInput"
               outputVariable="initiateTaskResponseMessage"/>
          <receive name="receiveCompletedTask_BankPaymentVerification1"
                partnerLink="BankPaymentVerification.TaskService_1"
                portType="taskservice:TaskServiceCallback"
                operation="onTaskCompleted"
                variable="BankPaymentVerification1_globalVariable"
                createInstance="no"/>
        </sequence>
      </scope>
      <if name="taskIf">
        <documentation>Task outcome is APPROVE</documentation>
        <bpelx:annotation>
          <bpelx:pattern patternName="Task outcome is APPROVE"/>
        </bpelx:annotation>

<condition>$BankPaymentVerification1_globalVariable.payload/task:systemAttributes/
task:outcome = 'APPROVE'</condition>
        <sequence>
          <assign name="True">
            <copy>
              <from>"true"</from>
              <to>$outputVariable.payload/client:result</to>
            </copy>
          </assign>
        </sequence>
        <elseif>
```


Πανεπιστήμιο Πειραιώς

```
          <documentation>Task outcome is REJECT</documentation>
          <bpelx:annotation>
            <bpelx:pattern patternName="Task outcome is REJECT"/>
          </bpelx:annotation>

<condition>$BankPaymentVerification1_globalVariable.payload/task:systemAttributes/
task:outcome = 'REJECT'</condition>
          <sequence>
            <assign name="False">
              <copy>
                <from>("false")</from>
                <to>$outputVariable.payload/client:result</to>
              </copy>
            </assign>
          </sequence>
        </elseif>
        <else>
          <sequence>
            <assign name="False">
              <copy>
                <from>("false")</from>
                <to>$outputVariable.payload/client:result</to>
              </copy>
            </assign>
          </sequence>
        </else>
      </if>
      <invoke name="callbackClient" partnerLink="bankpaymentprocess_client"
portType="client:BankPaymentProcessCallback"
operation="BankPaymentProcessResponse" inputVariable="outputVariable"/>
    </sequence>
</process>
```

Κώδικας 52: BankPaymentVerification.task

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<taskDefinition
targetNamespace="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankP
aymentVerification"
xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20" xmlns:ora="http://schemas.oracle.com/xpath/extension"
xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc" xmlns:task="http://xmlns.oracle.com/bpel/workflow/task"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://xmlns.oracle.com/bpel/workflow/taskDefinition"
          xmlns:bpel2="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
```


Πανεπιστήμιο Πειραιώς

```
        xmlns:ns0="http://xmlns.oracle.com/bpel/workflow/common"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"

xmlns:evidence="http://xmlns.oracle.com/bpel/workflow/TaskEvidenceService"
        xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
        xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
        xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions">
  <name>BankPaymentVerification</name>
  <id>${domain_id}_${process_id}_${process_revision}_BankPaymentVerification</id>
  <title type="XPATH"><value>string('Bank Payment ')</value></title>
  <priority>3</priority>
  <description>Bank Payment Verification</description>
  <process processId="" processVersion=""/>
  <routingSlip xmlns="http://xmlns.oracle.com/bpel/workflow/routingSlip">
    <globalConfiguration>
      <applicationContext></applicationContext>
      <taskOwner type="STATIC" identityType="user">weblogic</taskOwner>
      <expirationDuration duration="P3D" type="STATIC"
                useBusinessCalendar="true"/>
      <sharePayloadConfiguration>
        <type>USE_SYSTEM_WIDE_GLOBAL_CONFIGURATION</type>
      </sharePayloadConfiguration>
    </globalConfiguration>
    <participants isAdhocRoutingSupported="false">
      <stage name="Stage1">
        <participant name="VerificationEmployee">
          <resource type="STATIC" identityType="user">weblogic</resource>
        </participant>
      </stage>
    </participants>
    <onErrorParticipant>
      <resource type="STATIC" identityType="user">weblogic</resource>
    </onErrorParticipant>
    <notification includeTaskAttachments="false" actionable="false"
          secureNotifications="false" hideWorklistUrlInEmail="false">
      <action name="ASSIGN" recipient="OWNER"><![CDATA[concat(string('Task '),
/task:task/task:title, string(' requires your attention.'))]]></action>
      <action name="EXPIRE" recipient="ASSIGNEES"><![CDATA[concat(string('Task '),
/task:task/task:title, string(' requires your attention.'))]]></action>
      <action name="ERROR" recipient="OWNER"><![CDATA[concat(string('Task '),
/task:task/task:title, string(' requires your attention.'))]]></action>
```


Πανεπιστήμιο Πειραιώς

```
<groupNotificationConfiguration>SEND_INDIVIDUAL_NOTIFICATION</groupNotificatio
nConfiguration>
    </notification>
  </routingSlip>
  <workflowConfiguration
xmlns="http://xmlns.oracle.com/bpel/workflow/configuration"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <outcomes>
      <outcome>APPROVE</outcome>
      <outcome>REJECT</outcome>
    </outcomes>
    <payload xmlSchemaDefinition="xsd/BankPaymentVerificationPayload.xsd">
      <messageAttribute name="BPOrderNumber" attributeType="SIMPLE_TYPE"
                type="xsd:unsignedInt" updatable="false"
                external="false"/>
      <messageAttribute name="BPOrderPrice" attributeType="SIMPLE_TYPE"
                type="xsd:float" updatable="false" external="false"/>
    </payload>
    <bpelEventListener>false</bpelEventListener>
    <bpelNoCallbacks>false</bpelNoCallbacks>
    <showCompleteGraphicalHistory>true</showCompleteGraphicalHistory>
    <reevalTranslatablesOnUpdate>false</reevalTranslatablesOnUpdate>
    <preActionMandatoryUserSteps/>
    <allowInitiatorEditParticipants>false</allowInitiatorEditParticipants>
    <allowParticipantsEditParticipants>false</allowParticipantsEditParticipants>
    <globalCreationTask>false</globalCreationTask>

<taskFlowFileLocation>file:/C:/JDeveloper/mywork/ThesisBankPayment/BankPayme
ntVerification/public_html/WEB-
INF/BankPaymentVerification_TaskFlow.xml</taskFlowFileLocation>
    <enableAutoClaim>true</enableAutoClaim>
    <workflowConditions/>
  </workflowConfiguration>
</taskDefinition>
```

Κώδικας 53: composite.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 1.0 at [2/11/12 4:30 PM]. -->
<composite name="BankPayment"
      revision="1.0"
      label="2012-02-11_16-30-35_807"
      mode="active"
      state="on"
      xmlns="http://xmlns.oracle.com/sca/1.0"
```


Πανεπιστήμιο Πειραιώς

```
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
        xmlns:ui="http://xmlns.oracle.com/soa/designer/">
 <import
namespace="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankPayme
ntProcess"
       location="BankPaymentProcess.wsdl" importType="wsdl"/>
 <import namespace="http://xmlns.oracle.com/bpel/workflow/taskService"
       location="oramds:/soa/shared/workflow/TaskServiceInterface.wsdl"
       importType="wsdl"/>
 <service name="bankpaymentprocess_client_ep"
        ui:wsdlLocation="BankPaymentProcess.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankPayment
Process#wsdl.interface(BankPaymentProcess)"

callbackInterface="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankP
aymentProcess#wsdl.interface(BankPaymentProcessCallback)"/>
  <binding.ws
port="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankPaymentProc
ess#wsdl.endpoint(bankpaymentprocess_client_ep/BankPaymentProcess_pt)"/>
  <callback>
   <binding.ws
port="http://xmlns.oracle.com/ThesisBankPayment/BankPayment/BankPaymentProc
ess#wsdl.endpoint(bankpaymentprocess_client_ep/BankPaymentProcessCallback_pt)"/
>
  </callback>
 </service>
 <component name="BankPaymentProcess" version="2.0">
  <implementation.bpel src="BankPaymentProcess.bpel"/>
 </component>
 <component name="BankPaymentVerification">
  <implementation.workflow src="BankPaymentVerification.task"/>
 </component>
 <wire>
  <source.uri>bankpaymentprocess_client_ep</source.uri>
  <target.uri>BankPaymentProcess/bankpaymentprocess_client</target.uri>
 </wire>
 <wire>

<source.uri>BankPaymentProcess/BankPaymentVerification.TaskService</source.uri>
  <target.uri>BankPaymentVerification/TaskService</target.uri>
 </wire>
 <wire>
```


Πανεπιστήμιο Πειραιώς

```
<source.uri>BankPaymentProcess/BankPaymentVerification.TaskService_1</source.uri
>
  <target.uri>BankPaymentVerification/TaskService</target.uri>
 </wire>
</composite>
```

6.11 Κώδικας Human Task διαδικασίας BPEL τιμολόγησης και διεκπεραίωσης διαδικασίας

Κώδικας 54: OrderFulfillment.bpel

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<!--
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
 Oracle JDeveloper BPEL Designer

 Created: Fri Feb 17 18:18:19 EET 2012
 Author:  Epameinondas Nikolopoulos
 Type: BPEL 2.0 Process
 Purpose: Asynchronous BPEL Process
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
-->
<process name="OrderFulfillment"

targetNamespace="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentH
umanTask/OrderFulfillment"
      xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"

xmlns:client="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHuma
nTask/OrderFulfillment"
      xmlns:ora="http://schemas.oracle.com/xpath/extension"
      xmlns:bpelx="http://schemas.oracle.com/bpel/extension"
      xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
      xmlns:ns1="http://xmlns.oracle.com/bpel/workflow/taskService"
      xmlns:task="http://xmlns.oracle.com/bpel/workflow/task"
      xmlns:wfcommon="http://xmlns.oracle.com/bpel/workflow/common"
      xmlns:wf="http://schemas.oracle.com/bpel/extensions/workflow"
      xmlns:ns2="http://xmlns.oracle.com/bpel/workflow/TaskEvidenceService"
      xmlns:ns3="http://xmlns.oracle.com/bpel/workflow/routingSlip"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
```


Πανεπιστήμιο Πειραιώς

```
ons.Xpath20"
      xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"
      xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
      xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
      xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
      xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions"
      xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
      xmlns:ns4="http://xmlns.oracle.com/ias/pcbpel/NotificationService">
  <import
namespace="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHuman
Task/OrderFulfillment"
        location="xsd/OrderFulfillment.xsd"
        importType="http://www.w3.org/2001/XMLSchema"/>
  <import
namespace="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHuman
Task/OrderFulfillment" location="OrderFulfillment.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
  <!--

///////////////////////////////////////////////////////////////////////////
///////////////////////////////////////////////////////////////////
     PARTNERLINKS
     List of services participating in this BPEL process

///////////////////////////////////////////////////////////////////////////
///////////////////////////////////////////////////////////////////
  -->
  <partnerLinks>
    <!--
   The 'client' role represents the requester of this service. It is
   used for callback. The location and correlation information associated
   with the client role are automatically set using WS-Addressing.
  -->
    <partnerLink name="orderfulfillment_client"
partnerLinkType="client:OrderFulfillment" myRole="OrderFulfillmentProvider"
partnerRole="OrderFulfillmentRequester"/>
    <partnerLink name="OrderFullfilment.TaskService_1"
```


Πανεπιστήμιο Πειραιώς

```
            partnerLinkType="ns1:TaskService" partnerRole="TaskService"
            myRole="TaskServiceCallbackListener"/>
    <partnerLink name="NotificationService1"
            partnerLinkType="ns4:NotificationServiceLink"
            partnerRole="NotificationServiceProvider"/>
    <partnerLink name="NotificationService2"
            partnerLinkType="ns4:NotificationServiceLink"
            partnerRole="NotificationServiceProvider"/>
  </partnerLinks>

  <!--

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
    VARIABLES
    List of messages and XML documents used within this BPEL process

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
  -->
  <variables>
    <!-- Reference to the message passed as input during initiation -->
    <variable name="inputVariable"
messageType="client:OrderFulfillmentRequestMessage"/>

    <!-- Reference to the message that will be sent back to the requester during callback
-->
    <variable name="outputVariable"
messageType="client:OrderFulfillmentResponseMessage"/>
    <variable name="initiateTask_OrderFullfilment1_initiateTask_InputVariable"
          messageType="ns1:initiateTaskMessage"/>
    <variable name="OrderFullfilment1_globalVariable"
          messageType="ns1:taskMessage"/>
    <variable name="OrderFullfilment2_globalVariable"
          messageType="ns1:taskMessage"/>
  </variables>

  <!--

////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
    ORCHESTRATION LOGIC
    Set of activities coordinating the flow of messages across the
    services integrated within this business process

////////////////////////////////////////////////////////////////////
```


Πανεπιστήμιο Πειραιώς

```
////////////////////////////////////////////////////////////////////////
  -->
  <sequence name="main">
    <!-- Receive input from requestor. (Note: This maps to operation defined in
OrderFulfillment.wsdl) -->
    <receive name="receiveInput" partnerLink="orderfulfillment_client"
portType="client:OrderFulfillment" operation="process" variable="inputVariable"
createInstance="yes"/>

    <!--
     Asynchronous callback to the requester. (Note: the callback location and
correlation id is transparently handled using WS-addressing.)
    -->
    <scope name="OrderFullfilment2"
        xmlns:wf="http://schemas.oracle.com/bpel/extensions/workflow"
        wf:key="OrderFullfilment1_globalVariable">
      <bpelx:annotation>
        <bpelx:pattern patternName="bpelx:workflow"/>
      </bpelx:annotation>
      <variables>
        <variable name="initiateTaskInput"
              messageType="ns1:initiateTaskMessage"/>
        <variable name="initiateTaskResponseMessage"
              messageType="ns1:initiateTaskResponseMessage"/>
      </variables>
      <sequence>
        <assign name="OrderFullfilment2_AssignTaskAttributes">
          <copy>
            <from>number(3)</from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:priority</query></to>
          </copy>
          <copy>
            <from><literal><payload
xmlns="http://xmlns.oracle.com/bpel/workflow/task">
 <Order xmlns="http://xmlns.oracle.com/bpel/workflow/task"/>
</payload></literal></from>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload</query></to>
          </copy>
          <copy>
            <from variable="inputVariable" part="payload"/>
            <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload/task:Order</query></to>
          </copy>
          <copy>
```


Πανεπιστήμιο Πειραιώς

```
                <from>concat(string('Order '),
$inputVariable.payload/client:OrderId)</from>
                <to variable="initiateTaskInput"
part="payload"><query>task:task/task:title</query></to>
              </copy>
            </assign>
            <invoke name="initiateTask_OrderFullfilment2"
                partnerLink="OrderFullfilment.TaskService_1"
                operation="initiateTask" portType="ns1:TaskService"
                inputVariable="initiateTaskInput"
                outputVariable="initiateTaskResponseMessage"/>
            <receive name="receiveCompletedTask_OrderFullfilment2"
                 partnerLink="OrderFullfilment.TaskService_1"
                 portType="ns1:TaskServiceCallback"
                 operation="onTaskCompleted"
                 variable="OrderFullfilment1_globalVariable"
                 createInstance="no"/>
          </sequence>
        </scope>
        <if name="CheckFulfillment">
          <documentation>Task outcome is YES</documentation>
          <bpelx:annotation>
            <bpelx:pattern patternName="Task outcome is YES"/>
          </bpelx:annotation>

<condition>$OrderFullfilment1_globalVariable.payload/task:systemAttributes/task:out
come = 'YES'</condition>
          <sequence>
            <assign name="ReturnTrueOutput">
              <copy>
                <from>true()</from>
                <to>$outputVariable.payload/client:result</to>
              </copy>
            </assign>
          </sequence>
          <elseif>
            <documentation>Task outcome is NO</documentation>
            <bpelx:annotation>
              <bpelx:pattern patternName="Task outcome is NO"/>
            </bpelx:annotation>

<condition>$OrderFullfilment1_globalVariable.payload/task:systemAttributes/task:out
come = 'NO'</condition>
            <sequence>
              <scope name="Email1">
                <bpelx:annotation>
```


Πανεπιστήμιο Πειραιώς

```
                <bpelx:pattern patternName="bpelx:email"/>
                <bpelx:general>
                  <bpelx:property
name="userLabel">EmailApproriateStaffOnNo</bpelx:property>
                </bpelx:general>
              </bpelx:annotation>
              <variables>
                <variable name="varNotificationReq"
                      messageType="ns4:EmailNotificationRequest"/>
                <variable name="varNotificationResponse"
                      messageType="ns4:ArrayOfResponse"/>
                <variable name="NotificationServiceFaultVariable"
                      messageType="ns4:NotificationServiceErrorMessage"/>
              </variables>
              <sequence name="Sequence1">
                <assign name="EmailParamsAssign">
                  <copy>
                    <from>string('Default')</from>
                    <to variable="varNotificationReq"

part="EmailPayload"><query>ns4:FromAccountName</query></to>
                  </copy>
                  <copy>
                    <from>string('')</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:Bcc</query></to>
                  </copy>
                  <copy>
                    <from>string('')</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:Cc</query></to>
                  </copy>
                  <copy>
                    <from>string('')</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:ReplyToAddress</query></to>
                  </copy>
                  <copy>
                    <from>concat(string('Order: '),
$inputVariable.payload/client:OrderId, string(' Fulfillment Error'))</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:Subject</query></to>
                  </copy>
                  <copy>
                    <from>string('thesiscompanyb@mailinator.com')</from>
                    <to variable="varNotificationReq"
```


Πανεπιστήμιο Πειραιώς

```
                    part="EmailPayload"><query>ns4:To</query></to>
                </copy>
                <copy>
                  <from>string('No selected please Check it!')</from>
                  <to variable="varNotificationReq"

part="EmailPayload"><query>ns4:Content/ns4:ContentBody</query></to>
                </copy>
                <copy>
                  <from>string('text/html; charset=UTF-8')</from>
                  <to variable="varNotificationReq"

part="EmailPayload"><query>ns4:Content/ns4:MimeType</query></to>
                </copy>
              </assign>
              <invoke name="InvokeNotificationService"
                   portType="ns4:NotificationService"
                   partnerLink="NotificationService1"
                   inputVariable="varNotificationReq"
                   outputVariable="varNotificationResponse"
                   operation="sendEmailNotification"/>
            </sequence>
          </scope>
          <assign name="ReturnFalseOutput1">
            <copy>
              <from>false()</from>
              <to>$outputVariable.payload/client:result</to>
            </copy>
          </assign>
        </sequence>
      </elseif>
      <else>
        <sequence>
          <scope name="Email2">
            <bpelx:annotation>
              <bpelx:pattern patternName="bpelx:email"/>
              <bpelx:general>
                <bpelx:property
name="userLabel">EmailApproriateStaffOnOtherSituations</bpelx:property>
              </bpelx:general>
            </bpelx:annotation>
            <variables>
              <variable name="varNotificationReq"
                    messageType="ns4:EmailNotificationRequest"/>
              <variable name="varNotificationResponse"
                    messageType="ns4:ArrayOfResponse"/>
```

Πανεπιστήμιο Πειραιώς

```
                <variable name="NotificationServiceFaultVariable"
                      messageType="ns4:NotificationServiceErrorMessage"/>
              </variables>
              <sequence name="Sequence2">
                <assign name="EmailParamsAssign">
                  <copy>
                    <from>string('Default')</from>
                    <to variable="varNotificationReq"

part="EmailPayload"><query>ns4:FromAccountName</query></to>
                  </copy>
                  <copy>
                    <from>string('')</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:Bcc</query></to>
                  </copy>
                  <copy>
                    <from>string('')</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:Cc</query></to>
                  </copy>
                  <copy>
                    <from>string('')</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:ReplyToAddress</query></to>
                  </copy>
                  <copy>
                    <from>concat(string('Order: '),
$inputVariable.payload/client:OrderId, string(' Fulfillment Error'))</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:Subject</query></to>
                  </copy>
                  <copy>
                    <from>string('thesiscompanyb@mailinator.com')</from>
                    <to variable="varNotificationReq"
                      part="EmailPayload"><query>ns4:To</query></to>
                  </copy>
                  <copy>
                    <from>string('Other option selected please check')</from>
                    <to variable="varNotificationReq"

part="EmailPayload"><query>ns4:Content/ns4:ContentBody</query></to>
                  </copy>
                  <copy>
                    <from>string('text/html; charset=UTF-8')</from>
                    <to variable="varNotificationReq"
```


Πανεπιστήμιο Πειραιώς

```
part="EmailPayload"><query>ns4:Content/ns4:MimeType</query></to>
                    </copy>
                  </assign>
                  <invoke name="InvokeNotificationService"
                        portType="ns4:NotificationService"
                        partnerLink="NotificationService2"
                        inputVariable="varNotificationReq"
                        outputVariable="varNotificationResponse"
                        operation="sendEmailNotification"/>
                </sequence>
              </scope>
              <assign name="ReturnFalseOutput2">
                <copy>
                  <from>false()</from>
                  <to>$outputVariable.payload/client:result</to>
                </copy>
              </assign>
            </sequence>
          </else>
      </if>
      <invoke name="callbackClient" partnerLink="orderfulfillment_client"
portType="client:OrderFulfillmentCallback" operation="processResponse"
inputVariable="outputVariable"/>
    </sequence>
</process>
```

Κώδικας 55: OrderFulfillment.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema attributeFormDefault="unqualified"
          elementFormDefault="qualified"
          targetNamespace="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfi
llmentHumanTask/OrderFulfillment"
          xmlns="http://www.w3.org/2001/XMLSchema">
          <element name="process">
                    <complexType>
                              <sequence>
                                        <element name="orderId" type="int"
maxOccurs="unbounded"/>
                              </sequence>
                    </complexType>
          </element>
 <element name="orderProducts"/>
 <complexType name="OrderProducts2">
  <sequence>
```


Πανεπιστήμιο Πειραιώς

```
  <element name="Products" maxOccurs="unbounded">
   <complexType>
    <sequence>
     <element name="ProductId" type="unsignedInt"/>
     <element name="ProductName" type="string"/>
     <element name="ProductQuantity" type="unsignedInt"/>
     <element name="UnitPrice" type="float"/>
     <element name="TotalPrice" type="float"/>
    </sequence>
   </complexType>
  </element>
  <element name="OrderId" type="unsignedInt"/>
 </sequence>
</complexType>
<element name="processResponse">
                <complexType>
                        <sequence>
                                <element name="result" type="boolean"/>
                        </sequence>
                </complexType>
        </element>
</schema>
```

Κώδικας 56: composite.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 1.0 at [2/17/12 5:57 PM]. -->
<composite name="OrderFulfillmentHumanTask"
        revision="1.0"
        label="2012-02-17_17-57-23_667"
        mode="active"
        state="on"
        xmlns="http://xmlns.oracle.com/sca/1.0"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
        xmlns:ui="http://xmlns.oracle.com/soa/designer/">
 <import namespace="http://xmlns.oracle.com/bpel/workflow/taskService"
       location="oramds:/soa/shared/workflow/TaskServiceInterface.wsdl"
       importType="wsdl"/>
 <import
namespace="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHuman
Task/OrderFulfillment"
       location="OrderFulfillment.wsdl" importType="wsdl"/>
 <import namespace="http://xmlns.oracle.com/ias/pcbpel/NotificationService"
       location="NotificationService.wsdl" importType="wsdl"/>
```


Πανεπιστήμιο Πειραιώς

```
 <service name="orderfulfillment_client_ep"
       ui:wsdlLocation="OrderFulfillment.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHumanTa
sk/OrderFulfillment#wsdl.interface(OrderFulfillment)"

callbackInterface="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentH
umanTask/OrderFulfillment#wsdl.interface(OrderFulfillmentCallback)"/>
  <binding.ws
port="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHumanTask/O
rderFulfillment#wsdl.endpoint(orderfulfillment_client_ep/OrderFulfillment_pt)"/>
  <callback>
   <binding.ws
port="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHumanTask/O
rderFulfillment#wsdl.endpoint(orderfulfillment_client_ep/OrderFulfillmentCallback_pt)
"/>
  </callback>
 </service>
 <component name="OrderFullfilment">
  <implementation.workflow src="OrderFullfilment.task"/>
 </component>
 <component name="OrderFulfillment" version="2.0">
  <implementation.bpel src="OrderFulfillment.bpel"/>
 </component>
 <reference name="NotificationService1"
       ui:wsdlLocation="NotificationService.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.interface(Noti
ficationService)"/>
  <binding.wsif
port="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.endpoint(Notificat
ionService/JavaPort)"
          location="NotificationService.wsdl"/>
 </reference>
 <reference name="NotificationService2"
       ui:wsdlLocation="NotificationService.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.interface(Noti
ficationService)"/>
  <binding.wsif
port="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.endpoint(Notificat
ionService/JavaPort)"
          location="NotificationService.wsdl"/>
 </reference>
 <wire>
  <source.uri>orderfulfillment_client_ep</source.uri>
```


Πανεπιστήμιο Πειραιώς

```
  <target.uri>OrderFulfillment/orderfulfillment_client</target.uri>
 </wire>
 <wire>
  <source.uri>OrderFulfillment/OrderFullfilment.TaskService_1</source.uri>
  <target.uri>OrderFullfilment/TaskService</target.uri>
 </wire>
 <wire>
  <source.uri>OrderFulfillment/NotificationService1</source.uri>
  <target.uri>NotificationService1</target.uri>
 </wire>
 <wire>
  <source.uri>OrderFulfillment/NotificationService2</source.uri>
  <target.uri>NotificationService2</target.uri>
 </wire>
</composite>
```

Κώδικας 57: OrderFulfillmnent.task

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<taskDefinition
targetNamespace="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentH
umanTask/OrderFullfilment"
xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20" xmlns:ora="http://schemas.oracle.com/xpath/extension"
xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc" xmlns:task="http://xmlns.oracle.com/bpel/workflow/task"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://xmlns.oracle.com/bpel/workflow/taskDefinition"
          xmlns:bpel2="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
          xmlns:ns0="http://xmlns.oracle.com/bpel/workflow/common"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"

xmlns:evidence="http://xmlns.oracle.com/bpel/workflow/TaskEvidenceService"
          xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
          xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
          xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions">
  <name>OrderFullfilment</name>
  <id>${domain_id}_${process_id}_${process_revision}_OrderFullfilment</id>
  <title type="XPATH"><value>string('Prepare Order')</value></title>
  <priority>3</priority>
  <description>Order preparation for dispatch</description>
```


Πανεπιστήμιο Πειραιώς

```
  <process processId="" processVersion=""/>
  <routingSlip xmlns="http://xmlns.oracle.com/bpel/workflow/routingSlip">
    <globalConfiguration>
      <applicationContext></applicationContext>
      <taskOwner type="STATIC" identityType="user">weblogic</taskOwner>
      <expirationDuration duration="PT2H" type="STATIC"
                  useBusinessCalendar="true"/>
      <sharePayloadConfiguration>
        <type>USE_SYSTEM_WIDE_GLOBAL_CONFIGURATION</type>
      </sharePayloadConfiguration>
    </globalConfiguration>
    <participants isAdhocRoutingSupported="false">
      <stage name="AddInvoice">
        <participant name="FinanceStaff">
          <resource type="STATIC" identityType="user">finance</resource>
        </participant>
      </stage>
      <stage name="ClaimAndFullFillOrder">
        <participant name="OrderFulfillmentStaff">
          <resource type="STATIC" identityType="user">orderfulfillment</resource>
          <resource type="STATIC" identityType="user">orderfulfillment1</resource>
        </participant>
      </stage>
    </participants>
    <onErrorParticipant>
      <resource type="STATIC" identityType="user">orderfulfillment</resource>
      <resource type="STATIC" identityType="user">orderfulfillment1</resource>
    </onErrorParticipant>
    <notification includeTaskAttachments="false" actionable="false"
            secureNotifications="false" hideWorklistUrlInEmail="false">
      <action name="ASSIGN" recipient="ASSIGNEES"><![CDATA[concat(string('Task '),
/task:task/task:title, string(' requires your attention.'))]]></action>
      <action name="COMPLETE" recipient="CREATOR"><![CDATA[concat(string('Task
'), /task:task/task:title, string(' requires your attention.'))]]></action>
      <action name="ERROR" recipient="OWNER"><![CDATA[concat(string('Task '),
/task:task/task:title, string(' requires your attention.'))]]></action>

<groupNotificationConfiguration>SEND_INDIVIDUAL_NOTIFICATION</groupNotificatio
nConfiguration>
    </notification>
  </routingSlip>
  <workflowConfiguration
xmlns="http://xmlns.oracle.com/bpel/workflow/configuration"

xmlns:ns0="http://xmlns.oracle.com/ThesisOrderFulfillment/OrderFulfillmentHumanT
ask/OrderFulfillment">
```


Πανεπιστήμιο Πειραιώς

```
    <outcomes>
      <outcome>YES</outcome>
      <outcome>NO</outcome>
    </outcomes>
    <payload xmlSchemaDefinition="xsd/OrderFullfilmentPayload.xsd">
      <messageAttribute name="Order" attributeType="COMPLEX_TYPE"
                type="ns0:OrderProducts2" updatable="false"
                external="false"/>
    </payload>
    <bpelEventListener>false</bpelEventListener>
    <bpelNoCallbacks>false</bpelNoCallbacks>
    <showCompleteGraphicalHistory>true</showCompleteGraphicalHistory>
    <reevalTranslatablesOnUpdate>false</reevalTranslatablesOnUpdate>
    <preActionMandatoryUserSteps/>
    <allowInitiatorEditParticipants>true</allowInitiatorEditParticipants>
    <allowParticipantsEditParticipants>false</allowParticipantsEditParticipants>
    <globalCreationTask>false</globalCreationTask>

<taskFlowFileLocation>file:/C:/JDeveloper/mywork/ThesisOrderFulfillment/OrderFulf
illmentHumanTaskInterface/public_html/WEB-
INF/OrderFullfilment_TaskFlow.xml</taskFlowFileLocation>
    <enableAutoClaim>true</enableAutoClaim>
    <workflowConditions/>
  </workflowConfiguration>
</taskDefinition>
```

6.12 Κώδικας Human Task διαδικασίας BPEL παραλαβής προϊόντων από προμηθευτές

Κώδικας 58: InsertSupplierOrderWhenIsReceived.bpel

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<!--
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
 Oracle JDeveloper BPEL Designer

 Created: Thu Feb 16 21:13:31 EET 2012
 Author:  Epameinondas Nikolopoulos
 Type: BPEL 2.0 Process
 Purpose: One Way BPEL Process
////////////////////////////////////////////////////////////////////////////////
```


Πανεπιστήμιο Πειραιώς

```
////////////////////////////////////////////////////////////////////////
-->
<process name="InsertSupplierOrderWhenIsReceived"

targetNamespace="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHumanTask/InsertSupplierOrderWhenIsReceived"
        xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"

xmlns:client="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHumanTask/InsertSupplierOrderWhenIsReceived"
        xmlns:ora="http://schemas.oracle.com/xpath/extension"
        xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
      xmlns:bpelx="http://schemas.oracle.com/bpel/extension"
      xmlns:ns1="http://xmlns.oracle.com/bpel/workflow/taskService"
      xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.Xpath20"

xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.ExtFunc"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue"
      xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
      xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
      xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"
      xmlns:xdk="http://schemas.oracle.com/bpel/extension/xpath/function/xdk"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions"
      xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap"
      xmlns:task="http://xmlns.oracle.com/bpel/workflow/task"
      xmlns:wfcommon="http://xmlns.oracle.com/bpel/workflow/common"
      xmlns:ns2="http://webservice.brandname.store.com/"
      xmlns:wf="http://schemas.oracle.com/bpel/extensions/workflow"
      xmlns:ns3="http://xmlns.oracle.com/bpel/workflow/TaskEvidenceService"
      xmlns:ns4="http://xmlns.oracle.com/ias/pcbpel/NotificationService"
      xmlns:ns5="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:ns6="http://xmlns.oracle.com/bpel/workflow/routingSlip">

 <import
namespace="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHumanTask/InsertSupplierOrderWhenIsReceived"
location="InsertSupplierOrderWhenIsReceived.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
```


Πανεπιστήμιο Πειραιώς

```
  <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
    PARTNERLINKS
    List of services participating in this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
 -->
 <partnerLinks>
  <!--
    The 'client' role represents the requester of this service. It is
    used for callback. The location and correlation information associated
    with the client role are automatically set using WS-Addressing.
  -->
  <partnerLink name="insertsupplierorderwhenisreceived_client"
partnerLinkType="client:InsertSupplierOrderWhenIsReceived"
myRole="InsertSupplierOrderWhenIsReceivedProvider"/>
  <partnerLink name="Repository" partnerLinkType="ns2:Service1"
          partnerRole="RepositoryWS"/>
  <partnerLink name="NotificationService1"
          partnerLinkType="ns4:NotificationServiceLink"
          partnerRole="NotificationServiceProvider"/>
  <partnerLink name="NotificationService2"
          partnerLinkType="ns4:NotificationServiceLink"
          partnerRole="NotificationServiceProvider"/>
  <partnerLink name="SubmitThatSupplierOrderReceived.TaskService_1"
          partnerLinkType="ns1:TaskService" partnerRole="TaskService"
          myRole="TaskServiceCallbackListener"/>
 </partnerLinks>

 <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
    VARIABLES
    List of messages and XML documents used within this BPEL process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
 -->
 <variables>
  <!-- Reference to the message passed as input during initiation -->
  <variable name="inputVariable"
messageType="client:InsertSupplierOrderWhenIsReceivedRequestMessage"/>
```


Πανεπιστήμιο Πειραιώς

```
  <variable name="Invoke1_updateProductsToOrderStatus_InputVariable"
        messageType="ns2:updateProductsToOrderStatus"/>
  <variable name="Invoke1_updateProductsToOrderStatus_OutputVariable"
        messageType="ns2:updateProductsToOrderStatusResponse"/>
  <variable name="SubmitThatSupplierOrderReceived1_globalVariable"
        messageType="ns1:taskMessage"/>
  <variable name="SubmitThatSupplierOrderReceived2_globalVariable"
        messageType="ns1:taskMessage"/>
 </variables>

 <!--

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
   ORCHESTRATION LOGIC
   Set of activities coordinating the flow of messages across the
   services integrated within this business process

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
 -->
 <sequence name="main">

  <!-- Receive input from requestor. (Note: This maps to operation defined in
InsertSupplierOrderWhenIsReceived.wsdl) -->
  <receive name="receiveInput"
partnerLink="insertsupplierorderwhenisreceived_client"
portType="client:InsertSupplierOrderWhenIsReceived"
        operation="SendSupplierOrder" variable="inputVariable"
createInstance="yes"/>
  <scope name="SubmitThatSupplierOrderReceived2"
      xmlns:wf="http://schemas.oracle.com/bpel/extensions/workflow"
      wf:key="SubmitThatSupplierOrderReceived1_globalVariable">
   <bpelx:annotation>
    <bpelx:pattern patternName="bpelx:workflow"/>
   </bpelx:annotation>
   <variables>
    <variable name="initiateTaskInput"
          messageType="ns1:initiateTaskMessage"/>
    <variable name="initiateTaskResponseMessage"
          messageType="ns1:initiateTaskResponseMessage"/>
   </variables>
   <sequence>
    <assign name="SubmitThatSupplierOrderReceived2_AssignTaskAttributes">
     <copy>
      <from>number(3)</from>
```


Πανεπιστήμιο Πειραιώς

```
        <to variable="initiateTaskInput"
part="payload"><query>task:task/task:priority</query></to>
      </copy>
      <copy>
       <from><literal><payload
xmlns="http://xmlns.oracle.com/bpel/workflow/task">
 <OrderID xmlns="http://xmlns.oracle.com/bpel/workflow/task"/>
 <OrderPrice xmlns="http://xmlns.oracle.com/bpel/workflow/task"/>
</payload></literal></from>
        <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload</query></to>
      </copy>
      <copy>
       <from variable="inputVariable"
part="payload"><query>client:SupplierOrderId</query></from>
        <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload/task:OrderID</query></to>
      </copy>
      <copy>
       <from variable="inputVariable"
part="payload"><query>client:SupplierOrderPrice</query></from>
        <to variable="initiateTaskInput"
part="payload"><query>task:task/task:payload/task:OrderPrice</query></to>
      </copy>
      <copy>
       <from>concat(string('Supplier OrderID: '),
$inputVariable.payload/client:SupplierOrderId)</from>
        <to variable="initiateTaskInput"
part="payload"><query>task:task/task:title</query></to>
      </copy>
    </assign>
    <invoke name="initiateTask_SubmitThatSupplierOrderReceived2"
         partnerLink="SubmitThatSupplierOrderReceived.TaskService_1"
         operation="initiateTask" portType="ns1:TaskService"
         inputVariable="initiateTaskInput"
         outputVariable="initiateTaskResponseMessage"/>
    <receive name="receiveCompletedTask_SubmitThatSupplierOrderReceived2"
         partnerLink="SubmitThatSupplierOrderReceived.TaskService_1"
         portType="ns1:TaskServiceCallback" operation="onTaskCompleted"
         variable="SubmitThatSupplierOrderReceived1_globalVariable"
         createInstance="no"/>
   </sequence>
  </scope>
  <if name="CheckIfSupplierOrderReceived">
   <documentation>Task outcome is YES</documentation>
   <bpelx:annotation>
```


Πανεπιστήμιο Πειραιώς

```
      <bpelx:pattern patternName="Task outcome is YES"/>
    </bpelx:annotation>

<condition>$SubmitThatSupplierOrderReceived1_globalVariable.payload/task:systemA
ttributes/task:outcome = 'YES'</condition>
    <sequence>
     <assign name="InputForUpdatingOrderProducts">
      <copy>
       <from>$inputVariable.payload/client:SupplierOrderId</from>

<to>$Invoke1_updateProductsToOrderStatus_InputVariable.parameters/SupplierOrder
Id</to>
      </copy>
     </assign>
     <invoke name="UpdateOrderProductStatus" partnerLink="Repository"
          portType="ns2:RepositoryWS"
          operation="updateProductsToOrderStatus"
          inputVariable="Invoke1_updateProductsToOrderStatus_InputVariable"
          outputVariable="Invoke1_updateProductsToOrderStatus_OutputVariable"
          bpelx:invokeAsDetail="no"/>
    </sequence>
    <elseif>
     <documentation>Task outcome is NO</documentation>
     <bpelx:annotation>
      <bpelx:pattern patternName="Task outcome is NO"/>
     </bpelx:annotation>

<condition>$SubmitThatSupplierOrderReceived1_globalVariable.payload/task:systemA
ttributes/task:outcome = 'NO'</condition>
      <sequence>
       <scope name="Email1">
        <bpelx:annotation>
         <bpelx:pattern patternName="bpelx:email"/>
         <bpelx:general>
          <bpelx:property
name="userLabel">EmailOrderSupportStaff1</bpelx:property>
         </bpelx:general>
        </bpelx:annotation>
        <variables>
         <variable name="varNotificationReq"
               messageType="ns4:EmailNotificationRequest"/>
         <variable name="varNotificationResponse"
               messageType="ns4:ArrayOfResponse"/>
        <variable name="NotificationServiceFaultVariable"
               messageType="ns4:NotificationServiceErrorMessage"/>
        </variables>
```


Πανεπιστήμιο Πειραιώς

```
        <sequence name="Sequence1">
         <assign name="EmailParamsAssign">
          <copy>
           <from>string('Default')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:FromAccountName</query></to>
          </copy>
          <copy>
           <from>string('')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Bcc</query></to>
          </copy>
          <copy>
           <from>string('')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Cc</query></to>
          </copy>
          <copy>
           <from>string('')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:ReplyToAddress</query></to>
          </copy>
          <copy>
           <from>concat(string('Error for Supplier Order '),
$inputVariable.payload/client:SupplierOrderId)</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Subject</query></to>
          </copy>
          <copy>
           <from>string('thesiscompanyb@mailinator.com')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:To</query></to>
          </copy>
          <copy>
           <from>string('Please resolve this error')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Content/ns4:ContentBody</query></to>
          </copy>
          <copy>
           <from>string('text/html; charset=UTF-8')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Content/ns4:MimeType</query></to>
          </copy>
         </assign>
         <invoke name="InvokeNotificationService"
              portType="ns4:NotificationService"
```


Πανεπιστήμιο Πειραιώς

```
            partnerLink="NotificationService1"
            inputVariable="varNotificationReq"
            outputVariable="varNotificationResponse"
            operation="sendEmailNotification"/>
       </sequence>
      </scope>
     </sequence>
    </elseif>
    <else>
     <sequence>
      <scope name="Email2">
       <bpelx:annotation>
        <bpelx:pattern patternName="bpelx:email"/>
        <bpelx:general>
         <bpelx:property
name="userLabel">EmailOrderSupportStaff2</bpelx:property>
        </bpelx:general>
       </bpelx:annotation>
       <variables>
        <variable name="varNotificationReq"
              messageType="ns4:EmailNotificationRequest"/>
        <variable name="varNotificationResponse"
              messageType="ns4:ArrayOfResponse"/>
        <variable name="NotificationServiceFaultVariable"
              messageType="ns4:NotificationServiceErrorMessage"/>
       </variables>
       <sequence name="Sequence2">
        <assign name="EmailParamsAssign">
         <copy>
          <from>string('Default')</from>
          <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:FromAccountName</query></to>
         </copy>
         <copy>
          <from>string('')</from>
          <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Bcc</query></to>
         </copy>
         <copy>
          <from>string('')</from>
          <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Cc</query></to>
         </copy>
         <copy>
          <from>string('')</from>
          <to variable="varNotificationReq"
```


Πανεπιστήμιο Πειραιώς

```
part="EmailPayload"><query>ns4:ReplyToAddress</query></to>
          </copy>
          <copy>
           <from>concat(string('Error for Supplier Order '),
$inputVariable.payload/client:SupplierOrderId)</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Subject</query></to>
          </copy>
          <copy>
           <from>string('thesiscompanyb@mailinator.com')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:To</query></to>
          </copy>
          <copy>
           <from>string('Please resolve this error')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Content/ns4:ContentBody</query></to>
          </copy>
          <copy>
           <from>string('text/html; charset=UTF-8')</from>
           <to variable="varNotificationReq"
part="EmailPayload"><query>ns4:Content/ns4:MimeType</query></to>
          </copy>
         </assign>
         <invoke name="InvokeNotificationService"
              portType="ns4:NotificationService"
              partnerLink="NotificationService2"
              inputVariable="varNotificationReq"
              outputVariable="varNotificationResponse"
              operation="sendEmailNotification"/>
       </sequence>
      </scope>
     </sequence>
    </else>
   </if>
 </sequence>
</process>
```

Κώδικας 59: InsertSupplierOrderWhenIsReceived.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema attributeFormDefault="unqualified"
        elementFormDefault="qualified"
        targetNamespace="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/Suppli
erOrderHumanTask/InsertSupplierOrderWhenIsReceived"
        xmlns="http://www.w3.org/2001/XMLSchema">
        <element name="process">
```


Πανεπιστήμιο Πειραιώς

```
                <complexType>
                        <sequence>
                                <element name="SupplierOrderId" type="string"
maxOccurs="1"/>
  <element name="SupplierOrderPrice" maxOccurs="1" type="float"/>
 </sequence>
                </complexType>
        </element>
</schema>
```

Κώδικας 60: SubmitThatSupplierOrderReceived.task

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<taskDefinition
targetNamespace="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrde
rHumanTask/SubmitThatSupplierOrderReceived"
xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functi
ons.Xpath20" xmlns:ora="http://schemas.oracle.com/xpath/extension"
xmlns:oraext="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.func
tions.ExtFunc" xmlns:task="http://xmlns.oracle.com/bpel/workflow/task"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://xmlns.oracle.com/bpel/workflow/taskDefinition"
          xmlns:bpel2="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
          xmlns:ns0="http://xmlns.oracle.com/bpel/workflow/common"

xmlns:dvm="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.LookupValue
"

xmlns:evidence="http://xmlns.oracle.com/bpel/workflow/TaskEvidenceService"
          xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
          xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
          xmlns:bpm="http://xmlns.oracle.com/bpmn20/extensions"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXP
athFunctions">
  <name>SubmitThatSupplierOrderReceived</name>

<id>${domain_id}_${process_id}_${process_revision}_SubmitThatSupplierOrderReceive
d</id>
  <title type="XPATH"><value>string('Order ')</value></title>
  <priority>3</priority>
  <description>Submit an order from one of the suppliers when it is
received.</description>
  <process processId="" processVersion=""/>
  <routingSlip xmlns="http://xmlns.oracle.com/bpel/workflow/routingSlip">
    <globalConfiguration>
      <applicationContext></applicationContext>
```


Πανεπιστήμιο Πειραιώς

```
      <taskOwner type="STATIC"
identityType="user">supplierorderreceiver</taskOwner>
      <expirationDuration duration="P10D" type="STATIC"
                  useBusinessCalendar="true"/>
      <sharePayloadConfiguration>
        <type>USE_SYSTEM_WIDE_GLOBAL_CONFIGURATION</type>
      </sharePayloadConfiguration>
    </globalConfiguration>
    <participants isAdhocRoutingSupported="false">
      <stage name="Stage1">
        <participant name="RepositoryStuff">
          <resource type="STATIC"
identityType="user">supplierorderreceiver</resource>
        </participant>
      </stage>
    </participants>
    <onErrorParticipant>
      <resource type="STATIC" identityType="user">supplierorderreceiver</resource>
    </onErrorParticipant>
    <notification includeTaskAttachments="false" actionable="false"
            secureNotifications="false" hideWorklistUrlInEmail="false">
      <action name="ASSIGN" recipient="ASSIGNEES"><![CDATA[concat(string('Task '),
/task:task/task:title, string(' requires your attention.'))]]></action>
      <action name="COMPLETE" recipient="CREATOR"><![CDATA[concat(string('Task
'), /task:task/task:title, string(' requires your attention.'))]]></action>
      <action name="ERROR" recipient="OWNER"><![CDATA[concat(string('Task '),
/task:task/task:title, string(' requires your attention.'))]]></action>

<groupNotificationConfiguration>SEND_INDIVIDUAL_NOTIFICATION</groupNotificatio
nConfiguration>
    </notification>
  </routingSlip>
  <workflowConfiguration
xmlns="http://xmlns.oracle.com/bpel/workflow/configuration"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <outcomes>
      <outcome>YES</outcome>
      <outcome>NO</outcome>
    </outcomes>
    <payload
xmlSchemaDefinition="xsd/SubmitThatSupplierOrderReceivedPayload.xsd">
      <messageAttribute name="OrderID" attributeType="SIMPLE_TYPE"
                  type="xsd:string" updatable="false"
                  external="false"/>
      <messageAttribute name="OrderPrice" attributeType="SIMPLE_TYPE"
                  type="xsd:float" updatable="false" external="false"/>
```


Πανεπιστήμιο Πειραιώς

```
    </payload>
    <bpelEventListener>false</bpelEventListener>
    <bpelNoCallbacks>false</bpelNoCallbacks>
    <showCompleteGraphicalHistory>true</showCompleteGraphicalHistory>
    <reevalTranslatablesOnUpdate>false</reevalTranslatablesOnUpdate>
    <preActionMandatoryUserSteps/>
    <allowInitiatorEditParticipants>false</allowInitiatorEditParticipants>
    <allowParticipantsEditParticipants>false</allowParticipantsEditParticipants>
    <globalCreationTask>false</globalCreationTask>

<taskFlowFileLocation>file:/C:/JDeveloper/mywork/ThesisSupplierOrderSubmit/Supp
lierOrdetHumanTaskInterface/public_html/WEB-
INF/SubmitThatSupplierOrderReceived_TaskFlow.xml</taskFlowFileLocation>
    <enableAutoClaim>true</enableAutoClaim>
    <workflowConditions/>
  </workflowConfiguration>
</taskDefinition>
```

Κώδικας 61: composite.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 1.0 at [2/16/12 9:12 PM]. -->
<composite name="SupplierOrderHumanTask"
        revision="1.0"
        label="2012-02-16_21-12-38_227"
        mode="active"
        state="on"
        xmlns="http://xmlns.oracle.com/sca/1.0"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
        xmlns:ui="http://xmlns.oracle.com/soa/designer/">
 <import
namespace="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHum
anTask/InsertSupplierOrderWhenIsReceived"
       location="InsertSupplierOrderWhenIsReceived.wsdl" importType="wsdl"/>
 <import namespace="http://xmlns.oracle.com/bpel/workflow/taskService"
       location="oramds:/soa/shared/workflow/TaskServiceInterface.wsdl"
       importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"

location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"
       importType="wsdl"/>
 <import namespace="http://webservice.brandname.store.com/"
       location="RepositoryWSWrapper1.wsdl" importType="wsdl"/>
 <import namespace="http://xmlns.oracle.com/ias/pcbpel/NotificationService"
       location="NotificationService.wsdl" importType="wsdl"/>
```


Πανεπιστήμιο Πειραιώς

```
 <service name="insertsupplierorderwhenisreceived_client_ep"
       ui:wsdlLocation="InsertSupplierOrderWhenIsReceived.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHuman
Task/InsertSupplierOrderWhenIsReceived#wsdl.interface(InsertSupplierOrderWhenIs
Received)"/>
  <binding.ws
port="http://xmlns.oracle.com/ThesisSupplierOrderSubmit/SupplierOrderHumanTask
/InsertSupplierOrderWhenIsReceived#wsdl.endpoint(insertsupplierorderwhenisreceiv
ed_client_ep/InsertSupplierOrderWhenIsReceived_pt)"/>
 </service>
 <component name="InsertSupplierOrderWhenIsReceived" version="2.0">
  <implementation.bpel src="InsertSupplierOrderWhenIsReceived.bpel"/>
 </component>
 <component name="SubmitThatSupplierOrderReceived">
  <implementation.workflow src="SubmitThatSupplierOrderReceived.task"/>
 </component>
 <reference name="Repository"

ui:wsdlLocation="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WS
DL">
  <interface.wsdl
interface="http://webservice.brandname.store.com/#wsdl.interface(RepositoryWS)"/>
  <binding.ws
port="http://webservice.brandname.store.com/#wsdl.endpoint(RepositoryWS/Reposit
oryWSPort)"

location="http://176.34.182.139:8888/ThesisOrderServices/RepositoryWS?WSDL"
          soapVersion="1.1">
    <property name="weblogic.wsee.wsat.transaction.flowOption"
          type="xs:string" many="false">WSDLDriven</property>
  </binding.ws>
 </reference>
 <reference name="NotificationService1"
       ui:wsdlLocation="NotificationService.wsdl">
  <interface.wsdl
interface="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.interface(Noti
ficationService)"/>
  <binding.wsif
port="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.endpoint(Notificat
ionService/JavaPort)"
          location="NotificationService.wsdl"/>
 </reference>
 <reference name="NotificationService2"
       ui:wsdlLocation="NotificationService.wsdl">
  <interface.wsdl
```


Πανεπιστήμιο Πειραιώς

```
interface="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.interface(Notification
Service)"/>
  <binding.wsif
port="http://xmlns.oracle.com/ias/pcbpel/NotificationService#wsdl.endpoint(NotificationSer
vice/JavaPort)"
          location="NotificationService.wsdl"/>
 </reference>
 <wire>
  <source.uri>insertsupplierorderwhenisreceived_client_ep</source.uri>

<target.uri>InsertSupplierOrderWhenIsReceived/insertsupplierorderwhenisreceived_client</target.uri>
 </wire>
 <wire>
  <source.uri>InsertSupplierOrderWhenIsReceived/Repository</source.uri>
  <target.uri>Repository</target.uri>
 </wire>
 <wire>
  <source.uri>InsertSupplierOrderWhenIsReceived/NotificationService1</source.uri>
  <target.uri>NotificationService1</target.uri>
 </wire>
 <wire>
  <source.uri>InsertSupplierOrderWhenIsReceived/NotificationService2</source.uri>
  <target.uri>NotificationService2</target.uri>
 </wire>
 <wire>

<source.uri>InsertSupplierOrderWhenIsReceived/SubmitThatSupplierOrderReceived.TaskService_1</source.uri>
  <target.uri>SubmitThatSupplierOrderReceived/TaskService</target.uri>
 </wire>
</composite>
```

6.13 Κώδικας SQL Βάσεων Δεδομένων

Κώδικας 62: store01db

```
CREATE DATABASE  IF NOT EXISTS `store01db` /*!40100 DEFAULT CHARACTER SET
utf8 */;
USE `store01db`;
-- MySQL dump 10.13  Distrib 5.5.16, for Win32 (x86)
--
-- Host: 176.34.182.139    Database: store01db
-- ------------------------------------------------------
-- Server version          5.5.20
```


Πανεπιστήμιο Πειραιώς

```
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `products_to_order`
--

DROP TABLE IF EXISTS `products_to_order`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `products_to_order` (
 `ORDER_ID` int(11) unsigned NOT NULL,
 `PRODUCT_ID` int(11) unsigned NOT NULL,
 `QUANTITY` int(11) NOT NULL,
 `SUPPLIER_ORDER_ID` varchar(20) DEFAULT NULL,
 `STATUS_ID` int(1) unsigned NOT NULL DEFAULT '1',
 PRIMARY KEY (`ORDER_ID`,`PRODUCT_ID`),
 KEY `ORDER_ID` (`ORDER_ID`),
 KEY `PRODUCT_ID` (`PRODUCT_ID`),
 KEY `Index_5` (`SUPPLIER_ORDER_ID`),
 KEY `FK_products_to_order_3` (`STATUS_ID`),
 CONSTRAINT `FK_products_to_order_1` FOREIGN KEY (`PRODUCT_ID`) REFERENCES
`products` (`PRODUCT_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION,
 CONSTRAINT `FK_products_to_order_2` FOREIGN KEY (`ORDER_ID`) REFERENCES
`orders` (`ORDER_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION,
 CONSTRAINT `FK_products_to_order_3` FOREIGN KEY (`STATUS_ID`) REFERENCES
`product_order_status` (`PRODUCT_ORDER_STATUS_ID`) ON DELETE NO ACTION ON
UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `addresses`
--
```


Πανεπιστήμιο Πειραιώς

```
DROP TABLE IF EXISTS `addresses`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `addresses` (
  `ADDRESS_ID` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `ADDRESS_STREET` varchar(45) NOT NULL,
  `ADDRESS_TOWN` varchar(45) NOT NULL,
  `ADDRESS_STATE` varchar(45) NOT NULL,
  `ADDRESS_POSTCODE` int(10) unsigned NOT NULL,
  `ADDRESS_RECIPIENT_NAME` varchar(45) NOT NULL,
  `USER_ID` int(11) unsigned NOT NULL,
  PRIMARY KEY (`ADDRESS_ID`),
  KEY `FK_addresses_1` (`USER_ID`),
  CONSTRAINT `FK_addresses_1` FOREIGN KEY (`USER_ID`) REFERENCES `users`
(`USER_ID`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `suppliers`
--

DROP TABLE IF EXISTS `suppliers`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `suppliers` (
  `SUPPLIER_ID` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `SUPPLIER_BRAND_NAME` varchar(45) NOT NULL,
  `SUPPLIER_PHONE1` varchar(45) NOT NULL,
  `SUPPLIER_PHONE2` varchar(45) DEFAULT NULL,
  `SUPPLIER_FAX` varchar(45) DEFAULT NULL,
  `SUPPLIER_EMAIL1` varchar(45) NOT NULL,
  `SUPPLIER_EMAIL2` varchar(45) DEFAULT NULL,
  `SUPPLIER_ADDRESS` varchar(45) NOT NULL,
  `SUPPLIER_TOWN` varchar(45) NOT NULL,
  `SUPPLIER_STATE` varchar(45) NOT NULL,
  `SUPPLIER_POSTCODE` varchar(45) NOT NULL,
  PRIMARY KEY (`SUPPLIER_ID`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `payment_methods`
--

DROP TABLE IF EXISTS `payment_methods`;
```


Πανεπιστήμιο Πειραιώς

```
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `payment_methods` (
  `PAYMENT_METHOD_ID` int(2) unsigned NOT NULL,
  `PAYMENT_METHOD_NAME` varchar(15) NOT NULL,
  PRIMARY KEY (`PAYMENT_METHOD_ID`) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `product_order_status`
--

DROP TABLE IF EXISTS `product_order_status`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `product_order_status` (
  `PRODUCT_ORDER_STATUS_ID` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `PRODUCT_ORDER_STATUS_NAME` varchar(45) NOT NULL,
  PRIMARY KEY (`PRODUCT_ORDER_STATUS_ID`) USING BTREE
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `products`
--

DROP TABLE IF EXISTS `products`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `products` (
  `PRODUCT_ID` int(11) unsigned NOT NULL,
  `PRODUCT_NAME` varchar(45) NOT NULL,
  `PRODUCT_DESCRIPTION` varchar(400) DEFAULT NULL,
  `PRODUCT_PRICE` float NOT NULL,
  `SUPPLIER_ID` int(11) unsigned NOT NULL,
  `REPOSITORY_QUANTITY` int(5) unsigned NOT NULL,
  `SUPPLIER_PRODUCT_ID` int(11) unsigned NOT NULL,
  PRIMARY KEY (`PRODUCT_ID`),
  KEY `KEY` (`SUPPLIER_ID`) USING BTREE,
  CONSTRAINT `FK_products_1` FOREIGN KEY (`SUPPLIER_ID`) REFERENCES `suppliers`
(`SUPPLIER_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
```


Πανεπιστήμιο Πειραιώς

```
-- Table structure for table `order_status`
--

DROP TABLE IF EXISTS `order_status`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `order_status` (
  `ORDER_STATUS_ID` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `ORDER_STATUS_NAME` varchar(20) NOT NULL,
  PRIMARY KEY (`ORDER_STATUS_ID`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `receipts`
--

DROP TABLE IF EXISTS `receipts`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `receipts` (
  `RECEIPT_ID` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `RECEIPT_TYPE_ID` int(2) unsigned NOT NULL,
  `RECEIPT_DATETIME` datetime NOT NULL,
  PRIMARY KEY (`RECEIPT_ID`) USING BTREE,
  KEY `FK_receipts_1` (`RECEIPT_TYPE_ID`),
  CONSTRAINT `FK_receipts_1` FOREIGN KEY (`RECEIPT_TYPE_ID`) REFERENCES
`receipt_type` (`RECEIPT_TYPE_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=36 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `receipt_type`
--

DROP TABLE IF EXISTS `receipt_type`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `receipt_type` (
  `RECEIPT_TYPE_ID` int(2) unsigned NOT NULL,
  `RECEIPT_TYPE_NAME` varchar(15) NOT NULL,
  PRIMARY KEY (`RECEIPT_TYPE_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
```

Πανεπιστήμιο Πειραιώς

```
-- Table structure for table `orders`
--

DROP TABLE IF EXISTS `orders`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `orders` (
 `ORDER_ID` int(11) unsigned NOT NULL AUTO_INCREMENT,
 `USER_ID` int(11) unsigned NOT NULL,
 `ORDER_PRICE` float NOT NULL,
 `ORDER_DATE` datetime NOT NULL,
 `ADDRESS_ID` int(11) unsigned NOT NULL,
 `PAYMENT_METHOD_ID` int(2) unsigned NOT NULL,
 `RECEIPT_ID` int(11) unsigned DEFAULT NULL,
 `COMMENT` varchar(160) DEFAULT NULL,
 `ORDER_STATUS_ID` int(2) unsigned NOT NULL DEFAULT '1',
 `SHIPPING_TRACKING_ID` varchar(45) DEFAULT NULL,
 PRIMARY KEY (`ORDER_ID`),
 KEY `USER_ID` (`USER_ID`),
 KEY `FK_orders_3` (`PAYMENT_METHOD_ID`),
 KEY `FK_orders_4` (`ORDER_STATUS_ID`),
 KEY `FK_orders_2` (`RECEIPT_ID`),
 KEY `Index_6` (`ADDRESS_ID`),
 CONSTRAINT `FK_orders_1` FOREIGN KEY (`USER_ID`) REFERENCES `users`
(`USER_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION,
 CONSTRAINT `FK_orders_2` FOREIGN KEY (`RECEIPT_ID`) REFERENCES `receipts`
(`RECEIPT_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION,
 CONSTRAINT `FK_orders_3` FOREIGN KEY (`PAYMENT_METHOD_ID`) REFERENCES
`payment_methods` (`PAYMENT_METHOD_ID`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
 CONSTRAINT `FK_orders_4` FOREIGN KEY (`ORDER_STATUS_ID`) REFERENCES
`order_status` (`ORDER_STATUS_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION,
 CONSTRAINT `FK_orders_5` FOREIGN KEY (`ADDRESS_ID`) REFERENCES `addresses`
(`ADDRESS_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=93 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `users`
--

DROP TABLE IF EXISTS `users`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `users` (
 `USER_ID` int(11) unsigned NOT NULL AUTO_INCREMENT,
```


Πανεπιστήμιο Πειραιώς

```
  `USER_NAME` varchar(45) NOT NULL,
  `USER_SURNAME` varchar(45) NOT NULL,
  `USER_EMAIL` varchar(45) NOT NULL,
  `USER_PASSWORD` varchar(45) NOT NULL,
  `USER_PHONE` varchar(45) DEFAULT NULL,
  `USER_MOBILE` varchar(45) DEFAULT NULL,
  `USER_TIN` varchar(10) DEFAULT '0',
  `USER_TAX_OFFICE` varchar(15) DEFAULT NULL,
  `USER_PROFESSION` varchar(15) DEFAULT NULL,
  PRIMARY KEY (`USER_ID`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
```

Πανεπιστήμιο Πειραιώς