

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ**  
**ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ**  
**ΣΠΟΥΔΩΝ ΣΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ**

**ΔΗΜΗΤΡΙΟΣ ΠΑΠΑΔΗΜΗΤΡΙΟΥ**

**Σύγχρονα εργαλεία και τεχνολογίες ανάπτυξης**  
**Πληροφοριακών Συστημάτων – Η περίπτωση του**  
**Microsoft .NET**

**Επιβλέπων Καθηγητής: Κωνσταντίνος Ταραμπάνης**  
**Εξεταστής: Γεώργιος Χαραμής**

**Θεσσαλονίκη 2003**

UNIVERSITY OF MACEDONIA  
MASTER'S IN INFORMATION SYSTEMS

**DIMITRIS PAPADIMITRIOU**

**Modern Tools and Technologies for the Development  
of Information Systems – The case of Microsoft .NET**

**Supervisor: Konstantinos Tarabanis**

**Examiner: Georgios Haramis**

Thessaloniki 2003

# Περιεχόμενα

Περίληψη .....	3
Summary .....	5
1. Εισαγωγή .....	7
1.1 Σκοπός της εργασίας .....	7
1.2 Microsoft .NET – Τι είναι; .....	7
1.3 Θέμα της Εργασίας .....	9
1.4 Απευθυνόμενο αναγνωστικό κοινό .....	10
2. Εισαγωγή στο .NET Framework.....	11
2.1 Common Language Runtime (CLR) .....	11
2.2 Class Libraries .....	12
3. Προγραμματίζοντας για το .NET Framework .....	16
3.1 Ένα απλό παράδειγμα σε VB7 και C#.....	16
3.2 Μια ματιά στο παρασκήνιο.....	19
3.3 Προγραμματίζοντας για διάφορα περιβάλλοντα .....	20
4. Visual Basic .NET .....	22
4.1 Inheritance .....	22
4.2 Free Threading .....	23
4.3 Garbage Collection .....	25
4.4 Παράλληλη χρήση με την Visual Basic 6 (Interoperability) .....	26
5. Διαχείριση Δεδομένων – ADO.NET.....	28
5.1 Φυσική Πηγή Δεδομένων .....	29
5.2 Data Providers .....	30
5.3 Data Sets .....	32
6. Web Services .....	35
6.1 Υπάρχουσες τεχνολογίες .....	35
6.2 Ορισμός των Web Services .....	37
6.3 Υλοποίηση Web Services με το Visual Studio .NET .....	41
6.4 Προτάσεις υλοποίησης λύσεων με Web Services .....	44
6.5 Παράδειγμα χρήσης του Web Service του Google .....	45
7. XML Schema.....	50
7.1 Αναπαράσταση και περιγραφή πληροφορίας σε XML.....	50
7.2 Τύποι Δεδομένων στο XML Schema .....	52
7.3 Namespaces .....	53
7.4 XML Schema στο .NET Framework και το Visual Studio .NET .....	56

8. ASP.NET .....	58
8.1 Ο ρόλος της ASP.NET στα σύγχρονα πληροφοριακά συστήματα.	61
8.2 Σημαντικοί όροι στην ASP.NET .....	62
8.3 Η εφαρμογή IBuySpy .....	66
9. Active Directory .....	74
9.1 Χαρακτηριστικά και πλεονεκτήματα του Microsoft Active Directory	77
9.2 Active Directory στο .NET Framework – System.DirectoryServices	79
10. Εφαρμογή στο χώρο του e-government.....	81
10.1 Ανάλυση του συστήματος CertServer .....	82
10.2 Η βάση δεδομένων.....	84
10.3 Πρόσβαση στη βάση δεδομένων - .NET Assembly και ADO.NET	86
10.4 Διαχείριση πιστοποιητικών – έκδοση κλειδιών ψηφιακών υπογραφών (Windows Application).....	90
10.5 Αποστολή και λήψη πιστοποιητικών - Web Services.....	93
10.6 Διάφορες λειτουργίες του .NET Framework και του Visual Studio .NET	94
10.7 Ένα σενάριο λειτουργίας του CertServer .....	99
11. Συμπεράσματα.....	102
11.1 Microsoft .NET και Sun J2EE.....	102
11.2 Μετάβαση στο .NET Framework.....	104
Ευρετήριο .....	106
Αναφορές .....	107
Βιβλιογραφία – Αρθρογραφία .....	107
Internet .....	108
Παραρτήματα .....	109

## **Περιεχόμενα του CD-ROM**

1. Η εργασία σε μορφή PDF
2. Microsoft .NET Framework SDK
3. Microsoft .NET Redistributable SP2
4. Παραδείγματα κώδικα που σχολιάζονται στην εργασία
5. Ολοκληρωμένες εφαρμογές που σχολιάζονται στην εργασία
6. Πρόσθετα πακέτα για το Visual Studio .NET και το .NET Framework
7. Εφαρμογή – Παράδειγμα στο χώρο του e-government
8. HTML σελίδα συνδέσμων με πηγές

## Περίληψη

Στην εργασία αυτή εξετάζονται νέες τεχνολογίες ανάπτυξης πληροφοριακών συστημάτων και προβάλλονται οι νέες προοπτικές, που αναδύονται με την ευρεία αποδοχή τους από το σύνολο σχεδόν του κόσμου της πληροφορικής. Οι εξέταση αυτών των τεχνολογιών γίνεται παράλληλα με την μελέτη υλοποίησής τους, με την βοήθεια των μέσων ανάπτυξης λογισμικού της πλατφόρμας Microsoft .NET, η οποία αρχίζει να αποκτά ολοένα και περισσότερους υποστηρικτές.

Αρχικά γίνεται μια παρουσίαση του κορμού των εργαλείων ανάπτυξης της πλατφόρμας, δηλαδή του .NET Framework, εξετάζοντας μερικές από τις σημαντικότερες πτυχές του, όπως είναι το Common Language Runtime και οι βιβλιοθήκες κλάσεων. Στη συνέχεια, σχολιάζονται οι νέες δυνατότητες που προκύπτουν, κυρίως μέσω του .NET Framework, στη γλώσσα προγραμματισμού Visual Basic 7, μια από τις γλώσσες προγραμματισμού που υποστηρίζονται από την πλατφόρμα. Δεν λείπουν αναφορές και αντιπαραθέσεις με την εξίσου σημαντική γλώσσα προγραμματισμού της πλατφόρμας, την C# (C Sharp).

Η διαχείριση, περιγραφή και αναπαράσταση δεδομένων, στυλοβάτης στην ανάπτυξη και λειτουργία ενός πληροφοριακού συστήματος, εξετάζεται μέσω της τεχνολογίας ADO.NET του .NET Framework. Η μελέτη του ADO.NET περιλαμβάνει και τις γλώσσες XML και XML Schema, που αποτελούν αναπόσπαστο μέρος της και ταυτόχρονα ευρέως αποδεκτά πρότυπα του World Wide Web Consortium (W3C).

Άλλο ένα πρότυπο του W3C, τα Web Services, θεμελιώδης έννοια του .NET και φιλόδοξη τεχνολογία επικοινωνίας και διασύνδεσης συστημάτων, εξετάζεται διεξοδικά. Αναλύονται τα πρωτόκολλα από τα οποία απαρτίζεται ένα Web Service, οι τρόποι που αυτά ικανοποιούνται μέσω του Microsoft Visual Studio .NET, αλλά και οι νέες προοπτικές που διανοίγονται με την υλοποίησή τους.

Η τεχνολογία ASP.NET, διάδοχος της πολύ επιτυχημένης ASP, αποτελεί μια νέα ουσιαστικά πλατφόρμα ανάπτυξης Web εφαρμογών. Η πληθώρα δυνατοτήτων που περιλαμβάνει και εξετάζονται και στην παρούσα εργασία, αλλάζει ριζικά τον τρόπο ανάπτυξης και εκμετάλλευσης αυτού του είδους

των εφαρμογών, οι οποίες κερδίζουν όλο και περισσότερο έδαφος έναντι των τυπικών Windows προγραμμάτων. Η ASP.NET φιλοδοξεί να καλύψει το σημαντικό κενό που υπήρχε στα εργαλεία ανάπτυξης διαδικτυακών εφαρμογών, σε σχέση με την πληθώρα πλούσιων εργαλείων ανάπτυξης Windows προγραμμάτων.

Μια ακόμα τεχνολογία που εξετάζεται είναι το Microsoft Active Directory και η χρήση του μέσω του .NET Framework. Η τεχνολογία αυτή, η οποία υλοποιείται μέσω λειτουργικών συστημάτων όπως τα Windows 2000 Server και .NET Server 2003, προσπαθεί να βάλει σε τάξη τη διαχείριση των πόρων ενός συστήματος υπολογιστών και χρηστών.

Τέλος, με σκοπό την παρουσίαση ενός μοντέλου αξιοποίησης των παραπάνω τεχνολογιών, γίνεται η μελέτη και η, εν μέρει, ανάπτυξη ενός πραγματικού συστήματος. Το σύστημα αυτό έχει ως στόχο του την αυτοματοποίηση ανταλλαγής πιστοποιητικών μεταξύ κυβερνητικών ή μη κυβερνητικών υπηρεσιών (εφορίες, ασφαλιστικούς οργανισμούς κ.α). Κάνοντας κατάλληλη χρήση των Web Services, του XML Schema, ψηφιακών υπογραφών και άλλων τεχνολογιών, παρουσιάζεται μια ιδέα που θα μπορούσε να δώσει τέλος στην ταλαιπωρία συλλογής και υποβολής των δικαιολογητικών, που απαιτούνται σε διάφορες συναλλαγές των πολιτών με τις εν λόγω υπηρεσίες.

## Summary

This dissertation examines new technologies for the development of Information Systems and introduces the new perspectives, which derive from their wide acceptance by the community of informatics. The inspection of these technologies is accompanied by their implementation using the Microsoft .NET platform.

The start is made by presenting the core tools of the .NET development platform, the .NET Framework. These core tools include the Common Language Runtime and the .NET Framework Class Libraries. A brief analysis of the new functions of Visual Basic 7 follows, as they mainly derive from the .NET Framework. Some of these functions are presented in confrontation with their implementation using C# (C Sharp), the other important programming language of the platform.

The manipulation, description and presentation of data, foundations of the development and operation of any information system, are examined through the ADO.NET technology of the .NET Framework. The revision of XML and XML Schema is included in this study. XML and XML Schema constitute integral parts of ADO.NET, and widely accepted standards recommended by the World Wide Web Consortium (W3C).

Another recommended standard of the W3C, fundamental concept of .NET and ambitious technology for system communication and system integration, the Web Services, is thoroughly examined. The analysis includes the protocols that constitute a Web Service, the ways these protocols are implemented using the Microsoft Visual Studio .NET and also the new perspectives that derive by their usage.

ASP.NET, successor of the successful ASP web authoring technology, actually comprises a new platform for the development of Web applications. The abundance of new functions, which are also examined in this dissertation, dramatically changes the way Web applications are developed and utilized. ASP.NET covers the significant gap among the tools use for the development of web and windows applications.

Another technology examined, is the Microsoft Active Directory and it's usage through the .NET Framework. This technology, which is implemented

using operating systems such as Windows 2000 Server and .NET Server 2003, undertakes the difficult task of managing the resources of medium and large scale computer systems.

Finally, for the purpose of presenting a model for the utilization of these technologies, this dissertation is concluded by the study and, partly, development of a real information system. The object of this system is the automatization of certificates exchange among government and/or non-government services (tax services, insurance organizations etc.). By putting together technologies such as Web Services, XML Schema and digital signatures, an idea is presented, that may put an end to the hardships of collecting and applying the certificates required for various transaction of the citizens.

# 1. Εισαγωγή

## 1.1 Σκοπός της εργασίας

Αυτή η εργασία αποτελεί αφορμή για εξέταση νέων τεχνολογιών, που αναδείχθηκαν τα τελευταία χρόνια με την ευρεία χρήση του Internet, αλλά και την αύξηση της επεξεργαστικής ισχύος των σύγχρονων υπολογιστών ή αντίστοιχα την συμπίεση του όγκου τους και τη χρήση τους σε όλο και μικρότερες σε φυσικό μέγεθος συσκευές. Η εξέταση των τεχνολογιών αυτών θα γίνει, χρησιμοποιώντας το Microsoft .NET.

Πριν όμως δοθούν περαιτέρω λεπτομέρειες για το περιεχόμενο της παρούσας εργασίας, θα ήταν σκόπιμο να οριοθετήσουμε την πλατφόρμα του .NET, περιγράφοντας κάποιες βασικές έννοιες. Το ενδεχόμενο να γίνει αναφορά σε συστήματα και τεχνολογίες, που είναι άγνωστες, ίσως επειδή δεν έκαναν ακόμα σημαντικά βήματα στην ελληνική πραγματικότητα, επιβάλλει κατά πάσα πιθανότητα τη συμβουλευτική χρήση σχετικών συνδέσμων στο Internet για περισσότερες πληροφορίες.

## 1.2 Microsoft .NET – Τι είναι;

Ο ορισμός, που δίνεται από την Microsoft για το .NET, είναι ο εξής: «ένα σύνολο τεχνολογιών λογισμικού για τη σύνδεση πληροφοριών, ανθρώπων, συστημάτων και συσκευών». Το Microsoft .NET δίνει στα παραπάνω στοιχεία τη δυνατότητα επικοινωνίας μέσω της σχεδίασης και ανάπτυξης ολοκληρωμένων (integrated) λύσεων, βασισμένων στα XML Web Services, τα οποία θα αναλυθούν λεπτομερέστερα στο Κεφάλαιο 6. Μολονότι μπορεί κανείς να θεωρήσει τα Web Services μόνον ως ένα τμήμα της φιλοσοφίας του .NET, θα καταλήξει ωστόσο στο συμπέρασμα ότι αποτελούν επίσης και τον συνδετικό κρίκο μεταξύ όλων των υπόλοιπων τεχνολογιών, που περιλαμβάνονται σε αυτό.

Ο αρχικός ορισμός, που δόθηκε, ίσως να φαίνεται αόριστος. Όταν αναφέρεται κανείς σε επικοινωνία ανθρώπων και υπολογιστών, στην εποχή των bugs και των κενών ασφάλειας, καταντά ίσως λίγο ρομαντικός. Για να αρχίσει όμως να αντιλαμβάνεται κανείς τη φιλοσοφία σύνδεσης «πληροφοριών, ανθρώπων, συστημάτων και συσκευών», θα πρέπει

παρατηρήσει τις βασικές οντότητες - στυλοβάτες του Microsoft .NET, οι οποίες είναι οι εξής:

- **Smart Clients** (προσωπικοί υπολογιστές, υπολογιστές χειρός, κινητά τηλέφωνα και γενικότερα ασύρματες συσκευές, Tablet PCs κ.α.) μέσω των οποίων αποκτά κανείς πρόσβαση σε κάποιο σύστημα. Για το .NET οι smart clients βασίζονται σε λειτουργικά συστήματα όπως Windows 2000/XP, Windows CE .NET ή Windows XP Embedded. (βλ. <http://www.microsoft.com/net/products/devices.asp>)
- **XML Web Services** μέσω των οποίων γίνεται η επικοινωνία και ανταλλαγή πληροφοριών μεταξύ των συστημάτων, ανεξάρτητα από τον τύπο του υπολογιστή στον οποίο εκτελούνται, από το λειτουργικό του σύστημα, όπως επίσης και από την πλατφόρμα προγραμματισμού στην οποία έχουν κατασκευαστεί (.NET ή όχι). (βλ. <http://www.microsoft.com/net/basics/xmlservices.asp> και κεφάλαιο 6)
- **Microsoft Servers**. Αποτελούν τη βάση για τη φιλοξενία και λειτουργία των .NET εφαρμογών. Μπορεί να πρόκειται για λειτουργικά συστήματα, όπως Windows 2000 Server Family ή Windows .NET Server 2003, αλλά και για εφαρμογές όπως Microsoft BizTalk Server, Microsoft Exchange Server, Microsoft Share Point Portal Server κλπ. (βλ. <http://www.microsoft.com/net/products/servers.asp>)
- **Εργαλεία Ανάπτυξης** με την βοήθεια των οποίων γίνεται η σχεδίαση και ανάπτυξη των εφαρμογών .NET. Πρόκειται για το .NET Framework, .NET Compact Framework και το Microsoft Visual Studio .NET κ.α. (βλ. <http://www.microsoft.com/net/products/tools.asp>)

Ειδοποιός διαφορά σε σχέση με προηγούμενες πλατφόρμες, είναι ο σαφής διαχωρισμός των διάφορων τεχνολογιών και εργαλείων σε διακριτά επίπεδα, π.χ. χαμηλό επίπεδο – σχεδίαση και ανάπτυξη λογισμικού, μεσαίο επίπεδο – διαχείριση και εκτέλεση, υψηλό επίπεδο – τελικοί χρήστες. Μελετώντας κανείς ξεχωριστά τα επιμέρους επίπεδα, μπορεί επίσης να διακρίνει και ένα σαφή διαχωρισμό στα εργαλεία ανάπτυξης. Ο προγραμματιστής, για

παράδειγμα, έχει τη δυνατότητα είτε να εμβαθύνει στα πρωτόκολλα, που απαρτίζουν τα Web Services και να δημιουργήσει εξελιγμένες εφαρμογές, είτε να δημιουργήσει πολύ εύκολα Web Services, γράφοντας μόνον μερικές γραμμές κώδικα (χρησιμοποιώντας το Visual Studio .NET) στη γλώσσα προγραμματισμού της επιλογής του.

Εύκολα θα διαπιστώσει κανείς ότι το Microsoft .NET - περισσότερο από κάθε άλλη πλατφόρμα ανάπτυξης λογισμικού που εμφανίστηκε έως σήμερα - μπορεί να διαχειρίζεται μεγάλο μέρος των εργασιών, που απαιτούνται για την ανάπτυξη διάφορων εφαρμογών, προσφέροντας έτσι στους προγραμματιστές και τους ειδικούς της πληροφορικής περισσότερο χρόνο για την καλύτερη ανάλυση των επιχειρησιακών κανόνων και την ικανοποίηση των αναγκών του υπό ανάπτυξης συστήματος.

### **1.3 Θέμα της Εργασίας**

Βασικό θέμα της εργασίας θα αποτελέσουν τα 'Εργαλεία Ανάπτυξης', δηλαδή το .NET Framework. Μετά την παρουσίαση των βασικών οντοτήτων του .NET, γίνεται ίσως πλέον καλύτερα αντιληπτός, ο τρόπος με τον οποίο οριοθετούνται τα εργαλεία ανάπτυξης καθώς και η θέση τους στο Microsoft .NET.

Συγκεκριμένα, θα εξεταστούν οι παρακάτω τεχνολογίες, οι οποίες αποτελούν και τον βασικό κορμό του .NET Framework:

- Class Libraries και Common Language Runtime,
- Visual Basic .NET (VB 7),
- ADO.NET,
- Web Services,
- XML Schema,
- ASP.NET και
- Active Directory, η οποία αποτελεί σημαντικό σημείο για την ολοκλήρωση των συστημάτων που αναπτύσσονται με το .NET Framework.

## **1.4 Απευθυνόμενο αναγνωστικό κοινό**

Η εργασία αυτή απευθύνεται στον προγραμματιστή ή/ και στον αναλυτή πληροφοριακών συστημάτων που θέλει να γνωρίσει το Microsoft .NET και τις δυνατότητες που προσφέρει στην ανάπτυξη πληροφοριακών συστημάτων μεσαίας και μεγάλης κλίμακας.

Τα κεφάλαια που ακολουθούν επεξηγούν κάθε μια από τις τεχνολογίες που αναφέρθηκαν στην προηγούμενη παράγραφο, με σκοπό την κατανόηση των βασικών τους πτυχών. Η επεξήγηση αυτή, όπου είναι δυνατό, συνοδεύεται με ανάλογα απλά παραδείγματα κώδικα, γραμμένα κυρίως σε Visual Basic .NET ή C#. Γι αυτό το λόγο θεωρείται απαραίτητη η ύπαρξη κάποιας βασικής εμπειρίας σε κάποια γλώσσα προγραμματισμού, κατά προτίμηση στην Visual Basic.

Ειδικά για το κεφάλαιο 4, όπου γίνεται η παρουσίαση της Visual Basic .NET αντιπαραβάλλοντάς την με την προηγούμενη έκδοσή της, ο αναγνώστης θα πρέπει να κατέχει αρκετά τον προγραμματισμό σε Visual Basic 6.

## 2. Εισαγωγή στο .NET Framework

Το βασικό σώμα εργαλείων ανάπτυξης του Microsoft .NET είναι το .NET Framework, το οποίο αποτελεί ταυτόχρονα και την υποδομή της πλατφόρμας, καθώς συνιστά το πλαίσιο προγραμματισμού της Microsoft για τη σχεδίαση και ανάπτυξη συστημάτων, που βασίζονται σε τεχνολογίες, όπως Web Services, Internet και Windows.

Το .NET Framework προσφέρει πλήρως ελεγχόμενο (managed), προστατευμένο και πλούσια εξοπλισμένο περιβάλλον εκτέλεσης εφαρμογών (runtime environment), απλουστευμένο περιβάλλον ανάπτυξης και άρρηκτη σύνδεση με ένα εύρος γλωσσών προγραμματισμού.

Υπάρχουν τρεις εκδόσεις του .NET Framework. Η έκδοση SDK (Software Development Kit), περιλαμβάνει όλα τα απαραίτητα εργαλεία για τη δημιουργία, αποσφαλμάτωση (debugging) και εκτέλεση εφαρμογών .NET, καθώς και πλήρη τεκμηρίωση σε ηλεκτρονική μορφή. Η έκδοση προς διανομή (Redistributable Version) περιλαμβάνει μόνο τα απαραίτητα στοιχεία για την εκτέλεση εφαρμογών .NET σε ένα υπολογιστή. Τέλος, η compact έκδοση (.NET Framework Compact Edition) αποτελεί μια μικρότερη μορφή της προηγούμενης, που προορίζεται για χρήση σε υπολογιστές με λειτουργικό σύστημα Windows CE (συνήθως για υπολογιστές χειρός - palmtop).

Τα δυο βασικά μέρη του .NET Framework είναι το Common Language Runtime και οι Βιβλιοθήκες Κλάσεων (Class Libraries).

### 2.1 Common Language Runtime (CLR)

Το CLR αποτελεί το επίπεδο χειρισμού του κώδικα, που γράφεται από τους προγραμματιστές. Διαχειρίζεται ζητήματα όπως η ασφάλεια (του κώδικα και των πόρων του συστήματος), η μνήμη, οι διεργασίες (processes) κ.α. Είναι κοινό για όλες τις γλώσσες προγραμματισμού του .NET - όπως η Visual Basic .NET (VB 7), η C# (διαβάζεται C Sharp) και η J# - και είναι παράλληλα επεκτάσιμο, ώστε να χρησιμοποιείται και με άλλες γλώσσες (υπό ανάπτυξη βρίσκονται επίσης γλώσσες όπως η Perl, η COBOL, η Fortran και η Python). Αξιοσημείωτο είναι επίσης ότι το CLR αντικαθιστά τα υπάρχοντα επίπεδα

εκτέλεσης εφαρμογών, όπως το COM (Component Object Model), τα COM+ (MTS/DCOM) κλπ.

Ο κώδικας, που γράφεται με σκοπό να εκτελεστεί από το CLR, ονομάζεται ελεγχόμενος (managed code). Τα πλεονεκτήματα, που συγκεντρώνει ο ελεγχόμενος κώδικας, είναι η δια-γλωσσική ολοκλήρωση (cross-language integration), ο δια-γλωσσικός χειρισμός σφαλμάτων (cross-language error handling), η ασφάλεια, η υποστηριζόμενη εγκατάσταση και ο έλεγχος εκδόσεων, όπως επίσης και άλλα χαρακτηριστικά που φέρει το .NET Framework.

Η δυνατότητα άμεσης διάδρασης μεταξύ των γλωσσών, που υποστηρίζονται από το CLR, επιτρέπει τη συνεργασία και χρήση έτοιμων τμημάτων κώδικα μεταξύ προγραμματιστών, που έχουν διαφορετικές εμπειρίες σε διάφορες γλώσσες. Ουσιαστικά, το γεγονός αυτό σημαίνει ότι ένα ολοκληρωμένο πρόγραμμα μπορεί να συνταχθεί σε δυο ή περισσότερες γλώσσες (ίσως το τυπικότερο παράδειγμα είναι η ταυτόχρονη χρήση VB7, C# και J#), εκμεταλλευόμενο κατά το βέλτιστο τρόπο τους ανθρώπινους πόρους μιας εταιρείας παραγωγής λογισμικού. Μια τέτοια μεθοδολογία, χωρίς το κέλυφος του CLR, θα δημιουργούσε προβλήματα διασύνδεσης των τμημάτων του κώδικα, καθώς και μείωση της απόδοσης της εφαρμογής, λόγω κλίσεων (call overhead) μεταξύ διαφορετικών πλαισίων εκτέλεσης κώδικα (runtime environments).

Στην εργασία αυτή η γλώσσα που κατά βάση θα χρησιμοποιηθεί είναι η Visual Basic .NET (VB7). Ωστόσο θα παρουσιαστούν και κάποια παραδείγματα κώδικα, γραμμένα σε γλώσσα C#, ούτως ώστε να φανούν οι ομοιότητες του κώδικα, όπως γράφεται στις δυο γλώσσες. Τέλος, θα γίνει ιδιαίτερη αναφορά στα νέα χαρακτηριστικά που παρουσιάζονται στην VB7, έτσι όπως αυτά αντλούνται από τη λειτουργικότητα του .NET Framework.

## **2.2 Class Libraries**

Το δεύτερο χαρακτηριστικό που καθορίζει τον χαρακτήρα του .NET Framework είναι οι βιβλιοθήκες κλάσεων. Πρόκειται για ένα σημαντικό αριθμό βιβλιοθηκών, που καθορίζουν τις διαθέσιμες ενσωματωμένες λειτουργίες του .NET Framework. Οι βιβλιοθήκες αυτές είναι σχηματισμένες σε μορφή αρχείων dll και για να είναι δυνατή η χρήση τους από ένα

πρόγραμμα που δημιουργείται με το .NET Framework, θα πρέπει το πρόγραμμα αυτό να διαθέτει τις αντίστοιχες παραπομπές (references). Η έννοια του reference είναι γνωστή σε όσους έχουν επαφή με την VB6, ενώ είναι παρόμοια με την χρήση του Import keyword στην C++.



**Σχήμα 1 - Δομή της assembly System.Security**

Σε αυτό το σημείο αξίζει να αναφερθεί ότι είναι η δυνατή η δημιουργία παραπομπών (references) και σε COM Components, δηλαδή σε dll που έχουν δημιουργηθεί με προγενέστερα συστήματα όπως η VB6, η C++ κ.α. Το γεγονός αυτό καθιστά δυνατή την χρήση υπαρχόντων εφαρμογών ή/ και κώδικα σε συνδυασμό με μια νέα εφαρμογή. Δοθείσης της ευκαιρίας, μπορεί να γίνει και πάλι αναφορά στην έννοια του managed code. Η προσθήκη μιας παραπομπής ενός COM Component και η εκτέλεσή του από μια .NET εφαρμογή, έχει ως αποτέλεσμα το Component αυτό να εκτελείται μέσα στο runtime πλαίσιο του συστήματος, κάτω από το οποίο αναπτύχθηκε. Για παράδειγμα, αν το component δημιουργήθηκε σε C++, θα εκτελεστεί μέσω του C++ Runtime Environment. Αυτός ο κώδικας δεν χαρακτηρίζεται ως ελεγχόμενος από το .NET Framework (managed code) και δεν φέρει τα χαρακτηριστικά, που αναφέρθηκαν στην προηγούμενη παράγραφο.

Οι managed βιβλιοθήκες του .NET Framework είναι γνωστές και ως assemblies. Στο Σχήμα 1 φαίνεται η δομή της assembly System.Security η

οποία περιλαμβάνει λειτουργικότητα για την ασφάλεια των δεδομένων που διαχειρίζεται μια εφαρμογή, καθώς διακρίνονται οι κλάσεις, τα interfaces και οι σταθερές (constants) που διαθέτει η assembly, όπως αυτές είναι διαθέσιμες σε κάθε εφαρμογή που έχει παραπομπή στην συγκεκριμένη βιβλιοθήκη.

Από το προηγούμενο παράδειγμα, παρατηρεί κανείς ότι η ονοματολογία των στοιχείων που απαρτίζουν μια assembly ακολουθεί τη μορφή `AssemblyName.ClassName`, π.χ. `Security.PermissionSet`. Καθώς είναι επίσης δυνατό και το όνομα της assembly να περιλαμβάνει τελείες στο εσωτερικό του, π.χ. `System.Security`, μια κλάση της assembly μπορεί να έχει ονομασία του τύπου `System.Security.PermissionSet`. Το σύνολο των ονομάτων που είναι διαθέσιμα από μια assembly ονομάζεται namespace και είναι διαθέσιμο στην τεκμηρίωσή της (.NET Framework Documentation).

Άλλα παραδείγματα βιβλιοθηκών, που περιλαμβάνονται στο .NET Framework, φαίνονται στον πίνακα 1. Αυτές βρίσκονται σε αντιπαράθεση με τους πιο χαρακτηριστικούς και κοινούς τρόπους που υλοποιούνται με προγενέστερα συστήματα. Όπου δεν υπήρχαν παλαιότερες μορφές ή όπου δεν υπήρχε κάποιος συγκεκριμένος και κοινά αποδεκτός τρόπος για την υλοποίησή τους, τότε αυτές δεν αναφέρονται.

<b>Πεδίο</b>	<b>Namespace</b>	<b>Προγενέστερη Μορφή</b>
<i>Δίκτυα - Επικοινωνίες</i>	System.Net	Winsock <sup>1</sup>
<i>Διαχείριση Αρχείων</i>	System.IO	Scripting.FileSystem Object, Win32 API
<i>Διαχείριση Δεδομένων</i>	System.Data	ADO <sup>2</sup> (Active Data Objects)
<i>OLEDB και SQL Server</i>	System.Data.OleDb System.Data.SqlClient	ADO
<i>Ρυθμίσεις εφαρμογών</i>	System.Configuration	Registry Access
<i>Active Directory</i>	System.DirectoryServices	ADSI (Active Directory Services)

<sup>1</sup> Μια σειρά από βιβλιοθήκες που κάνουν χρήση των Windows Sockets για την εκμετάλλευση δικτύων.

<sup>2</sup> Components της Microsoft για την πρόσβαση σε βάσεις δεδομένων, τη λήψη και επεξεργασία δεδομένων κλπ.

		Interface)
<i>Σχεδίαση</i>	System.Drawing	Win32 API <sup>3</sup>
<i>COM+, DCOM</i>	System.EnterpriseServices	COM+ COM Library
<i>Παγκοσμιοποίηση εφαρμογών</i>	System.Globalization	Resource Files
<i>Distributed Applications</i>	System.Runtime.Remoting	-
<i>Κρυπτογραφία</i>	System.Security.Cryptography	Win32 Crypto API
<i>Threading</i>	System.Threading	-
<i>Web</i>	System.Web	Microsoft Internet Components <sup>4</sup>
<i>Windows Application</i>	System.Windows.Forms	-
<i>XML</i>	System.XML	XML Parsers <sup>5</sup>

**Πίνακας 1 - .NET Assemblies και προγενέστερες υλοποιήσεις**

Με μια πρώτη ματιά στον παραπάνω πίνακα, αντιλαμβάνεται κανείς εύκολα τη λειτουργικότητα των βιβλιοθηκών του .NET Framework, καθώς και τη λογική της ενοποίησης, που το χαρακτηρίζει. Αρκεί να σκεφτεί κανείς, τα χαρακτηριστικά των managed assemblies που προαναφέρθηκαν, καθώς και άλλα πλεονεκτήματα όπως τα ενοποιημένα εγχειρίδια χρήσης, η ομοιόμορφη αντιμετώπιση από τις κοινότητες του Internet, η κοινή χρήση τους από όλες τις γλώσσες προγραμματισμού του .NET Framework – οπότε και από κοινού χρήση από προγραμματιστές εξοικειωμένους με διαφορετικές γλώσσες, όπως η VB6 και η C++.

---

<sup>3</sup> αναφέρεται σε κλήσεις σε κλασικές βιβλιοθήκες dll των windows (π.χ. kernel32, advapi32.dll, user32.dll) για την εκτέλεση λειτουργιών που προσφέρονται από αυτές. Η χρήση του Win32 API ιδιαίτερα μέσω της Visual Basic ήταν πάντα εκτός της φιλοσοφίας της γλώσσας, καθιστώντας την δυσκολονόητη από τους προγραμματιστές σε VB, ενώ συχνά, κυρίως λόγω λάθους χρήσης, καθιστούσε της εφαρμογές ασταθείς ιδιαίτερα κατά την ανάπτυξή τους.

<sup>4</sup> Διάφορα components της Microsoft για την χρήση πρωτοκόλλων υψηλού επιπέδου του TCP/IP, όπως HTTP, SMTP, POP3 κλπ.

<sup>5</sup> Components για την ευκολότερη χρήση εγγράφων XML.

## 3. Προγραμματίζοντας για το .NET Framework

### 3.1 Ένα απλό παράδειγμα σε VB7 και C#

Σε αυτή την παράγραφο θα δοθεί ένα παραδείγματα κώδικα χρήσης του .NET Framework, σε Visual Basic 7 και C#, μέσω του CLR και των Class Libraries. Το παράδειγμα αυτό μπορεί να υλοποιηθεί χρησιμοποιώντας το .NET Framework SDK, το οποίο μπορεί κανείς να βρει και να αντιγράψει ελεύθερο από τον ιστοχώρο του MSDN<sup>6</sup> (<http://msdn.microsoft.com>) και το notepad των Windows για τη συγγραφή του κώδικα. Είναι προφανές πως αντίστοιχη μεθοδολογία δεν θα μπορούσε να χρησιμοποιηθεί σε κανονικές εφαρμογές, παρά μόνο σε πολύ μικρά προγράμματα.

Οι βασικές οντότητες στις οποίες γράφεται κώδικας στο .NET Framework είναι οι κλάσεις (classes), τόσο στην VB7 όσο και στην C# (επεκτάσεις αρχείων .vb και .cs αντίστοιχα). Ειδικά στην VB7, υπάρχει η δυνατότητα χρήσης modules, που προκύπτει μάλλον περισσότερο για λόγους συμβατότητας με τις προηγούμενες εκδόσεις, χωρίς αυτό να σημαίνει ότι τα modules χάνουν τη λειτουργικότητά τους<sup>7</sup> στην VB7.

Στο παρακάτω παράδειγμα δίνεται ένα απλό module σε VB7 και η αντίστοιχη κλάση σε C#. Ο κώδικας αυτός μπορεί να γραφεί στο notepad των Windows και αποθηκευτεί με επέκταση .vb ή .cs αντίστοιχα (π.χ. sample1.vb, sample1.cs). Η μετατροπή του κώδικα σε εκτελέσιμο αρχείο γίνεται με την μεταγλώττιση του (compilation). Για το σκοπό αυτό υπάρχει ο compiler της VB7 (vbc.exe) και της C# (csc.exe) αντίστοιχα, που περιλαμβάνονται στο .NET Framework SDK. Ξεκινώντας το Visual Studio .NET Command Prompt (αν έχει εγκατασταθεί το Visual Studio .NET από το

---

<sup>6</sup> Περιλαμβάνεται και στο συνοδευτικό CD-ROM αυτής της εργασίας

<sup>7</sup> Τα modules της VB δίνουν τη δυνατότητα σύνταξης και κλήσης κώδικα χωρίς πρώτα να δημιουργηθεί μια οντότητα ενός αντικειμένου (instance) που περιλαμβάνει τον κώδικα αυτό, όπως απαιτείται με μια κλάση. Δίνουν έτσι τη δυνατότητα διάσπασης και οργάνωσης του κώδικα σε πολλά φυσικά αρχεία (.bas σε προγενέστερες εκδόσεις, πλέον .vb), χωρίς την χρήση όμως περισσότερων γραμμών κώδικα για να χρησιμοποιηθεί. Αντίστοιχη λειτουργικότητα μπορεί να επιτευχθεί στην VB7 με τη χρήση του keyword Shared ή του keyword Static στην C# (βλ. VB7 ή C# Reference).

Start->Programs->Microsoft Visual Studio .NET) και γράφοντας μια από τις παρακάτω εντολές:

```
vbc.exe /t :exe /out :sample1.exe sample1.vb
csc.exe /t :exe /out :sample1.exe sample1.cs
```

δημιουργείται ένα εκτελέσιμο αρχείο sample1.exe, το οποίο αν εκτελεστεί<sup>8</sup> γράφει το μήνυμα 'Hello World!' στην κονσόλα των Windows.

```
Imports System
Public Module modSample
    Sub Main()
        Console.WriteLine("Hello World!")
    End Sub
End Module
VB7
```

```
using System;
public class modSample : object {
    static void Main() {
        Console.Write("Hello World!");
    }
}
C#
```

### Παράδειγμα 1 – Απλό παράδειγμα μιας κλάσης

Όπως επεξηγήθηκε και νωρίτερα, στην πρώτη γραμμή του κώδικα γίνεται δήλωση ότι η κλάση θα χρησιμοποιήσει την System Assembly (με την δήλωση Imports ή using για VB7 και C# αντίστοιχα). Στη συνέχεια γίνεται η δήλωση της έναρξης της κλάσης (του module για το συγκεκριμένο παράδειγμα σε VB) και τέλος η δήλωση της μεθόδου Main.

Είναι ευκαιρία σε αυτό το σημείο να επεξηγηθούν κάποιες βασικές αρχές που χρησιμοποιούνται στον παραπάνω κώδικα και ίσως να περνούν απαρατήρητες:

- Αφαιρώντας την δήλωση Imports System και προσπαθώντας να μεταγλωττίσουμε και πάλι το αρχείο, ο compiler θα μας επιστρέψει το μήνυμα «Name 'Console' is not declared». Αν παρόλα αυτά στη

---

<sup>8</sup> Εκτελέστε το αρχείο απευθείας από το παράθυρο του command line. Κάνοντας διπλό κλικ στον windows explorer επίσης θα εκτελεστεί, αλλά η κονσόλα που θα εμφανιστεί το μήνυμα θα κλείσει αμέσως.

συνέχεια γράψουμε την εντολή `Console.WriteLine` με την εξής σύνταξη: `System.Console.WriteLine`, δεν θα έχουμε κανένα πρόβλημα κατά τη μεταγλώττιση. Αυτό γίνεται γιατί η κλάση `Console` είναι μέρος της `assembly System`.

Δηλώνοντας όμως στην αρχή του `module` ότι θα χρησιμοποιούμε την `assembly System` (μέσω της δήλωσης `Imports`), δεν χρειάζεται να επαναλαμβάνουμε το πρόθεμα `'System.'` Για τη χρήση των περιεχομένων της αντίστοιχης `assembly`. Ουσιαστικά με μια δήλωση `Imports` εισάγουμε στον κώδικά μας ένα σύνολο από ονόματα (`namespace`) που μπορούν άμεσα να χρησιμοποιηθούν. Αυτή η λειτουργικότητα αποκτά μεγαλύτερη σημασία όταν χρησιμοποιούνται `assemblies` και κλάσεις με μεγαλύτερα ονόματα.

- Στην `Visual Basic 6` (και στις προγενέστερες εκδόσεις) δεν χρειαζόταν να δηλώσουμε την αρχή και το τέλος ενός `module` (`Public Module – End Module`), μια που το κάθε αρχείο (με επέκταση `.bas` σε αυτές τις εκδόσεις) ήταν υποχρεωτικά ένα και μόνο `module`. Στην `VB7`, λοιπόν, έχουμε τη δυνατότητα να ενσωματώνουμε σε ένα φυσικό αρχείο περισσότερα από ένα `modules` ή κλάσεις, δηλώνοντας την αρχή και το τέλος τους με τις λέξεις `Public/Private/Global` και `End` αντίστοιχα.
- Η κλήση της `κονσόλας` για την εγγραφή του μηνύματος είναι μια κλήση σε μέθοδο, όχι σε συνάρτηση, δηλαδή δεν επιστρέφει κάποια τιμή-αποτέλεσμα. Μια τέτοια κλήση, στην περίπτωση της `VB6` δεν θα απαιτούσε τη χρήση παρενθέσεων (μάλιστα ο `compiler` της `VB6` στις περισσότερες περιπτώσεις θεωρούσε κάτι τέτοιο συντακτικό σφάλμα). Στην `VB7` είναι υποχρεωτική η χρήση παρενθέσεων σε όλες τις κλήσεις μεθόδων ή συναρτήσεων.

Θα μπορούσαν να γίνουν άλλες παρόμοιες παρατηρήσεις κατά τη διάρκεια αυτής της εργασίας, παρόλα αυτά κάτι τέτοιο θα ήταν πέραν του σκοπού της. Για το σκοπό αυτό κρίνεται αναγκαία η προσοχή και παρατηρητικότητα του αναγνώστη στο κώδικα που είναι γραμμένος τόσο σε `VB7` όσο και σε `C#`, για

να γίνουν γρήγορα αντιληπτές διαφορές στη σύνταξη σε σχέση με την VB6 και την κλασική C αντίστοιχα<sup>9</sup>.

### 3.2 Μια ματιά στο παρασκήνιο

Συνήθως θεωρείται περιττό να ασχοληθεί ο προγραμματιστής με το τι γίνεται στο παρασκήνιο της εκτέλεσης του κώδικά του. Παρόλα αυτά, τα λίγα λεπτά που απαιτούνται για την ανάγνωση αυτής της παραγράφου θα διαφωτίσουν αρκετά και θα βοηθήσουν τόσο στην κατανόηση της λειτουργίας του .NET Framework, όσο και στην αποσαφήνιση μερικών εννοιών που σίγουρα θα συναντήσει κανείς κατά την ενασχόλησή του με αυτό.

Το εκτελέσιμο (.exe) αρχείο, που δημιουργήθηκε στο προηγούμενο παράδειγμα, χαρακτηρίζεται με τον όρο .NET PE (portable executable – μεταφερόμενο εκτελέσιμο), σε αντιδιαστολή με τα εκτελέσιμα αρχεία προηγούμενων εκδόσεων που χαρακτηρίζονται με τον όρο Windows PE. Η βασικότερη διαφορά μεταξύ των δυο, είναι ότι τα πρώτα εκτελούνται από το Common Language Runtime του .NET Framework, ενώ τα δεύτερα εκτελούνται άμεσα από το λειτουργικό σύστημα των Windows, με την υποβοήθηση του Runtime Environment της γλώσσας με την οποία δημιουργήθηκαν. Ο λόγος για τον οποίο γίνεται κάτι τέτοιο, είναι το γεγονός ότι τα .NET PE δεν περιέχουν τυπικό κώδικα μηχανής (machine code), αλλά περιέχουν κώδικα σε μια ενδιάμεση γλώσσα που ονομάζεται IL (intermediate language). Ουσιαστικά λοιπόν, ο compiler της VB7, της C# ή οποιασδήποτε άλλης γλώσσας υποστηρίζεται από το CLR (π.χ. J#), μετατρέπει τον κώδικα, που γράφεται από τον προγραμματιστή, σε κώδικα IL. Σε αντίθεση, δηλαδή, με παλαιότερους compilers, που μετέτρεπαν τον κώδικα της κάθε γλώσσας προγραμματισμού απευθείας σε τυπικό κώδικα μηχανής και επομένως σε εκτελέσιμα αρχεία άμεσα εκτελούμενα από το λειτουργικό σύστημα. Αν και η IL μοιάζει αρκετά στην assembly (στην γλώσσα προγραμματισμού και όχι στον τύπο dll που αναφέρθηκε παραπάνω), κυρίως ως προς το ότι περιέχει εντολές διαχείρισης τιμών σε

---

<sup>9</sup> Στο CD-ROM που συνοδεύει αυτή την εργασία περιλαμβάνεται σχετικό έγγραφο που παρουσιάζει με λεπτομέρεια τις συντακτικές διαφορές μεταξύ VB6 και VB7.

registers, διαφέρει ωστόσο στο ότι ο κώδικάς της δεν είναι γραμμένος για χρήση από κάποιο συγκεκριμένο επεξεργαστή.

Η τελική μεταγλώττιση του κώδικα IL, που περιέχεται πλέον στο εκτελέσιμο αρχείο του .NET, γίνεται κατά την εκτέλεση του προγράμματος από τον τελικό χρήστη. Την εργασία αυτή αναλαμβάνει ο J.I.T. compiler (Just-In-Time Compiler). Ο λόγος για τον οποίο υπεισέρχεται αυτό το επιπλέον επίπεδο μεταγλώττισης, είναι η δημιουργία εκτελέσιμων αρχείων ανεξάρτητων από την αρχιτεκτονική του συστήματος στο οποίο θα εκτελεστούν. Για παράδειγμα τα εκτελέσιμα αρχεία προηγούμενων εκδόσεων π.χ. της VB, έπρεπε να μεταγλωττιστούν με διαφορετικό compiler για χρήση του αποτελέσματος σε ένα σύστημα αρχιτεκτονικής Intel x86 από ότι σε ένα σύστημα Alpha. Επειδή όμως ο J.I.T. του .NET Framework προσφέρεται ανάλογα με την αρχιτεκτονική του συστήματος που θα χρησιμοποιηθεί, μπορεί κάθε φορά να μετατρέπει το κοινό εκτελέσιμο αρχείο σε κατάλληλο κώδικα μηχανής.

### **3.3 Προγραμματίζοντας για διάφορα περιβάλλοντα**

Ο αυξημένος αριθμός συσκευών που κυκλοφορούν στην αγορά με τα ολοένα εξελισσόμενα χαρακτηριστικά, σε συνδυασμό με την αύξηση της επεξεργαστικής τους ισχύος, δημιουργεί απαιτήσεις προγραμματισμού για την καλύτερη εκμετάλλευσή τους και την ικανοποίηση των αναγκών που προκύπτουν από αυτές. Συσκευές όπως προσωπικοί υπολογιστές, διακομιστές, υπολογιστές παλάμης (palmtop), tablet pc's κλπ. Απαιτούν παραμετροποίηση και προσαρμογή στις ανάγκες του εκάστοτε χρήστη ή επιχείρησης.

Οι διαφορές μεταξύ αυτών των συσκευών είναι αρκετές:

- Τάξη της επεξεργαστικής ισχύος (μικρή στα κινητά τηλέφωνα, μέση στα palmtops, μεγάλη στους PC)
- Μέσα I/O. Διαφόρων μεγεθών και δυνατοτήτων οθόνες και πληκτρολόγια, διάφορες θύρες επικοινωνίας (υπέρυθρες, USB, παράλληλη, Ethernet κλπ.) και μέσα αποθήκευσης (HD, FDD, Flash Memory, CD/DVD κλπ.)
- Ταχύτητες σύνδεσης με το Internet

- Προσανατολισμός και χρήση (γραφείο, διασκέδαση – multimedia, εν κινήσει κλπ.)

Ο τρόπος με τον οποίο προσπαθεί να αντιμετωπίσει αυτή την πληθώρα συσκευών η πλατφόρμα .NET, είναι με την δημιουργία εκδόσεων του .NET Framework και κοινή χρήση του CLR και των Class Libraries ανάλογα με την κάθε περίπτωση.

Το μεγαλύτερο μέρος λοιπόν του CLR είναι κοινό σε όλες τις συσκευές που υποστηρίζεται το .NET και επομένως είναι κοινά και τα χαρακτηριστικά των γλωσσών προγραμματισμού. Υπάρχει για παράδειγμα η δυνατότητα χρήσης όλων των γλωσσών προγραμματισμού, όπως Visual Basic, C#, J# κλπ. Στην δημιουργία ενός προγράμματος που θα λειτουργεί σε έναν υπολογιστή παλάμης. Άλλωστε η μεταγλώττιση του προγράμματος σε εκτελέσιμο αρχείο θα δημιουργήσει το ίδιο αρχείο σε γλώσσα IL όπως αναφέρεται και παραπάνω. Αυτό έχει επίσης ως αποτέλεσμα τη χρήση προγραμμάτων που δημιουργήθηκαν για άλλο περιβάλλον εκτέλεσης, απευθείας σε κάποια συσκευή. Για παράδειγμα, μια assembly (αρχείο .dll) που περιλαμβάνει ένα μαθηματικό αλγόριθμο, μέρος μιας ολοκληρωμένης εφαρμογής, μπορεί απευθείας να χρησιμοποιηθεί και σε ένα palmtop, χωρίς να είναι απαραίτητη κάποια τροποποίηση ή ακόμα και μεταγλώττιση.

## 4. Visual Basic .NET

Η Visual Basic .NET ή Visual Basic 7 είναι μια εξέλιξη της Visual Basic για χρήση στο .NET Framework. Στα γνωστά πλεονεκτήματα της γλώσσας με βασικότερο την ταχύτητα στην ανάπτυξη εφαρμογών σε περιβάλλον Windows, προστέθηκαν χαρακτηριστικά που μέχρι σήμερα ίσως απαιτούσαν τη χρήση της C++. Τέτοια χαρακτηριστικά, τα οποία θα μελετηθούν σε αυτό το κεφάλαιο, είναι η κληρονομικότητα (inheritance), η ελεύθερη διαχείριση των νημάτων των εφαρμογών (free threading), το garbage collection για την καλύτερη διαχείριση της μνήμης, ο δομημένος έλεγχος των σφαλμάτων (structured exception handling) κ.α.

Είναι σημαντικό να αναφέρουμε ότι για πρώτη φορά μετά από αρκετές εκδόσεις της Visual Basic, ο κώδικας που γράφτηκε σε προγενέστερες εκδόσεις δεν μπορεί να χρησιμοποιηθεί απευθείας στην Visual Basic .NET. Θα είναι γνωστό στους προγραμματιστές της γλώσσας, ότι projects που γράφτηκαν σε VB5, μπορούν χωρίς καμία τροποποίηση να υποστούν επεξεργασία και μεταγλώττιση από την έκδοση 6. Αυτό δεν ισχύει από την έκδοση 6 στην έκδοση 7, λόγω των σημαντικών αλλαγών που παρουσιάζονται στη σύνταξη της γλώσσας. Το Visual Studio .NET περιέχει έναν οδηγό που κάνει αρκετές από τις απαιτούμενες αλλαγές σε παλαιότερο κώδικα για την χρήση στην έκδοση 7. Παρόλα αυτά, τις περισσότερες φορές απαιτείται η επέμβαση του χρήστη για την ολοκλήρωση των αλλαγών. Σε κάθε περίπτωση, γίνεται γρήγορα αντιληπτό, ότι λόγω της πληθώρας των νέων βελτιωμένων χαρακτηριστικών της γλώσσας, τις περισσότερες φορές κρίνεται αποδοτικότερη η επανασύνταξη του κώδικα, μια που πολλές διαδικασίες μπορούν να πραγματοποιηθούν με σημαντική μείωση του κώδικα, κυρίως κάνοντας χρήση των Class Libraries του .NET Framework.

### 4.1 Inheritance

Ένα νέο χαρακτηριστικό που προστέθηκε στην VB7, ίσως το πιο εντυπωσιακό και πολύ-αναμενόμενο είναι το Inheritance. Πρόκειται για μια τυπική τεχνική προγραμματισμού για την δημιουργία νέων αντικειμένων, στηριζόμενα σε άλλα υπάρχοντα. Ένα κοινό παράδειγμα είναι η περίπτωση ενός αντικειμένου το οποίο μπορεί να περιγράψει την γενική οντότητα 'ζώο' και το οποίο, αν κληρονομηθεί από μια νέα κλάση και επεκταθεί, θα μπορεί

να περιγράψει συγκεκριμένα ζώα, όπως ο άνθρωπος. Αυτή η απλή περίπτωση φαίνεται στο Παράδειγμα 2.

```
Public Class Animal
    `ιδιότητες όλων των ζώων
    Public Age As Integer
    Public Sex As String
    Public Race As String

    `ενέργεια που μπορούν να κάνουν τα περισσότερα ζώα
    Public Sub Walk ()
        `εδώ μπορεί να γραφτεί κώδικας που εξομοιώνει το περπάτημα!
    End Sub
End Class

Public Class Human
    `ο άνθρωπος κληρονομεί τις ιδιότητες του ζώου
    Inherits Animal

    `επιπλέον μεταβλητές που περιγράφουν τον άνθρωπο
    Public IQ As Integer
    Public Height As Single

    `μια ενέργεια που μπορεί να κάνει μόνο ο άνθρωπος
    Public Sub Speak ()
        `εδώ μπορεί να γραφτεί κώδικας που εξομοιώνει την ομιλία!
    End Sub
End Class

Public Class Dog
    `ο σκύλος κληρονομεί τις ιδιότητες του ζώου
    Inherits Animal

    `μια ενέργεια που μπορεί να κάνει μόνο ο σκύλος
    Public Sub Bark ()
        `εδώ μπορεί να γραφτεί κώδικας που εξομοιώνει το γαύγισμα!
    End Sub
End Class
```

### Παράδειγμα 2 - Inheritance στην Visual Basic .NET

Η τεχνική αυτή βοηθά στην οργάνωση του κώδικα σε αντικείμενα (object oriented κώδικας) κυρίως στο γεγονός ότι μειώνει σημαντικά τον κώδικα που γράφεται βοηθώντας τόσο στην ανάπτυξη όσο και στην συντήρησή του.

## 4.2 Free Threading

Ο τυπικός τρόπος με τον οποίο εκτελείται ο κώδικας ενός προγράμματος είναι γραμμή προς γραμμή, από την πρώτη μέχρι την τελευταία, σε μια διαδικασία η οποία είναι γνωστή ως thread (νήμα). Αν κατά τη διάρκεια

εκτέλεσης του κώδικα γίνεται κλήση μιας συνάρτησης, τότε η διαδικασία εκτέλεσης σταματά προσωρινά, μέχρι να ολοκληρωθεί η εκτέλεση αυτής της συνάρτησης, οπότε και συνεχίζει και πάλι από το σημείο που είχε σταματήσει. Υπάρχουν όμως διάφορες διεργασίες σε ένα πρόγραμμα, οι οποίες θα θέλαμε να μην διακόπτονται, όταν πρέπει να εκτελεστούν κάποιες άλλες. Για παράδειγμα, η διαδικασία αποθήκευσης ή εκτύπωσης ενός εγγράφου από το Microsoft Word επιτρέπει στον χρήστη, αν αυτή η διαδικασία είναι χρονοβόρα, να συνεχίσει να γράφει το κείμενό του.

```
Imports System
Imports System.Threading

Module ThreadingExample
    Dim thread_A As New Thread(AddressOf Procedure_A)

    Sub Main()

        thread_A.Start()

        Console.WriteLine(«Το βασικό thread έχει ολοκληρωθεί.»)
        Console.WriteLine(«Πριν ολοκληρωθεί ξεκίνησε ένα νέο...»)

    End Sub

    Sub Procedure_A()

        Console.WriteLine(«Η διαδικασία A ξεκίνησε...»)

        Dim i As Integer
        For i = 1 To 15
            thread_A.Sleep(1000)
            Console.WriteLine("Διαδικασία A: " & (i*100/15) & "%")
        Next

        Console.WriteLine(«Η διαδικασία A ολοκληρώθηκε...»)

    End Sub

End Module
```

### Παράδειγμα 3 – Πολυνηματική εκτέλεση κώδικα με VB.NET

Για να επιτευχθεί αυτό, θα πρέπει το πρόγραμμα να δημιουργεί ένα νέο thread, δηλαδή μια νέα ακολουθία εκτέλεσης κώδικα, το οποίο θα εκτελείται παράλληλα με αυτό το οποίο το δημιούργησε. Στην περίπτωση του Microsoft Word, το thread που αναλαμβάνει να δέχεται την πληκτρολόγηση του χρήστη, δημιουργεί ένα νέο thread εκτέλεσης του κώδικα της

εκτύπωσης, το οποίο λειτουργεί ανεξάρτητα, επιτρέποντας στο χρήστη να συνεχίσει να γράφει.

Αυτή η τεχνική δεν ήταν δυνατή στην Visual Basic, μέχρι την έκδοση .NET. Η υλοποίηση του ελεύθερου threading στην έκδοση .NET γίνεται μέσω του System.Threading namespace του .NET Framework, το οποίο περιλαμβάνει κλάσεις για την διαχείριση των threads μιας εφαρμογής. Μια απλή υλοποίηση φαίνεται στο Παράδειγμα 3. Σε αυτό, η εκτέλεση του κώδικα ξεκινά από την γραμμή Sub Main. Πριν τη γραμμή που γράφει στην κονσόλα ότι η εκτέλεση ολοκληρώνεται, γίνεται η εκκίνηση της εκτέλεσης του κώδικα της Procedure\_A, μέσω του δεύτερου thread.

### 4.3 Garbage Collection

Η Visual Basic 6 και οι προκάτοχοί της χρησιμοποίησαν την μέθοδο του reference counting (καταμέτρηση αναφορών) για τη διαχείριση της μνήμης που καταλαμβάνει ένα πρόγραμμα κατά την εκτέλεσή του. Σύμφωνα με αυτή την μέθοδο, όταν κατά την εκτέλεση κώδικα δημιουργείται ένα αντικείμενο, το περιβάλλον εκτέλεσης του προγράμματος τηρεί σε ένα μετρητή, τον αριθμό των σημείων στο συνολικό πρόγραμμα όπου χρησιμοποιείται αυτό το αντικείμενο. Όταν ο μετρητής αυτός μηδενιστεί, τότε προφανώς το αντικείμενο δεν είναι απαραίτητο και η μνήμη που καταλαμβάνει απελευθερώνεται. Το μειονέκτημα αυτής της μεθόδου είναι ότι στηρίζεται στη σωστή διαχείριση των αντικειμένων από τον προγραμματιστή, ο οποίος πρέπει να εξασφαλίζει τη σωστή και έγκαιρη καταστροφή τους, για να ελευθερωθεί τελικά η μνήμη που αυτά καταλαμβάνουν.

Στο Common Language Runtime του .NET Framework, οπότε και στη Visual Basic .NET καθώς και στις υπόλοιπες γλώσσες προγραμματισμού που τρέχουν κάτω από αυτό, η διαχείριση της μνήμης γίνεται με την μέθοδο Garbage Collection. Κατά αυτή τη μέθοδο, η απελευθέρωση της μνήμης που καταλαμβάνει ένα αντικείμενο δεν γίνεται αμέσως μόλις αυτό απελευθερωθεί από όλα τα σημεία του κώδικα. Αντίθετα, το αντικείμενο παραμένει στη μνήμη που καταλαμβάνει το συνολικό πρόγραμμα, μαζί με άλλα αντικείμενα που μπορεί πλέον να μην είναι απαραίτητα, ενώ νέα αντικείμενα που δημιουργούνται καταλαμβάνουν αδέσμευτα σημεία της μνήμης. Επειδή βέβαια η διαθέσιμη μνήμη είναι πάντα σε ανεπάρκεια, ο Garbage Collector εντοπίζει κάποια κατάλληλη χρονική στιγμή για να μαζέψει όλα τα

παραπάνω σκουπίδια και να τα πετάξει από τη μνήμη, χωρίς αυτό να επηρεάσει την απρόσκοπτη λειτουργία του προγράμματος. Το χρονικό σημείο αυτό, μπορεί να επιλέγει και από την προγραμματιστή, επεμβαίνοντας στην όλη διαδικασία.

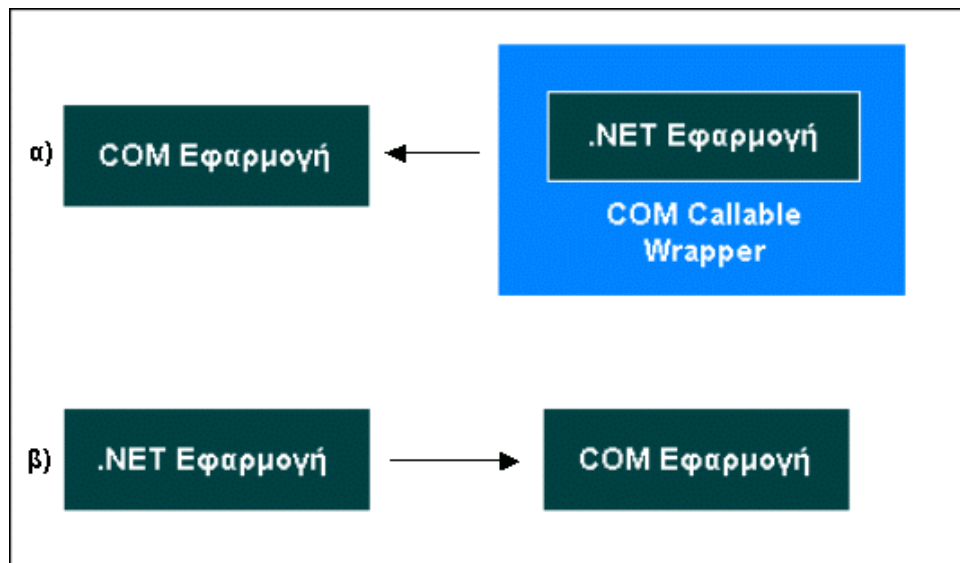
Το μειονέκτημα αυτής της μεθόδου είναι ότι η απελευθέρωση της μνήμης δεν συμβαίνει άμεσα μόλις καταστραφεί ένα αντικείμενο. Σε γενικές γραμμές, υπάρχουν υποστηρικτές και των δυο μεθόδων και μάλλον θα έπρεπε να περιμένουμε λίγο περισσότερο τα αποτελέσματα από την εφαρμογή και χρήση του Garbage Collection στο .NET Framework, πριν καταλήξουμε σε περαιτέρω συμπεράσματα.

#### **4.4 Παράλληλη χρήση με την Visual Basic 6 (Interoperability)**

Τόσο πριν, όσο και μετά την διάθεση του .NET Framework και του Visual Studio .NET 2002 από την Microsoft, κυριάρχησε στο δίκτυο μια πολύ μεγάλη συζήτηση σχετικά με το ποια είναι η κατάλληλη στιγμή στην οποία θα κανείς να κάνει το βήμα της μετάβασης σε αυτά. Σε κάθε περίπτωση, θα πρέπει να θεωρείται αυτονόητο, ότι αυτή η μετάβαση δεν είναι δυνατόν να αφήσει πίσω της αμέτρητες γραμμές κώδικα και ActiveX DLL (βιβλιοθήκες συναρτήσεων και αντικειμένων) που γράφηκαν σε προγενέστερα συστήματα, ούτε να αφήσει πίσω της την εμπειρία και την ταχύτητα δημιουργίας ενός προγράμματος ανάλογα με την εμπειρία του κάθε προγραμματιστή. Τα νέα, όμως, χαρακτηριστικά του Microsoft .NET δεν αφήνουν πολλά περιθώρια για καθυστερήσεις σε αυτή την μετάβαση. Σε αυτό ωστόσο, μπορεί να βοηθήσει σημαντικά το γεγονός της διαλειτουργικότητας και της δυνατότητας παράλληλης χρήσης του .NET με προγενέστερα συστήματα ανάπτυξης λογισμικού όπως η VB6, η C++. Διαλειτουργικότητα η οποία είναι δυνατή και προς τις δυο κατευθύνσεις, δηλαδή μια .NET εφαρμογή μπορεί να κάνει χρήση ενός DLL που δημιουργήθηκε από την VB6 αλλά και το αντίστροφο, όπως φαίνεται και στο Σχήμα 2.

Η κλήση COM εφαρμογής από μια .NET εφαρμογή, είναι διαλειτουργικότητα μάλλον θα πρέπει να θεωρηθεί αυτονόητη, μια που πολύ απλά το .NET Framework μπορεί να δημιουργήσει αντικείμενα χρησιμοποιώντας COM

κλάσεις, δηλαδή κλάσεις που γράφτηκαν με VB6 ή C++ (καθώς και άλλα συστήματα ανάπτυξης που υποστηρίζουν αυτή την τεχνολογία, όπως η Borland Delphi).

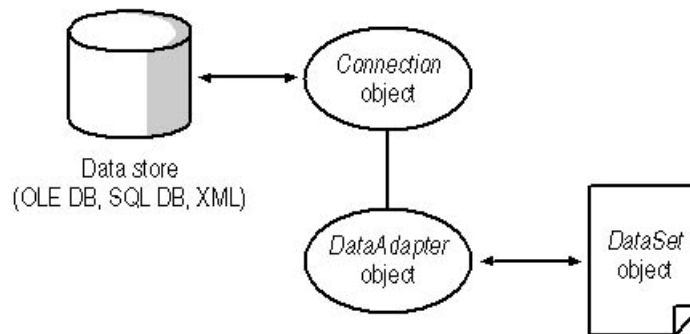


**Σχήμα 2 – Χρήση (α) .NET εφαρμογής από COM και (β) COM εφαρμογής από .NET**

Αξίζει όμως να σχολιάσουμε την αντίστροφη επικοινωνία, δηλαδή τον τρόπο με τον οποίο μια εφαρμογή που γράφεται με VB6 ή C++, δημιουργεί και χρησιμοποιεί αντικείμενα βασισμένα σε .NET κλάσεις, μια που το .NET Framework δεν προσφέρει τη δυνατότητα δημιουργίας ActiveX DLL που θα μπορούσαν να χρησιμοποιηθούν άμεσα. Για την επίτευξη αυτής της διαλειτουργικότητας, οι compilers των γλωσσών προγραμματισμού του .NET Framework (π.χ. VB7, C#, J# κλπ.) έχουν τη δυνατότητα να δημιουργούν μαζί με το εκτελέσιμο αρχείο (exe, dll) και ένα δεύτερο αρχείο με επέκταση tlb (type library) το οποίο συμπεριφέρεται σαν ActiveX DLL και επομένως μπορεί να χρησιμοποιηθεί από αντίστοιχες εφαρμογές. Η ύπαρξη και των δυο αρχείων για την χρήση τους είναι απαραίτητη, μια που το δεύτερο αποτελεί μόνο έναν αντιπρόσωπο (proxy) του .NET εκτελέσιμου αρχείου στον κόσμο του COM. Κάθε κλήση προς το αρχείο tlb μεταφέρεται στο αντίστοιχο .NET εκτελέσιμο, το αποτέλεσμα επιστρέφεται πίσω στο πρώτο και από εκεί την εφαρμογή από την οποία ξεκίνησε η κλήση (βλ. Σχήμα 2). Το αρχείο tlb ονομάζεται και COM Callable Wrapper (CCW).

## 5. Διαχείριση Δεδομένων – ADO.NET

Το ADO.NET (ADO: Active Data Objects) αποτελεί την εξέλιξη του Microsoft ADO στο .NET Framework. Αποτελείται από μια σειρά από κλάσεις, που περιλαμβάνονται στις βιβλιοθήκες του .NET Framework και σκοπό έχουν να αποτελέσουν τον συνδετικό κρίκο μεταξύ των .NET εφαρμογών και διάφορων πηγών δεδομένων. Αυτές οι πηγές δεδομένων μπορεί να είναι αρχεία XML και βάσεις δεδομένων<sup>10</sup>.



Σχήμα 3 – Τα τρία επίπεδα του ADO.NET<sup>11</sup>

Ο τρόπος λειτουργίας του ADO.NET μπορεί να διαχωριστεί στα παρακάτω τρία επίπεδα, τα οποία διακρίνονται και στο Σχήμα 3:

- **Η φυσική πηγή δεδομένων:** Μπορεί να είναι αρχεία XML, βάσεις δεδομένων OLEDB (δηλαδή βάσεις δεδομένων με υποστήριξη OLEDB πρόσβασης όπως SQL Server, Oracle, Sybase κλπ.) ή βάσεις δεδομένων SQL Server.
- **Οι κλάσεις παροχής δεδομένων (data providers):** Αποτελούν το σημείο σύνδεσης με τη φυσική πηγή δεδομένων. Μπορούν να μεταφέρουν δεδομένα από την πηγή στην μνήμη (data-read) και το αντίστροφο (data-write). Οι τρεις βασικές κλάσεις/ αντικείμενα που

---

<sup>10</sup> άλλες πηγές δεδομένων όπως αρχεία κειμένου ή ο Microsoft Exchange Server μπορούν να χρησιμοποιηθούν, στη παράγραφο όμως αυτή θα εξεταστούν τα αρχεία XML και οι βάσεις δεδομένων που αποτελούν και τις πιο συνήθεις περιπτώσεις.

<sup>11</sup> Microsoft Press – Developing Web Applications with Microsoft Visual Basic .NET and Microsoft Visual C# .NET

επιτελούν αυτό το ρόλο είναι το *Connection*, το *Command* και οι *Data Adapters*.

- **Τα σετ δεδομένων (data set):** Αποτελούν τις αναπαραστάσεις στην μνήμη πινάκων δεδομένων και συσχετίσεων μεταξύ τους. Η δημιουργία τους γίνεται μέσω των αντικειμένων παροχής δεδομένων.

Αν ο αναγνώστης έχει κάποια εμπειρία από τον προκάτοχο των ADO.NET, τα ADO, αντιλαμβάνεται αμέσως την φιλοσοφία διάσπασης των δεδομένων από την πηγή τους, κατά τη διαχείρισή τους από τις .NET εφαρμογές. Αν και η διαχείριση δεδομένων απομακρυσμένα από την πηγή τους ήταν δυνατή και από τα ADO, κάθε χρήση τους με αυτό τον τρόπο υπενθύμιζε συνεχώς τον πραγματικό προορισμό τους, δηλαδή τη συνεχή σύνδεση με την πηγή τους. Στο νέο σκηνικό που δημιουργείται με την ανάπτυξη του Internet και των Client/Server εφαρμογών, αναδεικνύεται ακόμα περισσότερο η ανάγκη για διαχείριση δεδομένων «μακριά από την πηγή τους». Αυτό βέβαια σημαίνει διαχείριση σε διαφορετικό υπολογιστή από τη βάση δεδομένων, χωρίς την απασχόληση του δικτύου που συνδέει τους δυο υπολογιστές, παρά μόνο κατά τη λήψη των δεδομένων και την επιστροφή των τροποποιήσεων για αποθήκευση.

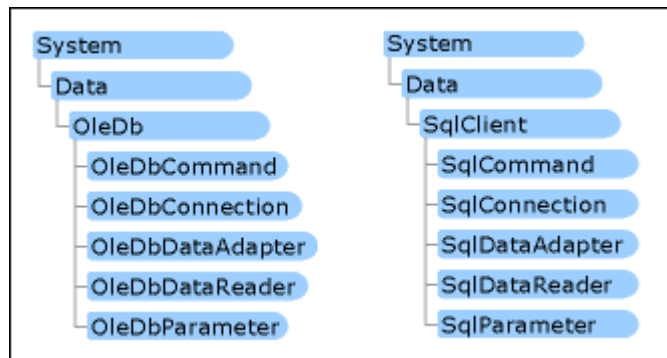
## 5.1 Φυσική Πηγή Δεδομένων

Το πρώτο επίπεδο, δηλαδή η φυσική πηγή δεδομένων, μάλλον δεν αποτελεί αντικείμενο του .NET Framework, οπότε δεν θα εξεταστεί. Αξίζει όμως να σημειωθεί ότι η Microsoft διανέμει τόσο μαζί με το Visual Studio .NET, όσο και με άλλα εργαλεία που προσφέρονται δωρεάν, μια δωρεάν έκδοση του SQL Server, το MSDE (Microsoft Desktop Engine). Η έκδοση αυτή είναι σχεδόν πανομοιότυπη έκδοση με τον SQL Server σε ότι αφορά τις δυνατότητές της. Τα τρία σημεία στα οποία υστερεί το MSDE και διαφοροποιούνται οι δυο εκδόσεις είναι η μη υποστήριξη Data Mining, ο περιορισμός των ταυτόχρονων ανοιχτών συνδέσεων σε 5, καθώς και η πλήρης έλλειψη γραφικού περιβάλλοντος διαχείρισης (Enterprise Manager). Παρόλα αυτά, αν το MSDE αποκτηθεί μέσω του Visual Studio .NET, δίνεται η άδεια για ελεύθερη διανομή του και εγκατάστασή του σε άλλους υπολογιστές, για την εκτέλεση εφαρμογών που χρησιμοποιούν βάσεις δεδομένων του SQL Server. Όπου δεν υπάρχει η ανάγκη για περισσότερες από ταυτόχρονες 5 συνδέσεις, το MSDE αποτελεί μια ιδανική και ανέξοδη

λύση, που μπορεί να υποκαταστήσει την χρήση της Microsoft Access που χρησιμοποιείται σε παρόμοιες περιπτώσεις. Όταν βέβαια αναφερόμαστε σε ταυτόχρονες συνδέσεις με τη χρήση του ADO.NET, μιλάμε μόνο για τη στιγμή λήψης και αποθήκευσης των δεδομένων, μια που στο ενδιάμεσο δεν απαιτείται σύνδεση με τη βάση δεδομένων.

## 5.2 Data Providers

Οι data providers αναλαμβάνουν να εκτελέσουν εντολές προς τις πηγές δεδομένων για την λήψη και ενημέρωση δεδομένων. Αυτές οι εντολές μπορεί να είναι ερωτήματα SQL ή εκτέλεση stored procedures<sup>12</sup>, εφόσον υποστηρίζονται από την πηγή δεδομένων. Στο Σχήμα 4 φαίνεται η δομή των κλάσεων των δυο βασικών Data Adapters του ADO.NET.



**Σχήμα 4 – Τα namespaces των δυο βασικών Data Adapters του ADO.NET<sup>13</sup>**

Όπως φαίνεται και στο σχήμα, τόσο ο OleDb, που χρησιμοποιείται για την πρόσβαση σε βάσεις μέσω OleDb Drivers, όσο και ο SqlClient, που χρησιμοποιείται για πρόσβαση αποκλειστικά σε βάσεις του Microsoft SQL Server, αποτελούνται από τα ίδια μέλη. Αν και ο OleDb μπορεί να χρησιμοποιηθεί για την πρόσβαση και στον SQL Server – άλλωστε αυτός ήταν ο τρόπος πρόσβασης για τις προγενέστερες εκδόσεις των ADO – σύμφωνα με την βιβλιογραφία της Microsoft, ο SqlClient θα πρέπει να

---

<sup>12</sup> Οι stored procedures είναι συναρτήσεις γραμμένες σε μορφή SQL ερωτημάτων και είναι αποθηκευμένες στις πηγές δεδομένων. Μπορεί να περιέχουν μια ή περισσότερες εντολές SQL οι οποίες εκτελούν διάφορες εργασίες πριν επιστρέψουν, όπως και ένα απλό SQL Query, ένα σετ δεδομένων ή μια επιστρεφόμενη τιμή/ αποτέλεσμα των εργασιών που εκτελέστηκαν.

<sup>13</sup> .NET Framework Documentation

θεωρείται αμεσότερος και επομένως γρηγορότερος τρόπος πρόσβασης στον SQL Server. Ανάλογος Data Adapter (Oracle .NET Data Provider) δημιουργήθηκε πρόσφατα και για την απευθείας πρόσβαση σε βάσεις δεδομένων Oracle, χωρίς την παρεμβολή της τεχνολογίας OLEDB που χρησιμοποιούνταν μέχρι σήμερα.

Η διαδικασία πρόσβασης και για τους τρεις Data Adapters αποτελείται από τα εξής βήματα:

- Δημιουργία ενός αντικειμένου DataAdapter (OleDbDataAdapter, SqlDataAdapter ή OracleDataAdapter) και καθορισμός του ερωτήματος sql που θα χρησιμοποιηθεί για την ανάγνωση των δεδομένων.
- Εκτέλεση της μεθόδου για ανάγνωση δεδομένων (π.χ. DataAdapter.Fill) και την μεταφορά στη μνήμη με τη μορφή ενός DataSet (βλ. Επόμενη παράγραφο). Η διαδικασία αυτή ενεργοποιεί εσωτερικά ένα αντικείμενο DataCommand για την υποβολή του ερωτήματος στην πηγή δεδομένων.
- Χρήση του DataSet για προβολή ή/ και τροποποίηση των δεδομένων που αναγνώστηκαν από την πηγή δεδομένων.
- Αν γίνει τροποποίηση των δεδομένων, εκτελείται μια εντολή ενημέρωσης στον DataAdapter η οποία ενεργοποιεί τρία αντικείμενα DataCommand τα οποία ενημερώνουν την πηγή δεδομένων με τις αλλαγές. Τα τρία αυτά αντικείμενα αντιστοιχούν σε εντολές δημιουργίας, διαγραφής και ενημέρωσης εγγραφών (INSERT, DELETE, UPDATE).

Στο Παράδειγμα 4 φαίνεται η παραπάνω διαδικασία γραμμένη σε κώδικα VB.NET. Πανομοιότυπη βέβαια είναι η διαδικασία χρησιμοποιώντας C#.

```
Private cDataSet As Data.DataSet

Private Sub Read_Change_Save
    'σύνδεση με την βάση δεδομένων Northwind
    Dim cConnection As New OleDb.OleDbConnection()
    cConnection.ConnectionString = _
        "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Northwind.mdb"
    cConnection.Open()

    'δημιουργία ενός command object και καθορισμός του
    'query που θα χρησιμοποιεί
```

```

Dim cSelectCommand As New OleDb.OleDbCommand()
cSelectCommand.CommandText = "SELECT * FROM Employees"
cSelectCommand.Connection = cConnection

'δημιουργία του data adapter και εκτέλεση του προηγούμενου
'command με ταυτόχρονη μεταφορά των αποτελεσμάτων σε ένα
'dataset
Dim cDataAdapter As New OleDb.OleDbDataAdapter()
cDataSet = New DataSet()
cDataAdapter.SelectCommand = cSelectCommand
cDataAdapter.Fill(cDataSet)

MessageBox.Show("Εγινε λήψη " & cDataSet.Tables(0).Rows.Count & "
εγγραφών!")

'///// εδώ μπορούν να γίνουν διάφορες τροποποιήσεις
'στα δεδομένα του dataset ////

'αυτόματη δημιουργία των εντολών INSERT/UPDATE/DELETE
'που θα ενημερώσουν τη βάση δεδομένων
Dim cCommandBuilder As OleDb.OleDbCommandBuilder = New
OleDb.OleDbCommandBuilder(cDataAdapter)
cDataAdapter.UpdateCommand = cCommandBuilder.GetUpdateCommand
cDataAdapter.InsertCommand = cCommandBuilder.GetInsertCommand
cDataAdapter.DeleteCommand = cCommandBuilder.GetDeleteCommand

'εκτέλεση της ενημέρωσης
cDataAdapter.Update(cDataSet)

End Sub

```

#### Παράδειγμα 4 – Παράδειγμα χρήσης του DataAdapter

### 5.3 Data Sets

Το αντικείμενο DataSet αναλαμβάνει την διαχείριση στην μνήμη των δεδομένων που λαμβάνονται μέσω ενός DataAdapter. Μπορεί να θεωρηθεί αντίστοιχο του ADO Recordset που υπήρχε στις προγενέστερες εκδόσεις των ADO, με τη διαφορά ότι δεν φέρει λειτουργικότητα για την λήψη και αποθήκευση δεδομένων όπως ένα Recordset, αφού αυτές οι λειτουργίες εκτελούνται από τους DataAdapters.

Σημαντικό να αναφερθεί είναι ότι στη δομή της βιβλιοθήκης Data (βλ. Σχήμα 4), το DataSet βρίσκεται στο ίδιο επίπεδο με τα OleDb και SqlClient, που σημαίνει ότι είναι κοινό αντικείμενο και για τα δυο. Και επειδή ένα DataSet μπορεί να δημιουργηθεί και από ένα έγγραφο XML, γίνεται αντιληπτό ότι η φιλοσοφία του είναι γενικά η διαχείριση δεδομένων στη μνήμη και όχι μόνο το τελικό επίπεδο διαχείρισης βάσεων δεδομένων.

Ίσως το σημαντικότερο σημείο του DataSet είναι η δυνατότητά του να φέρει δεδομένα από διαφορετικές πηγές και πίνακες μιας βάσης δεδομένων (DataSet.Tables). Εντυπωσιακότερο επίσης είναι το γεγονός ότι μπορεί να φέρει συσχετίσεις μεταξύ των πινάκων που περιέχει, με τη μορφή Foreign Key Constraints, τα οποία στην ορολογία του ADO.NET ονομάζονται DataRelations.

Άλλο ένα αντικείμενο το οποίο συμπληρώνει τη λειτουργικότητα του DataSet είναι το DataView. Η διαφορά του με το DataSet είναι ότι διαχειρίζεται δεδομένα μόνο για ανάγνωση και όχι για ενημέρωση. Μπορεί να φέρει μόνο ένα σύνολο δεδομένων, το οποίο όμως να προέρχεται με σύνδεση (JOIN) δεδομένων από πολλά tables ενός DataSet. Είναι λοιπόν δυνατή η δημιουργία δυναμικών JOIN σε επίπεδο μνήμης, χωρίς σύνδεση με την πηγή δεδομένων. Μπορούν επίσης να δημιουργηθούν στήλες που να φέρουν αποτελέσματα υπολογισμών άλλων στηλών, π.χ. το άθροισμα δυο εξ αυτών.

```
'αποθήκευση των περιεχομένων του DataSet στο δίσκο
'σε μορφή XML
cDataSet.WriteXml («c:\temp\data.xml»)

'αποθήκευση των metadata του DataSet στο δίσκο
'σε μορφή XML Schema
cDataSet.WriteXmlSchema („c:\temp\schema.xsd“)

'Ανάγνωση ενός XML εγγράφου από το δίσκο
Dim cDataSet2 AS New DataSet
cDataSet2.ReadXml (“c:\temp\data.xml“)
```

#### **Παράδειγμα 5 – Αποθήκευση δεδομένων και metadata ενός DataSet σε μορφή XML**

Τέλος, μεταξύ άλλων λειτουργιών, ένα DataSet μπορεί να επιστρέψει τα περιεχόμενά του (όλους του πίνακες που περιλαμβάνει) σε μορφή εγγράφου XML (βλ. Παράδειγμα 5). Το έγγραφο αυτό μπορεί να μεταφερθεί στο δίκτυο ή να αποθηκευτεί σε κάποια μονάδα αποθήκευσης, να ανακτηθεί αργότερα και να επανασυνδεθεί με την πηγή δεδομένων. Επίσης, είναι δυνατή η αποθήκευση του σχήματος του DataSet σε ένα έγγραφο XML Schema (περισσότερα για το XML Schema βλ. στο Κεφάλαιο 7) το οποία θα περιγράφει τη μορφή των δεδομένων (metadata) καθώς και τις συσχετίσεις μεταξύ των πινάκων που περιέχονται σε αυτό.

Τα έγγραφα XML που παράγονται από ένα DataSet δεν φέρουν κάποια ιδιαίτερη δομή, αποδεκτή μόνο από το .NET Framework ή την Microsoft γενικότερα. Ο συνδυασμός ενός XML εγγράφου που φέρει δεδομένα και ενός XML Schema που τα περιγράφει, δίνει τη δυνατότητα ανάγνωσής τους σε οποιοδήποτε σύστημα σε όποια πλατφόρμα και αν λειτουργεί (π.χ. Windows/ Linux PC, Sun Solaris κλπ., Visual Basic 6 και ADO, C++ κλπ.).

## 6. Web Services

Τα web services παρουσιάζονται ως κορμός του .NET Framework. Πρόκειται για μια τεχνολογία που επιτρέπει την επικοινωνία μεταξύ εφαρμογών οι οποίες μπορεί να λειτουργούν κάτω από οποιαδήποτε πλατφόρμα λογισμικού και οποιαδήποτε συσκευή (hardware), χρησιμοποιώντας πρότυπα του Internet που τυγχάνουν ευρείας αποδοχής. Η εξέλιξη των Web Services παρακολουθείται στενά από το W3C<sup>14</sup> (βλ. <http://www.w3.org/2002/ws>)

### 6.1 Υπάρχουσες τεχνολογίες

Όσο οι τεχνολογίες και οι εφαρμογές της πληροφορικής άρχισαν να γίνονται περισσότερο απαραίτητες στην καθημερινή μας ζωή, δημιουργήθηκε η ανάγκη να γίνουν και πολυπλοκότερες, σταθερότερες και περισσότερο καταναμημένες. Για παράδειγμα, η εργασία που επιτελούσε σε μια εταιρεία ένας υπολογιστής, διαχωρίζεται σε δυο ή περισσότερα μέρη (π.χ. PC-διακομιστής, PC-διακομιστής-διακομιστής βάσης δεδομένων-backup server, PC-web server κλπ.), με σκοπό την αποδοτικότερη συνεργασία μεταξύ των υπαλλήλων, την αύξηση της απόδοσης του συστήματος και την βελτίωση της αξιοπιστίας του. Το γεγονός αυτό, της διάσπασης των εφαρμογών και της επικοινωνίας τους μέσω δικτύου, δημιούργησε διάφορες τεχνολογίες διασύνδεσής τους.

Πρόκειται για τεχνολογίες οι οποίες συνήθως δεν είναι ορατές στον τελικό χρήστη, παρά μόνο στους αρχιτέκτονες, προγραμματιστές και ενδεχομένως και διαχειριστές (administrators) ενός συστήματος και παρέχονται από πακέτα ανάπτυξης λογισμικού ή λειτουργικά συστήματα. Μια τέτοια

---

<sup>14</sup> Πρόκειται για μια κοινοπραξία που ξεκίνησε το 1994 στο M.I.T. και παρέχει μια πλατφόρμα συζήτησης για διάφορα θέματα που αφορούν το Internet. Κατηγοριοποιημένα ανά θέματα, το W3C δίνει αφορμή για συζητήσεις και επεξεργάζεται έγγραφα, τα οποία καταλήγουν στα γνωστά W3C Recommendations, που προτείνουν τρόπους υλοποίησης και χρήσης τεχνολογιών του Internet, όπως Web Services, XML Schema, HTML και πολλά άλλα. Τυγχάνει της αναγνώρισης τουλάχιστον 500 γνωστών εταιριών και οργανισμών, γεγονός που καθιστά αυτά τα Recommendation Documents πάρα πολύ σημαντικά και πρακτικά απαραίτητα.

τεχνολογία είναι το DCOM (Distributed Component Object Model) της Microsoft, το οποίο επιτρέπει την επικοινωνία μεταξύ COM εφαρμογών<sup>15</sup> που βρίσκονται σε διαφορετικούς υπολογιστές και χρησιμοποιούν το λειτουργικό σύστημα των Windows<sup>16</sup>. Αντίστοιχη τεχνολογία είναι το Java RMI και η CORBA, τα οποία έχουν μερικά κοινά χαρακτηριστικά με το DCOM, όπως η διατήρηση σταθερής σύνδεσης μεταξύ των δυο μερών και η χρήση εξειδικευμένων πρωτοκόλλων επικοινωνίας. Και για να πούμε τα ίδια πράγματα με άλλες λέξεις, αυτές οι τεχνολογίες είναι ελάχιστα φιλικές με το Internet.

Τυπικό παράδειγμα, αν θέλαμε να εγκαταστήσουμε ένα σύστημα επικοινωνίας μεταξύ της εταιρείας μας και των πελατών της σε διάφορα σημεία του πλανήτη, χρησιμοποιώντας DCOM, θα έπρεπε καταρχήν να έχουμε μόνιμη και αξιόπιστη σύνδεση με αυτούς μέσω του Internet. Αρκετά δύσκολο βέβαια, λόγω της φύσης του δικτύου. Επίσης, θα έπρεπε να ζητήσουμε από τον διαχειριστή του δικτύου μας, να μας επιτρέψει να χρησιμοποιήσουμε ελεύθερα όλες τις απαραίτητες πόρτες που απαιτούν τα DCOM για την επικοινωνία τους. Επίσης πολύ δύσκολο, ίσως και αδύνατο! Ακόμα, όμως, και να μας επιτρεπόταν να κάνουμε κάτι τέτοιο, το πιθανότερο είναι ότι δεν θα επιτρεπόταν στους πελάτες μας, μια που θα έπρεπε και αυτοί να κάνουν αντίστοιχες τροποποιήσεις στο δικό τους δίκτυο.

Στα παραπάνω θέματα, καθώς και σε μερικά άλλα που φαίνονται στην επόμενη λίστα, δίνουν λύσεις τα web services:

- **Διαλειτουργικότητα:** δυνατότητα επικοινωνίας μεταξύ συστημάτων που λειτουργούν με διαφορετικές πλατφόρμες με τη χρήση κοινά αποδεκτών πρωτοκόλλων (π.χ. PC-Solaris, PC-Palmtop κλπ.).
- **Εύκολη Δικτυακή Υλοποίηση:** δυνατότητα λειτουργίας μέσω διάφορων αρχιτεκτονικών δικτύων, χωρίς ιδιαίτερες απαιτήσεις που παραβαίνουν τις συνήθεις ρυθμίσεις τους, κυρίως προς χάριν της

---

<sup>15</sup> Εφαρμογών-dll που δημιουργούνται για παράδειγμα χρησιμοποιώντας Visual Basic 6 ή C++

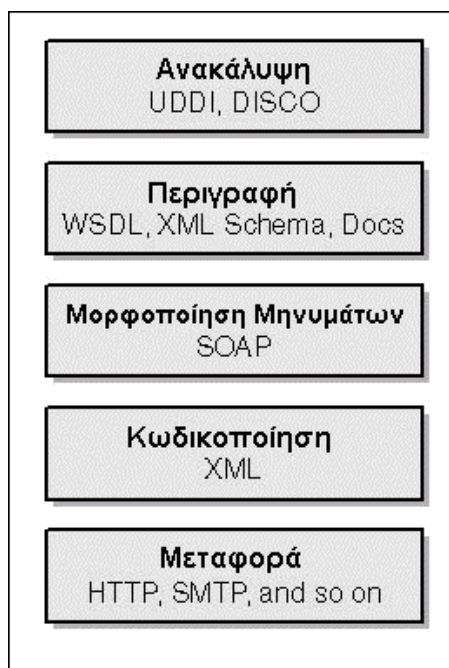
<sup>16</sup> Το DCOM έχει εισέλθει και σε μερικές ακόμα πλατφόρμες, χωρίς όμως να γνωρίσει μεγάλη αποδοχή σε αυτές.

ασφάλειας, εξασφαλίζοντας σταθερά επίπεδα αξιοπιστίας του μέσου μεταφοράς.

- **Υλοποίηση με διάφορες γλώσσες προγραμματισμού:** δυνατότητα ακόμα και ταυτόχρονης χρήσης διαφορετικών γλωσσών στις δυο επικοινωνούσες πλευρές.
- **Τρόποι Τεκμηρίωσης:** Παροχή τρόπων τεκμηρίωσης και περιγραφής των υπηρεσιών και λειτουργιών που προσφέρονται για επικοινωνία μεταξύ δυο συστημάτων<sup>17</sup>.

## 6.2 Ορισμός των Web Services

Τα Web Services είναι «ένα σύνολο πρωτοκόλλων (protocol stack), δηλαδή ένα σύνολο κανόνων, για την επικοινωνία εφαρμογών και συσκευών, ανεξάρτητα από την πλατφόρμα που χρησιμοποιούν ή τον σκοπό για τον οποία κατασκευάστηκαν». Αυτό το σύνολο των πρωτοκόλλων παρουσιάζεται στο Σχήμα 5 και περιγράφεται παρακάτω:



Σχήμα 5 – Επίπεδα των Web Services<sup>18</sup>

<sup>17</sup> Ενδεικτικά αναφέρεται ότι ο τρόπος τεκμηρίωσης και αναπαράστασης των παρεχόμενων λειτουργιών στις τεχνολογίες COM/DCOM είναι η χρήση type libraries.

<sup>18</sup> Scott Short – Building XML Web Services for the Microsoft .NET platform

- **Ανακάλυψη (DISCOVERY):** Δίνει τη δυνατότητα σε κάποιο χρήστη<sup>19</sup> που θέλει να χρησιμοποιήσει υπηρεσίες που παρέχονται από ένα σύστημα, να βρει την τοποθεσία αυτού του συστήματος. Πρόκειται ίσως για έναν τρόπο διαφήμισης των Web Services. Όπως η λίστα προγραμμάτων του μενού Start->Programs μας ενημερώνει για τα προγράμματα που είναι διαθέσιμα στον υπολογιστή μας, έτσι και μια υπηρεσία Discovery μας δίνει διαθέσιμους Servers και Web Services που μπορούμε να χρησιμοποιήσουμε. Ο βασικότερη υπηρεσία ανακάλυψης είναι το UDDI (Universal Description, Discovery and Integration). Μπορεί κανείς, ελεύθερα να δει μια λίστα διαθέσιμων Web Services παγκοσμίως στη διεύθυνση του UDDI: [www.uddi.org](http://www.uddi.org).
- **Περιγραφή:** Πρόκειται για τον τρόπο με τον οποίο περιγράφεται με λεπτομέρεια ο τρόπος χρήσης και οι επιστρεφόμενες υπηρεσίες ενός Web Service. Τα δυο βασικά μέρη του είναι:
  - **Η περιγραφή των παραμέτρων (WSDL):** (Web Services Description Language) Περιλαμβάνει απαρίθμηση των παραμέτρων και των τύπων τους (αλφαριθμητικό, ακέραιος κλπ.), που πρέπει να ορίσει ο χρήστης του Web Service, καθώς και τον τύπο των επιστρεφόμενων τιμών. Αν μιλούσαμε για την τεχνολογία DCOM, αντίστοιχη πληροφορία θα μπορούσαμε να αναζητήσουμε στις type libraries των αρχείων που αποτελούν την DCOM εφαρμογή. Στην περίπτωση των Web Services, η περιγραφή αυτή γίνεται μέσω ενός εγγράφου WSDL. Πρόκειται για ένα XML Document και μια σειρά από tags με τα οποία μπορούν να περιγραφούν τα παραπάνω στοιχεία. Η ανάπτυξη της WSDL έγινε από τις εταιρίες Arriba, IBM και Microsoft<sup>20</sup>.

---

<sup>19</sup> Χρήστης ενός web service συνήθως είναι μια εφαρμογή η οποία υλοποιείται από κάποιο προγραμματιστή και όχι ο τελικός χρήστης που έμμεσα θα χρησιμοποιήσει την τελική υπηρεσία μέσω ενός φιλικού περιβάλλοντος, όπως μια windows εφαρμογή.

<sup>20</sup> *A Field Guide to Services On Demand and Sun™ ONE*, Diana Reichardt, Sun Microsystems

- **Η περιγραφή του βασικού μηνύματος (XML Schema):**  
Όπως περιγράφεται και παρακάτω, το κυρίως μήνυμα που μεταφέρεται μέσω ενός Web Service μπορεί να είναι ένα απλό κείμενο (π.χ. βλ. Σχήμα 6, μεταξύ των tags <soap:body>). Τις περισσότερες φορές όμως, χρειάζεται να μεταφέρουμε αρκετά πιο πολύπλοκα μηνύματα, τα οποία έχει καθιερωθεί να κωδικοποιούμε με τη μορφή ενός εγγράφου XML. Επειδή όμως η σύνταξη ενός εγγράφου XML είναι ελεύθερη, με την έννοια ότι κανείς μπορεί να συντάξει με πάρα πολλούς τρόπους την ίδια πληροφορία, απαιτείται ένας τρόπος περιγραφής του. Μια γλώσσα η οποία περιγράφει ένα XML έγγραφο είναι το XML Schema, του οποίου η χρήση συνηθίζεται και στην περίπτωση των Web Services, όταν αυτά μεταφέρουν έγγραφα XML. Περισσότερες λεπτομέρειες για το XML Schema υπάρχουν στο κεφάλαιο 7.
- **Μορφοποίηση Μηνυμάτων (SOAP):** Ο τρόπος με τον οποίο θα γίνει η συνολική αναπαράσταση των παραμέτρων και των επιστρεφόμενων τιμών κ.α. στοιχείων που πρέπει να μεταφερθούν μεταξύ των επικοινωνούντων πλευρών. Η μορφοποίηση – πρωτόκολλο, που χρησιμοποιείται στα Web Services, ονομάζεται SOAP (Simple Object Access Protocol) και πρόκειται για ένα συγκεκριμένης μορφής έγγραφο XML του οποίου τα βασικά σημεία φαίνονται παρακάτω. Επίσης, στο Σχήμα 6 φαίνεται ένα παράδειγμα μηνύματος SOAP, στο οποίο η Suzanne (πεδίο From) υπενθυμίζει στον Scott (πεδίο To) ένα από τα συζυγικά του καθήκοντα! Τα τρία, λοιπόν, βασικά μέρη ενός μηνύματος SOAP είναι:
  - **Η επικεφαλίδα (header):** Περιλαμβάνει πληροφορίες του συστήματος που κάνει την κλήση του Web Service, πληροφορίες χρέωσης (αν η υπηρεσία δεν προσφέρεται δωρεάν) καθώς και άλλες πληροφορίες όπως η απευθείας δρομολόγηση του μηνύματος σε άλλους υπολογιστές,

ετικέτες πιστοποίησης<sup>21</sup>, πληροφορίες κρυπτογράφησης (π.χ. ψηφιακές υπογραφές) κλπ.

- **Το σώμα (body):** Φέρει το κύριο μέρος του μηνύματος και των πληροφοριών που μεταφέρονται. Μπορεί να περιλαμβάνει μια απλή λέξη, ένα δεύτερο έγγραφο XML ή οτιδήποτε μπορεί να αναπαρασταθεί ή να κωδικοποιηθεί με μορφή κειμένου, αρκεί να έχει συμφωνηθεί μεταξύ των δυο επικοινωνούντων πλευρών.
- **Η περιοχή σφαλμάτων (fault):** Σε περίπτωση που η εκτέλεση της υπηρεσίας αποτύχει, σε αυτή την περιοχή καταγράφεται η περιγραφή του προβλήματος.

Το SOAP είναι μια τεχνολογία που αναπτύχθηκε από τις εταιρίες DevelopMentor, IBM, Lotus, Microsoft και Userland<sup>22</sup>.

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <!--Optional header information goes here. -->
    <To>Scott</To>
    <From>Suzanne</From>
  </soap:Header>
  <soap:Body>
    <!--Message goes here. -->
    Please pick up some milk on your way home from work.
  </soap:Body>
</soap:Envelope>
```

**Σχήμα 6 – Παράδειγμα ενός μηνύματος SOAP<sup>23</sup>**

---

<sup>21</sup>Οι ετικέτες πιστοποίησης (authentication tickets) εξασφαλίζουν την αναγνώριση του συστήματος που ξεκινά την επικοινωνία από το σύστημα που την δέχεται. Αποτελούν σημαντική έννοια στην προσπάθεια εξασφάλισης ασφαλών τρόπων επικοινωνίας στην πλατφόρμα του Internet και όχι μόνο. Παραδείγματα είναι οι ετικέτες NT/ Kerberos για πιστοποίηση χρηστών ενός Microsoft Windows 2000 Server, τα.NET Passport tickets κ.α.

<sup>22</sup>*A Field Guide to Services On Demand and Sun™ ONE*, Diana Reichardt, Sun Microsystems

<sup>23</sup> Scott Short – Building XML Web Services for the Microsoft .NET platform

- **Κωδικοποίηση (Encoding):** Στο Σχήμα 6 φαίνεται ένα απλό μήνυμα που μπορεί να μεταφερθεί «πακεταρισμένο» στην μορφή ενός μηνύματος SOAP. Επειδή όμως συνήθως τα μηνύματα που μεταφέρονται είναι αρκετά πιο πολύπλοκα (σκεφτείτε να χρειαστεί, για παράδειγμα, η Suzanne να καταγράψει όλα αυτά που ξέχασε η ίδια να πάρει πριν πάει σπίτι και πρέπει τώρα να φέρει ο Scott!), χρειάζεται ένας καλύτερα δομημένος τρόπος αναπαράστασης των περιεχομένων του σώματος ενός μηνύματος SOAP. Συνήθως για το σκοπό αυτό χρησιμοποιούνται έγγραφα XML σε συνδυασμό με έγγραφα XML Schema (βλ. Κεφάλαιο 7) που τα περιγράφουν.
- **Μεταφορά (Transport):** Αφού σχηματιστεί ένα μήνυμα SOAP και κωδικοποιηθεί στο εσωτερικό του όλη η απαραίτητη πληροφορία, γίνεται η μεταφορά του σε ένα απομακρυσμένο υπολογιστή. Η μεταφορά αυτή μπορεί να γίνει με διάφορους τρόπους, εφόσον έχει συμφωνηθεί μεταξύ των επικοινωνούντων πλευρών, από τη στιγμή που εξασφαλίζεται η απρόσκοπτη μεταφορά του μηνύματος SOAP (θεωρητικά, θα μπορούσε κανείς ακόμα και να εκτυπώσει το μήνυμα και να το στείλει με FAX!). Συνήθως η μεταφορά αυτή γίνεται μέσω της πόρτας 80 του TCP/IP και του πρωτοκόλλου HTTP του Internet. Ο λόγος βέβαια είναι ότι πρόκειται για την πόρτα που θα βρούμε ανοιχτή σχεδόν σε οποιονδήποτε υπολογιστή, χωρίς προβλήματα μπλοκαρίσματος από firewalls. Ο τρόπος αποστολής και παραλαβής του μηνύματος SOAP γίνεται με τις εντολές HTTP/POST και HTTP/GET<sup>24</sup> αντίστοιχα.

### 6.3 Υλοποίηση Web Services με το Visual Studio .NET

Αν προσπαθήσει κανείς να υλοποιήσει τα παραπάνω χαρακτηριστικά με σκοπό να δημιουργήσει ένα web service θα συναντήσει πολλές δυσκολίες

---

<sup>24</sup> Οι εντολές αυτές είναι οι εντολές που ενεργοποιούνται όταν κάνουμε κλικ στο πλήκτρο που κάνει την αποστολή των στοιχείων μιας φόρμας σε μια σελίδα HTML. Σε αυτή την περίπτωση, ο web browser που χρησιμοποιούμε, στέλνει τα δεδομένα που περιλαμβάνει η φόρμα με την εντολή HTTP/POST στη διεύθυνση που ορίζει η HTML σελίδα, ενώ στον web server που αντιστοιχεί σε αυτή τη διεύθυνση, ενεργοποιείται η εντολή HTTP/GET η οποία λαμβάνει τα δεδομένα σε μορφή κειμένου.

και θα πρέπει να ανατρέξει σε αρκετά references, μέχρι τελικά να ικανοποιήσει όλες τις απαιτήσεις των παραπάνω πρωτοκόλλων και γλωσσών (HTTP, SOAP, XML Schema, WSDL).

Για το σκοπό αυτό έχουν αναπτυχθεί διάφορα βοηθητικά εργαλεία που επιταχύνουν αυτή τη διαδικασία. Ένα τέτοιο εργαλείο είναι και το Microsoft SOAP Toolkit, το οποίο αναλαμβάνει την διαδικασία δημιουργίας του SOAP μηνύματος και την αποστολή του μέσω του HTTP. Περιλαμβάνει επίσης άλλα εργαλεία, όπως έναν WSDL generator ο οποίος μπορεί να διαβάσει τον κώδικα που έχει γράψει ο προγραμματιστής και να δημιουργήσει αυτόματα ένα έγγραφο WSDL που περιγράφει τις διαθέσιμες συναρτήσεις και τις παραμέτρους.

Άλλες εργασίες, όπως η δημοσίευση σε ένα web server καθώς και η χρήση του web service από κάποιο τρίτο προγραμματιστή, συνήθως γίνεται χρησιμοποιώντας άλλα εργαλεία.

Παρόλα αυτά, με τη βοήθεια του Visual Studio .NET, μπορούμε να δημιουργήσουμε Web Services πολύ εύκολα και δαπανώντας ελάχιστο χρόνο ασχολούμενοι με τα παραπάνω πρωτόκολλα. Η λεπτομερής γνώση τους μπορεί ωστόσο να βοηθήσει σημαντικά στη χρήση του Visual Studio .NET και να κάνει τη χρήση του ακόμα πιο αποδοτική. Η υλοποίηση των Web Services με τη βοήθεια του Visual Studio .NET γίνεται μέσω εφαρμογών ASP.NET<sup>25</sup>, δηλαδή web εφαρμογών<sup>26</sup>.

Δημιουργώντας μια εφαρμογή Web Services στο Visual Studio .NET (μενού File->New->Project... -> ASP.NET Web Service) δημιουργείται μια σχεδόν κενή εφαρμογή, με προεπιλεγμένο όνομα WebService1 και αυτόματα δημιουργείται και ένα αντίστοιχο web folder στον IIS, στη διεύθυνση <http://localhost/webservice1>. Αυτή η σχεδόν κενή εφαρμογή, περιλαμβάνει

---

<sup>25</sup> Η ASP.NET αναφέρεται λεπτομερέστερα στο κεφάλαιο 8, παρόλα αυτά, επειδή τα web services κάνουν ελάχιστη χρήση της, μπορείτε να συνεχίσετε να διαβάζεται αυτό το κεφάλαιο, χωρίς να χρειάζεται απαραίτητα να ανατρέξετε πρώτα στο κεφάλαιο 8.

<sup>26</sup> Για τη δημιουργία Web Services, όπως και Web Εφαρμογών, απαιτείται η εγκατάσταση του Microsoft Visual Studio .NET σε υπολογιστή που περιλαμβάνει τον IIS 5.0 (Internet Information Server) σε λειτουργικό σύστημα τουλάχιστον Windows 2000/XP Professional.

το γνωστό παράδειγμα 'Hello World'. Ο κώδικας, σε VB.NET και C#, με τον οποίο υλοποιείται αυτό το web service φαίνεται στο Παράδειγμα 6.

```
VB7
<WebMethod()> Public Function HelloWorld() As String
    HelloWorld = "Hello World"
End Function

C#
[WebMethod]
public string HelloWorld()
{
    return "Hello World";
}
```

### Παράδειγμα 6 - Τμήμα εφαρμογής ASP.NET Web Service

Εκτελώντας την εφαρμογή, ανοίγει ο Internet Explorer στη διεύθυνση <http://localhost/WebService1/Service1.asmx>, από την οποία μπορούμε να εκτελέσουμε τις παρεχόμενες υπηρεσίες. Προφανώς, το Web Service δεν θα χρησιμοποιηθεί τελικά μέσω του Internet Explorer αλλά από κάποια άλλη εφαρμογή, εκτελώντας εντολές HTTP POST/GET και ανταλλάσσοντας μηνύματα SOAP με τον Web Server, δηλαδή τον τοπικό IIS. Επειδή όμως αυτή τη δυνατότητα την έχει και ο Internet Explorer, το Visual Studio .NET δημιουργεί μερικές σελίδες μέσω των οποίων μπορούμε να δοκιμάσουμε το Web Service, πριν δημιουργήσουμε κάποια άλλη εφαρμογή που θα κάνει χρήση του τελικά.

```
- <s:element name="HelloWorld">
  <s:complexType />
</s:element>
- <s:element name="HelloWorldResponse">
  - <s:complexType>
    - <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="HelloWorldResult" type="s:string" />
    </s:sequence>
  </s:complexType>
</s:element>
```

### Σχήμα 7 - Τμήμα του WSDL εγγράφου που αντιστοιχεί στο Παράδειγμα 6

Μέσω αυτών των σελίδων μπορούμε να δούμε και το WSDL έγγραφο που περιγράφει τις παρεχόμενες υπηρεσίες και αποτελεί, όπως αναφέρθηκε και στην παράγραφο 6.2, την type library του Web Service. Το έγγραφο αυτό δημιουργείται δυναμικά όταν ζητηθεί, ανάλογα με τον κώδικα που γράφηκε κατά την ανάπτυξη (δεν υπάρχει με τη μορφή αρχείου στην εφαρμογή). Ένα μέρος του εγγράφου αυτού φαίνεται στο Σχήμα 7 (είναι αρκετά μεγάλο για να

συμπεριληφθεί ολόκληρο), στο οποίο φαίνεται ότι παρέχεται μια συνάρτηση HelloWorld που επιστρέφει μια τιμή τύπου string (αλφαριθμητικό).

#### **6.4 Προτάσεις υλοποίησης λύσεων με Web Services**

Τα Web Services ανοίγουν ένα μεγάλο κεφάλαιο στην επικοινωνία μεταξύ εφαρμογών που λειτουργούν πάνω στο Internet. Αυτό πρακτικά σημαίνει ότι ανοίγει μια πολύ σημαντική πλατφόρμα υλοποίησης εφαρμογών, κυρίως σε επίπεδο B2B (Business-to-Business) καθώς και στη συνεργασία μεταξύ επιχειρήσεων και κυβερνητικών οργανισμών. Η σημασία μιας τέτοιας προσέγγισης πηγάζει κυρίως από την καθιέρωση και από κοινού αποδοχή των προτύπων που κάνουν χρήση τα Web Services (π.χ. Internet/HTTP, XML Schema), καθώς και η από κοινού αποδοχή των προτύπων των τεχνολογιών που πηγάζουν από αυτά (π.χ. WSDL, SOAP).

Τα πρότυπα των XML και XML Schema που χρησιμοποιούνται για την μεταφορά των πληροφοριών, είναι πλέον απλά μια πραγματικότητα στο χώρο της πληροφορικής τεχνολογίας. Η πλατφόρμα του WWW (τεχνικά το πρωτόκολλο HTTP-port 80) πάνω στην οποία λειτουργούν τα Web Services, είναι μάλλον απίθανο να μην είναι διαθέσιμη μεταξύ των δυο συνεργαζόμενων πλευρών. Άλλα πρωτόκολλα, όπως το SOAP και η γλώσσα WSDL για την περιγραφή των Web Services αποτελούν μάλλον και το μεγαλύτερο πλεονέκτημά τους, μια που αυτά θέτουν τις βάσεις δημιουργίας αξιόπιστων υπηρεσιών, αυστηρά καθορισμένων απαιτήσεων και παρεχόμενων υπηρεσιών, χωρίς κάποια ιδιαίτερη συνεννόηση μεταξύ των εταιρειών και οργανισμών που θέλουν να επικοινωνήσουν.

Αρκεί κανείς να γνωρίζει τη διεύθυνση URL - αν ακόμα και αυτή είναι άγνωστη, μπορεί να αναζητηθεί με τη βοήθεια των μηχανισμών ανακάλυψης (βλ. παράγραφο 6.2) - του υπολογιστή που λειτουργεί το Web Service, για να μπορέσει άμεσα ή έμμεσα, μέσω του περιβάλλοντος ανάπτυξης λογισμικού που χρησιμοποιεί, να πάρει την πληροφορία για τη χρήση της υπηρεσίας από το έγγραφο WSDL που το περιγράφει. Δεν απαιτείται λοιπόν κάποια ιδιαίτερη τεκμηρίωση της υπηρεσίας μεταξύ των δυο επικοινωνούντων πλευρών, όχι τουλάχιστον σε τεχνικό επίπεδο. Ενδεχομένως, ανάλογα με τη φύση της υπηρεσίας, να απαιτείται απλά μια περιγραφή των επιχειρηματικών κανόνων που την διέπουν.

Μερικά απλά παραδείγματα εφαρμογής μιας τέτοιου είδους συνεργασίας με τη χρήση Web Services είναι τα παρακάτω:

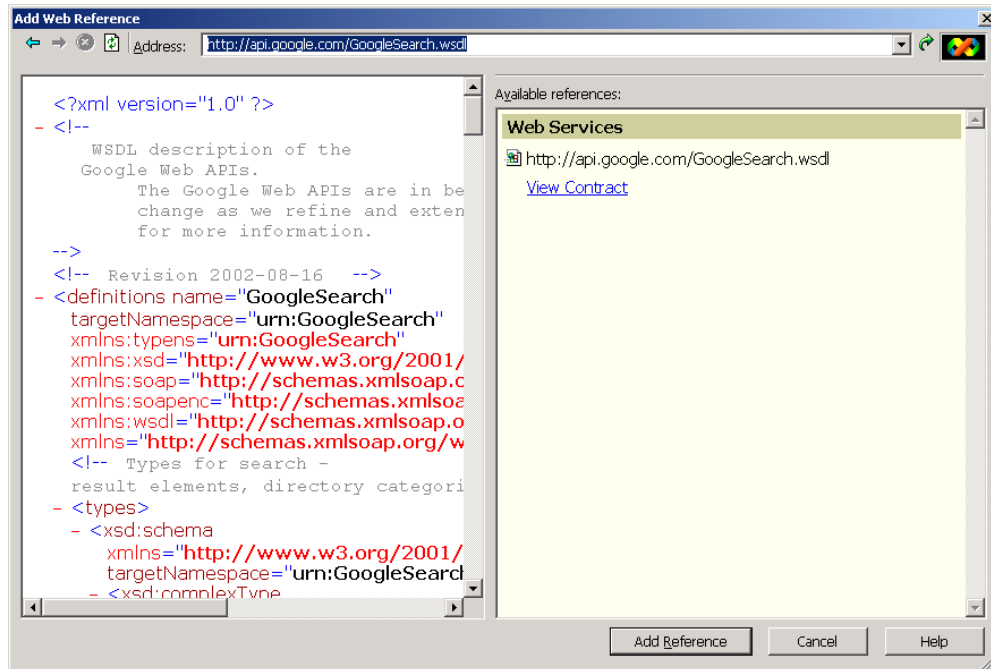
- Παροχή τραπεζικών υπηρεσιών οι οποίες θα είναι προσβάσιμες από εφαρμογές που χρησιμοποιεί ένας τελικός χρήστης, χωρίς να χρειάζεται ο τελευταίος να κάνει χρήση αυτών από το τυπικό web site της τράπεζας.
- Παροχή υπηρεσιών από κυβερνητικές υπηρεσίες που θα χρησιμοποιούνται απευθείας από πληροφοριακά συστήματα ιδιωτικών εταιρειών ή άλλων κυβερνητικών υπηρεσιών, για την λήψη πληροφοριών.
- Παροχή υπηρεσιών εκτέλεσης πολύπλοκων αλγορίθμων μεγάλης σημασίας, που δεν θα θέλαμε να φύγουν έξω από τα φυσικά όρια μιας επιχείρησης, πανεπιστημίου κλπ. Σε αυτή την περίπτωση θα μπορούσε να γίνεται αποστολή των στοιχείων εισόδου με τη μορφή μηνύματος SOAP και να επιστρέφεται με τον ίδιο τρόπο η απάντηση. Αυτή η επικοινωνία μπορεί να αναλαμβάνεται από μια εφαρμογή που θα λειτουργεί στους τελικούς χρήστες και θα αναλαμβάνει τη δημιουργία και προβολή αυτών των μηνυμάτων μέσω ενός φιλικού περιβάλλοντος διεπαφής.
- Ανάπτυξη καταμεμημένων συστημάτων, όπου μέρος της επεξεργασίας αναλαμβάνεται από απομακρυσμένους υπολογιστές σε άλλα σημεία του πλανήτη.
- Κατανομή φόρτου διακομιστών του Internet, με πρόσβαση πολλών διακομιστών που μπορεί να βρίσκονται σε διάφορα σημεία του πλανήτη, σε κεντρικές βάσεις δεδομένων (Web Farms).

Θα μπορούσαν να αναφερθούν αμέτρητα παρόμοια παραδείγματα. Αρκεί κανείς να επισκεφτεί του καταλόγους του UDDI (<http://www.uddi.org>, <http://uddi.ibm.com>, <http://uddi.microsoft.com>), για να δει χιλιάδες παραδείγματα από Web Services διαθέσιμα προς χρήση.

## **6.5 Παράδειγμα χρήσης του Web Service του Google**

Σε αυτή την παράγραφο θα γίνει η μια παρουσίαση του τρόπου χρήσης ενός υπάρχοντος web service, με τη βοήθεια του Visual Studio .NET. Για το

σκοπό αυτό επιλέχθηκε ένα αρκετά χαρακτηριστικό Web Service, αυτό της γνωστής μηχανής αναζήτησης Google (www.google.com), με τη χρήση του οποίου μπορούμε να πάρουμε τα αποτελέσματα μιας αναζήτησης στο Internet.



**Σχήμα 8 - Προσθήκη Web Reference σε ένα .NET Project**

Το συγκεκριμένο web service, το οποίο ονομάζεται Google Web API (Application Programming Interface)<sup>27</sup>, δίνει τη δυνατότητα σε προγραμματιστές να ενσωματώσουν στα προγράμματά τους τη δυνατότητα αναζήτησης μεταξύ εκατομμυρίων σελίδων. Πριν δούμε πως χρησιμοποιείται η υπηρεσία μέσω του Visual Studio .NET, θα ήταν ίσως ενδιαφέρον να ρίξουμε μια ματιά στο WSDL που περιγράφει τις διαθέσιμες λειτουργίες της. Άλλωστε αυτό το έγγραφο, μαζί με την πολυπλοκότητα που το διακρίνει, αποκρύπτεται, λόγω της φιλοσοφίας του Visual Studio, από τον προγραμματιστή, αφήνοντάς τον να χρησιμοποιήσει τις διαθέσιμες λειτουργίες του, μέσω ενός φιλικού και αντικειμενοστραφούς interface. Το έγγραφο WSDL παρουσιάζεται στο Παράρτημα 3 και με μια προσεκτική ματιά σε αυτό, διακρίνονται οι διάφορες λειτουργίες του.

---

<sup>27</sup> <http://www.google.com/apis>

Για παράδειγμα, στην αρχή του εγγράφου διακρίνεται το στοιχείο *GoogleSearchResult*, το οποίο φέρει διάφορες ιδιότητες, όπως *documentFiltering* (Boolean), *searchQuery* (string), *startIndex* (int), *endIndex* (int) κλπ. Αυτό το στοιχείο, όπως θα δούμε παρακάτω, χρησιμοποιείται για την εκτέλεση μιας αναζήτησης. Στα αποτελέσματά της αναζήτησης, μεταξύ άλλων, είναι και το στοιχείο *ResultElement*, το οποίο διακρίνεται αμέσως μετά, στο έγγραφο WSDL. Αυτό φέρει ιδιότητες όπως *summary* (string), *URL* (string), *title* (string) κλπ.

Για να είναι δυνατή η χρήση ενός Web Service μέσω μιας εφαρμογής που αναπτύσσεται με το Visual Studio, αρκεί να δημιουργηθεί μια αναφορά σε αυτό (Web Reference), όπως ακριβώς γίνεται και για να είναι δυνατή η χρήση ενός απλού dll (δηλαδή μιας Class Library - βλ. παράγραφο 2.2). Η αναφορά αυτή προστίθεται με τη βοήθεια του παραθύρου που φαίνεται στο Σχήμα 8 και το οποίο εμφανίζεται κάνοντας κλικ στο μενού Project->Add Web Reference του Visual Studio. Πληκτρολογώντας τη διεύθυνση του Web Service στο αντίστοιχο πεδίο, η οποία στην περίπτωση του Google είναι η <http://api.google.com/GoogleSearch.wsdl> και πατώντας το πλήκτρο Add Reference, το Visual Studio υλοποιεί όλα τα πρωτόκολλα που απαρτίζουν ένα Web Service και περιγράφηκαν στην παράγραφο 6.2. Η υλοποίηση αυτή γίνεται με την αυτόματη δημιουργία κλάσεων που αποτελούν μια αντικειμενοστραφή μορφή του Web Service. Η δομή των κλάσεων που δημιουργούνται στην περίπτωση των Google APIs φαίνεται στο Παράρτημα 3B.

Γίνεται εύκολα αντιληπτή η αντιστοιχία των κλάσεων και των περιεχομένων τους με τα στοιχεία του εγγράφου WSDL. Ουσιαστικά, η δημιουργία ενός αντικειμένου βάσει μιας από τις κλάσεις και η χρήση μιας συνάρτησής της, έχει ως αποτέλεσμα τη δημιουργία των κατάλληλων εγγράφων XML, που ικανοποιούν το έγγραφο WSDL, το πακετάρισμά τους σε μηνύματα SOAP και την αποστολή τους μέσω εντολών HTTP POST στον Web Server του Google, όπως ακριβώς προβλέπουν τα πρωτόκολλα των Web Services. Μετά την απάντηση του τελευταίου, γίνεται μετατροπή του αποτελέσματος στα αντίστοιχα αντικείμενα, τα οποία επιστρέφονται ως αποτελέσματα της συνάρτησης που εκτελέστηκε.

Στο παρακάτω παράδειγμα, γίνεται η εκτέλεση μιας αναζήτησης, η οποία θα επιστρέψει τις 10 πρώτες σελίδες που περιέχουν την φράση 'University of Macedonia':

```
Dim sGoogle As New Google.GoogleSearchService()
Dim sLicenseKey As String = "8W2+N/hQFH1lOZJ7OuPbufgrT7hKA1Q4"

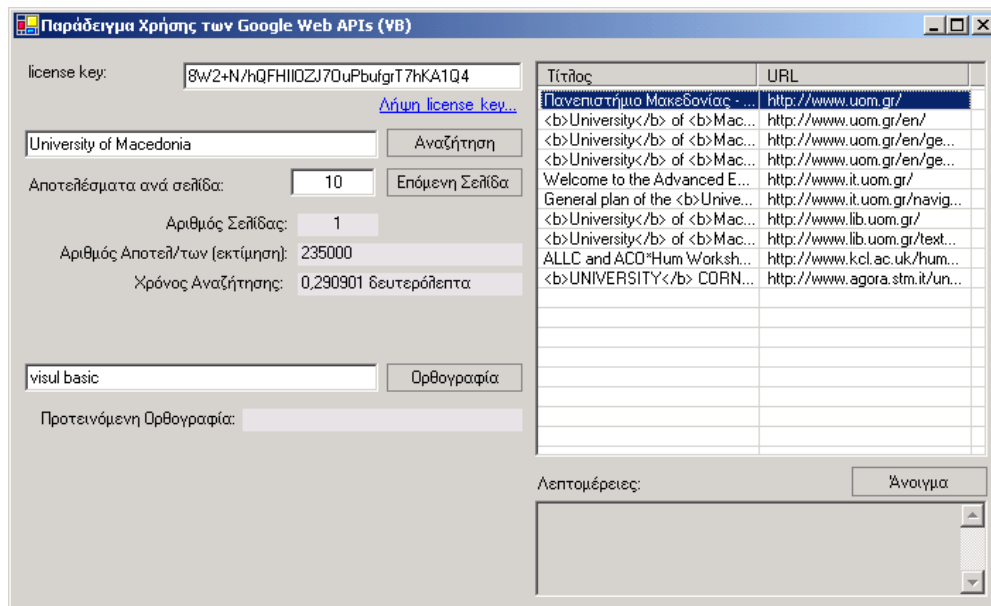
' εκτέλεσης της εντολής αναζήτησης
Dim r As Google.GoogleSearchResult =
sGoogle.doGoogleSearch(sLicenseKey, 0, 10, "University of Macedonia",
False, "", False, "", "", "")

' εμφάνιση αριθμού αποτελεσμάτων και χρόνου αναζήτησης
Console.WriteLine("Χρόνος αναζήτησης:" & r.searchTime)
Console.WriteLine("Αρ.Αποτελ/των:" & r.estimatedTotalResultsCount)

' εμφάνιση αποτελεσμάτων
Dim cResult As Google.ResultElement
For Each cResult In r.resultElements
    Console.WriteLine(cResult.title)
    Console.WriteLine(cResult.URL)
Next
```

Στο συγκεκριμένο παράδειγμα, αρχικοποιείται ένα αντικείμενο Google.GoogleSearchService (στη μεταβλητή sGoogle) και εκτελείται η συνάρτηση doGoogleSearch με τις παραμέτρους που φαίνονται στην 4<sup>η</sup> και 5<sup>η</sup> γραμμή του κώδικα. Το αποτέλεσμα αυτής της συνάρτησης είναι ένα αντικείμενο Google.GoogleSearchResult (μεταβλητή r), το οποίο χρησιμοποιείται για να γραφούν τα αποτελέσματα της αναζήτησης στην κονσόλα των Windows (εντολή Console.WriteLine). Η μεταβλητή sLicenseKey, η οποία είναι η πρώτη παράμετρος στη συνάρτηση doGoogleSearch, φέρει ένα αλφαριθμητικό κλειδί (license key) το οποίο αποτελεί την άδεια για τη χρήση του Web Service. Μπορεί κανείς να αποκτήσει ένα τέτοιο κλειδί δωρεάν, στη διεύθυνση <http://www.google.com/apis>.

Λεπτομέρειες για την κάθε ιδιότητα και συνάρτηση των αντικειμένων μπορεί κανείς να βρει στη διεύθυνση <http://www.google.com/apis/reference.html>, η οποία αποτελεί το reference των Google API.



**Σχήμα 9 - .NET Φόρμα αναζήτησης μέσω του Google**

Ο πλήρης κώδικας του παραδείγματος βρίσκεται στο CD-ROM που συνοδεύει αυτή την εργασία, ενώ το κεντρικό παράθυρο του παραδείγματος, με τα αποτελέσματα της προηγούμενης αναζήτησης φαίνεται στο παραπάνω σχήμα.

## 7. XML Schema

### 7.1 Αναπαράσταση και περιγραφή πληροφορίας σε XML

Η γλώσσα XML έχει πλέον καθιερωθεί ως γλώσσα αναπαράστασης δεδομένων, τόσο στο χώρο του Internet όσο και στο χώρο των desktop εφαρμογών. Η ελευθερία στο σχηματισμό και στον τρόπο αναπαράστασης των δεδομένων δίνει τη δυνατότητα να δημιουργηθούν μοντέλα δεδομένων σε οποιαδήποτε μορφή, όπως ιεραρχική και σχεσιακή ή ακόμα και συνδυασμός τους, ανάλογα με τις επιχειρησιακές ανάγκες που καλύπτει η κάθε εφαρμογή. Άλλος ένας βαθμός ελευθερίας που προσφέρεται από τα XML, είναι η οργάνωση των δεδομένων σε elements<sup>28</sup> και attributes<sup>29</sup>. Στο Σχήμα 10 φαίνεται αυτή ακριβώς η περίπτωση.

```
<?xml version="1.0"?>
<purchaseOrder xmlns="http://myCompany.org/po.xsd" orderDate="1999-10-20">
  <shipTo country="US">
    <name>Alice Smith</name>
    <street>123 Maple Street</street>
    <city>Mill Valley</city>
    <state>CA</state>
    <zip>90952</zip>
  </shipTo>
</purchaseOrder>
```

α) Αναπαράσταση πληροφορίας σε elements

---

```
<?xml version="1.0"?>
<purchaseOrder xmlns="http://myCompany.org/po1.xsd" orderDate="1999-10-20">
  <shipTo country="US" name="Alice Smith" street="123 Maple Street"
    city="Mill Valley" state="CA" zip="90952"/>
</purchaseOrder>
```

β) Αναπαράσταση της ίδιας πληροφορίας σε attributes

Σχήμα 10 - Αναπαράσταση της ίδιας πληροφορίας με δυο διαφορετικά XML

Είναι προφανές ότι για να επικοινωνήσουν δυο εφαρμογές μεταξύ τους, για παράδειγμα αποστέλλοντας ένα έγγραφο XML μέσω ενός Web Service (βλ. Κεφάλαιο 6), θα πρέπει να καθοριστεί μεταξύ τους μια κοινή μορφή-

<sup>28</sup> elements ενός XML είναι τα στοιχεία που ξεκινούν με το όνομά τους μεταξύ των συμβόλων < > και κλείνουν με το όνομά τους μεταξύ </ > (π.χ. <shipTo>εσωτερική πληροφορία</shipTo>)

<sup>29</sup> attributes ενός element σε ένα XML, είναι τα στοιχεία που γράφονται στο εσωτερικό του tag που ξεκινά ένα element (π.χ. <shipTo attribute1="πληροφορία" attribute2="πληροφορία">εσωτερική πληροφορία</shipTo>).

αναπαράσταση της πληροφορίας. Αυτό μπορεί να αφορά στην επιλογή ποιας πληροφορίας θα αναπαρίσταται σε elements και ποια σε attributes, στις ακριβείς ονομασίες τους (π.χ. η πληροφορία για τον παραλήπτη – βλ. Σχήμα 10 - να αναφέρεται ως shipTo και όχι SHIP\_TO), στον τύπο των δεδομένων (π.χ. στο πεδίο orderDate αναμένεται ημερομηνία), στον αριθμό φορών που θα μπορεί να συναντάται το κάθε element (π.χ. το element shipTo πρέπει να υπάρχει μια και μόνο μια φορά, ενώ κάποιο άλλο element όπως τα περιεχόμενα της παραγγελίας μπορεί να συναντώνται περισσότερες φορές - cardinality) κ.α.

Ο προτεινόμενος τρόπος με τον οποίο μπορεί να εκφραστεί όλη η παραπάνω πληροφορία (γνωστή στην ορολογία των βάσεων δεδομένων και ως metadata) που αναπαριστά την πληροφορία ενός XML, μπορεί να εκφραστεί με τη βοήθεια ενός εγγράφου XML Schema. Το XML Schema είναι μια γλώσσα, που περιλαμβάνει μια πλούσια συλλογή από tags τα οποία μπορούν να περιγράψουν έγγραφα XML, εξασφαλίζοντας την εγκυρότητά τους ανάλογα με τη χρήση τους. Μπορεί να αναπαρασταθεί σε ένα αυτόνομο αρχείο (συνήθως με επέκταση .xsd), ενώ στοιχεία του σε απλές περιπτώσεις μπορούν να ενσωματωθούν στο έγγραφο XML που περιέχει τα δεδομένα. Η διαχείριση του XML Schema γίνεται από το W3C και αποτελείται από τα παρακάτω δυο έγγραφα-προδιαγραφές (specifications):

- XML Schema Part 1: Structures (<http://www.w3.org/TR/xmlschema-1/>)
- XML Schema Part 2: Datatypes (<http://www.w3.org/TR/xmlschema-2/>)

Στο Σχήμα 11 φαίνεται ένα παράδειγμα XML Schema που περιγράφει τον α' τρόπο αναπαράστασης δεδομένων στο Σχήμα 10 (οι αριθμοί γραμμών υποβοηθούν την ανάγνωσή του και δεν είναι μέρος του εγγράφου). Σε αυτό διακρίνονται διάφορα tags, όπως type, που περιγράφει τον τύπο δεδομένων που θα έχει ένα element ή ένα attribute του τελικού XML, xs:element που υποδεικνύει ότι πρέπει στο συγκεκριμένο σημείο να υπάρχει ένα element, xs:attribute που υποδεικνύει ότι πρέπει στο συγκεκριμένο σημείο να υπάρχει ένα attribute και xs:complexType που συμβολίζει μια ομάδα από elements που μπορεί να συναντάται με τον ίδιο τρόπο σε διάφορα σημεία του XML.

Άλλα tags, που διακρίνονται, είναι το tag minOccurs και αντίστοιχα το tag maxOccurs, τα οποία καθορίζουν πόσες φορές μπορεί να συναντάται ένα στοιχείο στο τελικό XML. Υπάρχουν βέβαια αρκετά παρόμοια tags τα οποία καλύπτουν όλους τους διαφορετικούς τρόπους που μπορεί να αναπαρασταθεί μια πληροφορία σε ένα έγγραφο XML. Στο Παράρτημα 1 φαίνεται μια λίστα με διαθέσιμα tags.

```
1 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
2       targetNamespace="http://myCompany.org/po.xsd"
3       xmlns="http://myCompany.org/po.xsd"
4       elementFormDefault="qualified">
5   <xs:element name="purchaseOrder" type="PurchaseOrderType" />
6   <xs:complexType name="PurchaseOrderType">
7     <xs:sequence>
8       <xs:element name="shipTo" type="USAddress" />
9       <xs:element ref="comment" minOccurs="0" />
10    </xs:sequence>
11    <xs:attribute name="orderDate" type="xs:date" />
12  </xs:complexType>
13  <xs:complexType name="USAddress">
14    <xs:sequence>
15      <xs:element name="name" type="xs:string" />
16      <xs:element name="street" type="xs:string" />
17      <xs:element name="city" type="xs:string" />
18      <xs:element name="state" type="xs:string" />
19      <xs:element name="zip" type="xs:decimal" />
20    </xs:sequence>
21    <xs:attribute name="country" type="xs:NMTOKEN" fixed="US" />
22  </xs:complexType>
23 </xs:schema>
```

Σχήμα 11 - Παράδειγμα εγγράφου XSD / XML Schema

## 7.2 Τύποι Δεδομένων στο XML Schema

Αξίζει μια ιδιαίτερη αναφορά στους τύπους δεδομένων που είναι διαθέσιμοι στο XML Schema. Άλλωστε αποτελούν ξεχωριστό specification και για το W3C, όπως αναφέρεται και στην προηγούμενη παράγραφο.

Το XML Schema υποστηρίζει την περιγραφή και έλεγχο δεδομένων που μπορεί να αντιστοιχούν σε διάφορους τύπους δεδομένων, όπως οι κλασικοί προγραμματιστικοί τύποι string, integer, float, binary, date κλπ. Το σημαντικότερο πλεονέκτημα των τύπων δεδομένων του XML Schema είναι η δυνατότητά τους να αναπαραστήσουν δεδομένα, ανεξάρτητα από την πλατφόρμα από την οποία προέρχονται ή απευθύνονται, καθιστώντας δυνατή τη χρήση εγγράφων XML σε μηνύματα SOAP (βλ. Κεφάλαιο 6).

Οι τύποι δεδομένων ορίζονται σε ένα έγγραφο XML Schema με τη χρήση του attribute 'type', όπως φαίνεται και στο Σχήμα 11, και οι διαθέσιμες τιμές μπορεί να είναι αυτές που υποδεικνύει ο Πίνακας 2. Η περιγραφή του κάθε τύπου δεδομένων αναφέρεται στο Παράρτημα 2.

<i>XML Schema (XSD) type</i>			
anyURI	anyURI	month	short
base64Binary	anyURI	NCName	string
Boolean	anyURI	NCName	time
Byte	anyURI	negativeInteger	timePeriod
Date	anyURI	NMTOKEN	token
dateTime	anyURI	NMTOKENS	unsignedByte
decimal	anyURI	nonNegativeInteger	unsignedInt
Double	anyURI	nonPositiveInteger	unsignedLong
duration	anyURI	normalizedString	unsignedShort
ENTITIES	anyURI	NOTATION	
ENTITY	anyURI	positiveInteger	
Float	anyURI	QName	

Πίνακας 2 - Τύποι δεδομένων του XML Schema<sup>30</sup>

### 7.3 Namespaces

Στο παράδειγμα που φαίνεται στο Σχήμα 10, διακρίνονται στην αρχή του α' XML, το attribute xmlns (σύντμηση των λέξεων xml namespace) ακολουθούμενο από την τιμή "http://myCompany.org/ro.xsd". Αυτό το attribute δηλώνει ότι τα attributes, για παράδειγμα shipTo, name, street κλπ. που ακολουθούν στο έγγραφο XML αναφέρονται στο namespace http://myCompany.org/ro.xsd. Στο β' XML, φαίνεται ότι τα attributes που ακολουθούν αναφέρονται στο namespace http://myCompany.org/ro1.xsd, δηλαδή ακολουθούν τη μορφοποίηση ενός άλλου εγγράφου XSD και επομένως ενός διαφορετικού XML Schema. Ο ρόλος λοιπόν του προσδιορισμού του namespace σε ένα XML element, είναι να προσδιορίσει

<sup>30</sup> .NET Framework Documentation

τη λογική σημασία που έχουν τα elements και attributes που περιέχονται σε αυτό.

Τα namespaces βέβαια, δεν είναι κάτι νέο στο χώρο του προγραμματισμού. Ιδιαίτερο στο χώρο του αντικειμενοστραφούς προγραμματισμού, είναι προφανές ότι μπορούν να υπάρχουν πολλά αντικείμενα με το όνομα shipTo, τα οποία όμως μπορεί να περιέχουν διάφορες ιδιότητες ανάλογα με την έμπνευση και τη φαντασία προγραμματιστή τους και τις επιχειρησιακές ανάγκες που καλύπτουν. Για να μην υπάρχει λοιπόν σύγχυση μεταξύ αντικειμένων που έχουν το ίδιο όνομα, στο χώρο των COM αντικειμένων υπάρχει η έννοια του Class ID. Πρόκειται για ένα μοναδικό κωδικοποιημένο αλφαριθμητικό (GUID) που προσδιορίζει μοναδικά ένα αντικείμενο. Άλλη μια περίπτωση παρόμοιας ονοματολογίας συναντάμε και στον τρόπο που συντάσσεται ένα SQL query. Επειδή τα δεδομένα που θέλουμε να εξάγουμε με ένα query μπορεί να περιλαμβάνουν, για παράδειγμα, δυο πεδία με το ίδιο όνομα που ανήκουν σε διαφορετικούς πίνακες, συμπληρώνουμε το όνομα του πίνακα ακολουθούμενο από μια τελεία και στη συνέχεια το όνομα του πεδίου για να κάνουμε το διαχωρισμό.

```
<?xml version="1.0"?>
<order xmlns="http://myCompany.org/order.xsd">
  <shipTo>George Lucas</shipTo>
  <item xmlns="http://myCompany.org/dvd.xsd">
    <title xmlns="http://myCompany.org/titles.xsd">Star Wars II</title>
    <dvd_region>3</dvd_region>
    <audio_languages>english,greek</audio_languages>
  </item>
  <item xmlns="http://myCompany.org/vhs.xsd">
    <title xmlns="http://myCompany.org/titles.xsd">Star Wars II</title>
    <audio_language>greek</audio_language>
  </item>
  <item xmlns="http://myCompany.org/poster.xsd">
    <title xmlns="http://myCompany.org/titles.xsd">Skywalker</title>
    <size>large</size>
  </item>
</order>
```

### Σχήμα 12 – Χρήση namespaces χωρίς τη χρήση monikers

Επιστρέφοντας στο XML Schema, η απουσία του xmlns από τα υπόλοιπα elements υποδηλώνει ότι ακολουθούν το namespace του πατρικού element. Είναι δυνατόν ένα έγγραφο να περιέχει elements και attributes που αντιστοιχούν σε πολλά namespaces, πράγμα που δηλώνεται σε κάθε ένα από αυτά ξεχωριστά. Επειδή όμως κάτι τέτοιο θα αύξανε την πολυπλοκότητα του εγγράφου αλλά και το μέγεθός του, εισάγεται η έννοια

του moniker. Στο Σχήμα 12 και στο Σχήμα 13, φαίνεται το ίδιο XML χωρίς και με τη χρήση monikers αντίστοιχα. Στο δεύτερο, τα xml namespaces ορίζονται με ένα ψευδώνυμο (π.χ. dvd για το namespace <http://myCompany.org/dvd.xsd>), το οποίο αντικαθιστά τον ορισμό του στη συνέχεια του εγγράφου, κάνοντάς το πιο ευανάγνωστο.

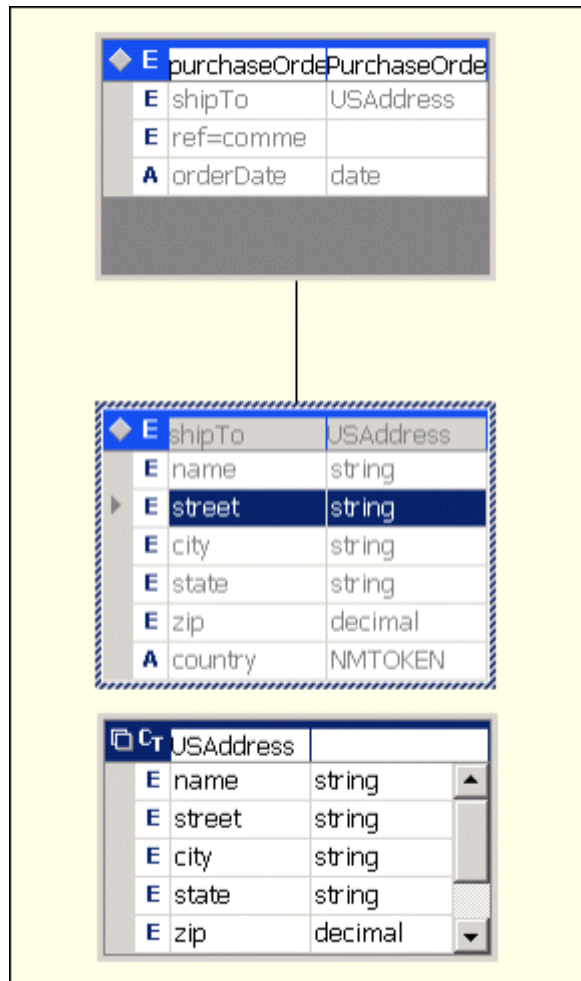
```
<?xml version="1.0"?>
<order:order xmlns:order="http://myCompany.org/order.xsd"
  xmlns:titles="http://myCompany.org/titles.xsd">
  <order:shipTo>George Lucas</order:shipTo>
  <dvd:item xmlns:dvd="http://myCompany.org/dvd.xsd">
    <titles:title>Star Wars II</titles:title>
    <dvd:dvd_region>3</dvd:dvd_region>
    <dvd:audio_languages>english,greek</dvd:audio_languages>
  </dvd:item>
  <vhs:item xmlns:vhs="http://myCompany.org/vhs.xsd">
    <titles:title>Star Wars II</titles:title>
    <vhs:audio_language>greek</vhs:audio_language>
  </vhs:item>
  <pst:item xmlns:pst="http://myCompany.org/poster.xsd">
    <titles:title>Skywalker</titles:title>
    <pst:size>large</pst:size>
  </pst:item>
</order:order>
```

Σχήμα 13 – Χρήση namespaces με τη χρήση monikers

### URL namespaces

Η πρώτη εντύπωση που δίνεται με την πρώτη επαφή με ένα XML που φέρει namespaces με μορφή URL, είναι ότι το έγγραφο αυτό συνδέεται με κάποιο τρόπο με αυτή τη διεύθυνση στο Internet. Κάτι τέτοιο βέβαια δεν είναι σωστό. Το έγγραφο μπορεί να χρησιμοποιηθεί χωρίς να υπάρχει σύνδεση στο Internet και βέβαια τα XML Schema που χρησιμοποιεί μπορούν να βρίσκονται αποθηκευμένα σε άλλες διευθύνσεις, τοπικά αρχεία στον δίσκο ενός υπολογιστή, σε μια βάση δεδομένων σε μορφή κειμένου κλπ. Ο λόγος που χρησιμοποιούνται URLs ως namespaces, δηλαδή για να προσδιοριστούν τα ονόματα των tags ενός XML, είναι γιατί πολύ απλά οι διευθύνσεις αυτές είναι μοναδικές σε παγκόσμιο επίπεδο. Για παράδειγμα, αν μια εταιρεία έχει κατοχυρώσει το domain name [myCompany.org](http://myCompany.org), τότε της δίνεται η δυνατότητα να το χρησιμοποιήσει για τη δημιουργία namespaces και για να ονοματίσει XML Schema που χρησιμοποιεί, εξασφαλίζοντας μια παγκόσμια μοναδικότητα σε αυτά. Όπως αναφέρεται και στην προηγούμενη παράγραφο, τα COM αντικείμενα χρησιμοποιούν GUIDs, π.χ. {77E7E1F7-

2182-4DB9-BA9F-FB989BFB59E0} για να προσδιορίσουν τον εαυτό τους μοναδικά. Είναι προφανές ότι η ονοματολογία με τη μορφή URL είναι πολύ πιο φιλική και ευανάγνωστη.



Σχήμα 14 - Επεξεργασία XML Schema στο Visual Studio .NET

## 7.4 XML Schema στο .NET Framework και το Visual Studio .NET

Στο Σχήμα 11 φαίνεται ένα παράδειγμα ενός αρχείου XSD που χρησιμοποιείται για την περιγραφή του α' XML που φαίνεται στο Σχήμα 10, δηλαδή ενός μικρού XML. Ήδη όμως, ακόμα και το XML Schema ενός τόσο μικρού XML είναι αρκετά δυσανάγνωστο και χρειάζεται κανείς να έχει αρκετή εμπειρία σε αυτή τη γλώσσα, για να καταλάβει πως ακριβώς πρέπει να είναι διαμορφωμένο το XML για να συμμορφώνεται με την απαιτούμενη μορφοποίηση. Σε περιπτώσεις μεγαλύτερων XML, που φέρουν μεγάλο αριθμό πληροφοριών, το αρχείο XSD μπορεί να είναι αρκετές σελίδες.

Η αναπαράσταση του συγκεκριμένου σχήματος μέσω του Visual Studio .NET φαίνεται στο Σχήμα 14, όπου τα πράγματα είναι σαφώς πιο απλά και ευανάγνωστα.

Ο ρόλος των XML Schema είναι σημαντικός για το .NET Framework και ιδιαίτερα για το ADO.NET. Μεταξύ άλλων, όπως αναφέρεται και στη παράγραφο 5.3, το XML Schema αποτελεί συστατικό στοιχείο ενός ADO.NET DataSet. Από το τελευταίο μπορούμε πολύ εύκολα να εξάγουμε την περιεχόμενη σχετική πληροφορία, δηλαδή πως είναι δομημένη η πληροφορία που μπορεί αυτό να περιέχει, χρησιμοποιώντας την εντολή WriteXmlSchema, όπως φαίνεται και στο Παράδειγμα 5.

## 8. ASP.NET

Η ASP.NET (Active Server Pages .NET) είναι μια πλατφόρμα για τη δημιουργία web εφαρμογών και εφαρμογών web services που λειτουργούν μέσω του IIS<sup>31</sup>. Κύριο χαρακτηριστικό της, έναντι άλλων τεχνολογιών για τη δημιουργία παρόμοιων εφαρμογών, είναι η αμεσότητα στη διασύνδεσή της με το .NET Framework και τα χαρακτηριστικά του, όπως η ασφάλεια, ο προγραμματισμός με χρήση του CLR (βλ. Παράγραφο 2.1), η πλούσια βιβλιοθήκη κλάσεων (βλ. Παράγραφο 2.2), η πρόσβαση σε δεδομένα με χρήση του ADO.NET (βλ. Κεφάλαιο 5) κ.α. Ο επίσημος ιστοχώρος της ASP.NET είναι στην προφανή διεύθυνση [www.asp.net](http://www.asp.net).

Τα βασικά στοιχεία που αποτελούν την ASP.NET είναι:

- **Τα εργαλεία ανάπτυξης του Visual Studio .NET:** περιλαμβάνουν διάφορα πρότυπα σελίδων και εργαλεία διαχείρισης για Web Εφαρμογές (πολλά από τα οποία είναι κοινά και για άλλες χρήσεις του Visual Studio)
- **Η κλάσεις του System.Web namespace:** πρόκειται για την περιοχή των κλάσεων του .NET Framework, που περιλαμβάνει λειτουργίες οι οποίες σχετίζονται με το Web, όπως εντολές HTTP, e-mail, διαλειτουργικότητα με browsers κλπ.
- **Server και HTML controls:** στοιχεία που χρησιμοποιούνται στις σελίδες οι οποίες απαρτίζουν μια εφαρμογή, αναλαμβάνοντας μεταξύ άλλων τη διάδραση με τον τελικό χρήστη.
- **Άλλα στοιχεία του .NET Framework,** όχι απαραίτητα μόνο για χρήση σε Web εφαρμογές, όπως το CLR και οι γλώσσες προγραμματισμού (VB.NET, C#, J# κλπ.), βιβλιοθήκες κλάσεων του Framework, ADO.NET κ.α.
- **Internet Information Server:** Χωρίς να αποτελεί στοιχείο του .NET Framework παίζει σημαντικό ρόλο, αφού αναλαμβάνει τη δημοσίευση των εφαρμογών στο Internet. Πρόκειται για τον web

---

<sup>31</sup> Microsoft Internet Information Server.

server της Microsoft, μέσω του οποίου μπορούν να δημοσιευτούν Internet εφαρμογές, όπως εφαρμογές WWW, FTP και SMTP. Είναι μέρος των λειτουργικών συστημάτων Windows NT 4.0, Windows 2000 PRO και Server, Windows XP PRO και Windows .NET Server 2003. Ελάχιστη έκδοση για την εκτέλεση ASP.NET εφαρμογών είναι η έκδοση 5.

Η ASP.NET λειτουργεί στον διακομιστή παράγοντας δυναμικά κώδικα HTML και αποστέλλοντας τον στον browser του χρήστη μέσω του IIS, όπως ακριβώς δηλαδή λειτουργούσε και η προηγούμενη έκδοσή της, καθώς και άλλες παρόμοιες τεχνολογίες, όπως η JSP και η PHP. Παρόλα αυτά, η σύγκριση μεταξύ τους, τουλάχιστον σε ότι αφορά την επιχειρησιακή τους αξία μάλλον είναι αδόκιμη. Θα γίνει γρήγορα αντιληπτό, μετά την παρουσίαση μερικών από τα χαρακτηριστικά της ASP.NET, ότι η επιχειρησιακή της αξία θα μπορούσε να συγκριθεί με γνωστές πλατφόρμες ή γλώσσες προγραμματισμού και ανάπτυξης ολοκληρωμένων εφαρμογών Windows, όπως η Visual Basic ή η Borland Delphi. Παρακάτω φαίνεται μια λίστα με μερικά σημαντικά χαρακτηριστικά της ASP.NET, τα οποία θα αποτελέσουν την αφορμή για περαιτέρω σχολιασμό:

- Χρήση της Visual Basic .NET, C# ή άλλης γλώσσας του CLR (π.χ. J#) για τον κώδικα που λειτουργεί στον server (server-side code). Μεταγλώττιση (compilation) του κώδικα με αποτέλεσμα την γρηγορότερη εκτέλεσή του, χωρίς τη χρήση γλωσσών script, όπως η VBScript και η Jscript.
- Πρόσβαση στις βιβλιοθήκες κλάσεων του .NET Framework (βλ. Παράγραφο 2.2) δίνοντας απεριόριστες δυνατότητες στον server-side κώδικα.
- Αυτόματη διαχείριση της κατάστασης των HTML στοιχείων μεταξύ των κλήσεων των χρηστών χωρίς χρήση κώδικα, με χρήση κρυφών HTML Forms και αποστολή από και προς τον server με διαφανείς, για τον προγραμματιστή, HTTP POST/GET εντολές. Είναι χαρακτηριστικό ότι για την επίτευξη της διατήρησης των στοιχείων που εισάγει ένας χρήστης σε μια σελίδα όπως αυτή στο Σχήμα 15, θα απαιτούνταν η δημιουργία μιας κρυφής φόρμας από τον προγραμματιστή (<FORM> html tag) και η σύνταξη κώδικα στον

server για την επαναφορά των στοιχείων της φόρμας και πάλι στα πλαίσια κειμένου, μετά το πάτημα του πλήκτρου 'Αποστολή' από τον τελικό χρήστη. Στην ASP.NET απαιτείται μόνο η οπτική σχεδίαση των πλαισίων κειμένου επάνω στη σελίδα. Η διαχείριση του HTML και server κώδικα γίνεται αυτόματα.

- Δυνατότητα δημιουργίας, διανομής και χρήσης server controls, που ενσωματώνουν διάφορες λειτουργίες, όπως επιχειρηματικούς κανόνες, στοιχεία διάδρασης με τον χρήστη κλπ.<sup>32</sup> Η έννοια των server controls αναλύεται περισσότερο στην επόμενη παράγραφο αλλά και στην εφαρμογή που παρουσιάζεται στην παράγραφο 8.3.
- Ενσωματωμένος έλεγχος της πρόσβασης σε συγκεκριμένες περιοχές της εφαρμογής με μεθόδους πιστοποίησης και έγκρισης πρόσβασης (authentication/ authorization) που μπορεί να προέρχονται από ένα Active Directory (βλ. Κεφάλαιο 9), από το .NET Passport<sup>33</sup> ή κάποια άλλη μέθοδο πιστοποίησης προσαρμοσμένη στις επιχειρησιακές ανάγκες της κάθε εφαρμογής.
- Υποστήριξη XML, Cascading Style Sheets και άλλων τυποποιήσεων.
- Δυνατότητες cached σελίδων ή στοιχείων που δημιουργούν μεγάλη κίνηση στον server, ανίχνευση και προσαρμογή της εφαρμογής ανάλογα με τον browser του χρήστη, υποστήριξη προσαρμογής εφαρμογής σε άλλες φυσικές γλώσσες (π.χ. αγγλικά, ελληνικά κλπ. στην ίδια εφαρμογή)

---

<sup>32</sup> Η χρήση και διανομή controls είναι ένα από τα στοιχεία που στηρίζεται η επιτυχία της Visual Basic 6, αφού δίνει τη δυνατότητα συνεργασίας μεταξύ προγραμματιστών, επεκτείνοντας απεριόριστα τις δυνατότητες της γλώσσας και των αρχικών ενσωματωμένων δυνατοτήτων της.

<sup>33</sup> Πρόκειται για ένα σύστημα πιστοποίησης της Microsoft, το οποίο δίνει τη δυνατότητα σε κάποιο χρήστη να χρησιμοποιεί πολλά web sites με το ίδιο κωδικό όνομα και μυστικό κωδικό πρόσβασης. Πριν την είσοδο σε προστατευμένες περιοχές ενός web site, ο χρήστης παραπέμπεται, χωρίς κάποια ενέργεια από το ίδιο, στο web site του .NET Passport, όπου δίνει το όνομα και τον κωδικό πρόσβασης που διαθέτει. Η υπηρεσία .NET Passport πιστοποιεί την ταυτότητά του και στη συνέχεια ο χρήστης παραπέμπεται και πάλι στην αρχική εφαρμογή, η οποία πλέον μπορεί να γνωρίζει την ταυτότητά του.

## 8.1 Ο ρόλος της ASP.NET στα σύγχρονα πληροφοριακά συστήματα

Είναι προφανές ότι οι web εφαρμογές θα παίξουν στο μέλλον πολύ σημαντικό ρόλο στην ανάπτυξη πληροφοριακών συστημάτων και αναμένεται να έχουν πολύ μεγαλύτερο βαθμό αποδοχής από το σύνολο των χρηστών, εταιρειών και κυβερνητικών οργανισμών, σε σχέση με άλλες τεχνολογίες όπως οι καταναμημένες Windows εφαρμογές. Τουλάχιστον σε ότι αφορά τη διάδοση πληροφοριών που σχετίζονται με παροχή υπηρεσιών και πώληση προϊόντων, οι web εφαρμογές αποτελούν μονόδρομο, έναντι των εφαρμογών Windows.

Τα εργαλεία και οι τεχνολογίες ανάπτυξης web εφαρμογών, που υπήρχαν μέχρι σήμερα, συνήθως έκαναν χρήση βάσεων δεδομένων δημιουργώντας δυναμικά ιστοσελίδες με απλό User Interface (UI). Το αποτέλεσμα ήταν οι πολύ απλοϊκές δυνατότητες διάδρασης με τον χρήστη, τις περισσότερες φορές με τη μορφή μερικών κλικ του ποντικιού με σκοπό την πλοήγηση μεταξύ των σελίδων. Ταυτόχρονα, η απαιτούμενη προγραμματιστική προσπάθεια για την επίτευξη και αυτών ακόμα των αποτελεσμάτων, ήταν σημαντική. Λύσεις όπως η ASP ή η PHP τις περισσότερες φορές θύμιζαν προσπάθειες προγραμματισμού της δεκαετίας του 80 και της αρχής της δεκαετίας του 90, όπου ελάχιστα δομημένος και μακροσκελής κώδικας, δύσκολα συντηρούμενος και επεκτάσιμος, γραφόταν από μεμονωμένους προγραμματιστές για την επίλυση πολύπλοκων επιχειρησιακών αναγκών. Τα περιβάλλοντα ανάπτυξης που χρησιμοποιούσαν αυτές τις τεχνολογίες ήταν περισσότερο βοηθήματα, παρά εργαλεία ανάπτυξης λογισμικού, αν συγκρινόταν με άλλα, σύγχρονά με αυτά, εργαλεία ανάπτυξης Windows εφαρμογών όπως η Microsoft Visual Basic, η Microsoft/Borland C++ ή η Borland Delphi.

Δεν είναι βέβαια δυνατόν, αυτές οι τεχνολογίες να καλύψουν τις αυξημένες ανάγκες της κοινωνίας της πληροφορίας που δημιουργείται, η οποία χαρακτηρίζεται από αυξημένες απαιτήσεις, ταχύτητα στην απόδοση λύσεων, διαλειτουργικότητα με διάφορα προϊόντα υλικού ή λογισμικού και εφαρμογή των σύγχρονων τυποποιήσεων στο χώρο της ανάπτυξης λογισμικού.

Γι αυτό λοιπόν, ρίχνοντας μια ματιά στην λίστα των χαρακτηριστικών που προαναφέρθηκαν αλλά και στη συνέχεια αυτού του κεφαλαίου, δεν έχουμε

παρά να καταλήξουμε στο συμπέρασμα ότι η ASP.NET είναι μια ολοκληρωμένη πλατφόρμα για την ανάπτυξη Web εφαρμογών, διαχωρίζοντάς την από τα προγενέστερα εργαλεία ανάπτυξης δυναμικών Web Sites.

## 8.2 Σημαντικοί όροι στην ASP.NET

Σε αυτή την παράγραφο επεξηγούνται μερικοί σημαντικοί όροι της ASP.NET.

- **Web Forms**

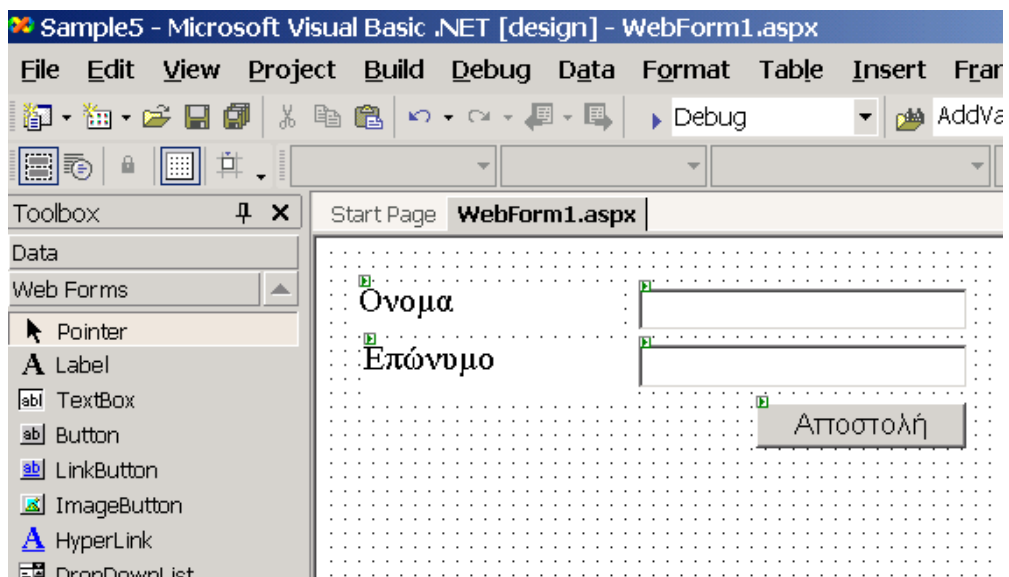
Πρόκειται για μια νέα μέθοδο δημιουργίας σελίδων διάδρασης με τον χρήστη, με τη χρήση κρυφών HTML φορμών που αναλαμβάνουν τη διαχείριση της κατάστασης των στοιχείων HTML μεταξύ των διαδοχικών κλήσεων από τον χρήστη. Αυτό δίνει τη δυνατότητα ανάπτυξης User Interface (UI) με μεθόδους WYSIWYG<sup>34</sup>, ελαχιστοποιώντας τις περιπτώσεις πρόσβασης στον κώδικα HTML, τουλάχιστον στις τυπικές εργασίες ανάπτυξης μιας εφαρμογής. Στο Σχήμα 13 φαίνεται ένα πολύ απλό παράδειγμα, στο οποίο υλοποιείται μια απλή φόρμα. Ακόμα και για αυτήν την απλή περίπτωση, θα χρειαζόταν η επέμβαση στον HTML κώδικα στην περίπτωση της προηγούμενης έκδοσης της ASP. Η κάθε Web Form αντιστοιχεί σε μια σελίδα με επέκταση .aspx.

- **Server Controls**

Τα Server Controls είναι η μεγάλη καινοτομία της ASP.NET και σε αυτή στηρίζεται το γεγονός ότι μπορεί να θεωρηθεί μια ολοκληρωμένη πλατφόρμα για την ανάπτυξη Web Εφαρμογών. Τα Server Controls μπορούν να ενσωματώνουν τμήματα του UI, με τέτοιο τρόπο που να αποκρύπτουν την εσωτερική λειτουργικότητά τους.

---

<sup>34</sup> What You See Is What You Get: Όρος που χρησιμοποιείται για να περιγράψει τις μεθόδους σύνταξης αντικειμένων τα οποία δείχνουν το τελικό αποτέλεσμα κατά τη διάρκεια της συγγραφής. Για παράδειγμα, στο Microsoft Word, η επιλογή Bold γίνεται ορατή απευθείας μετά την επιλογή της στον επεξεργαστή κειμένου και όχι αφού αποθηκευτεί το έγγραφο ή δημοσιευθεί σε κάποιο ειδικό πρόγραμμα προβολής.



**Σχήμα 15 - Παράδειγμα μιας απλής Web Form**

Μπορούν να τοποθετηθούν επάνω σε μια Web Form (σελίδα .aspx) και να προσθέσουν αυτόματα πολύπλοκες λειτουργίες, όπως εξελιγμένους πίνακες δεδομένων, μενού επιλογών κ.α. Τα Server Controls αποτελούν για την ASP.NET ό,τι και τα ActiveX Controls για την Visual Basic 6, με τη διαφορά ότι τα πρώτα εκτελούνται στον διακομιστή κατά την δυναμική δημιουργία της σελίδας, παράγοντας τον απαραίτητο HTML κώδικα που θα υλοποιήσει τελικά την ενσωματωμένη λειτουργικότητά τους στον browser του τελικού χρήστη.

```

Private Sub Page_Load(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles MyBase.Load
    'Ανάγνωση ενός XML εγγράφου από το δίσκο
    Dim cDataSet As New DataSet()
    cDataSet.ReadXml("c:\temp\data.xml")

    'Ορισμός του table που περιέχεται στο dataset
    'ως περιεχομένων του DataGrid
    DataGrid1.DataSource = cDataSet.Tables(0)
End Sub
  
```

**Παράδειγμα 7 - Κώδικας για την συμπλήρωση ενός πίνακα δεδομένων**

Ήδη κυκλοφορούν αμέτρητα Server Controls, είτε δωρεάν είτε αγοράζοντας κάποια άδεια χρήσης, τα οποία αναλαμβάνουν να προσθέσουν λειτουργικότητα σε μια ASP.NET εφαρμογή, με

ελάχιστη προσπάθεια από τον προγραμματιστή. Υπάρχει βέβαια και η δυνατότητα για δημιουργία νέων Server Controls, με χρήση της VB.NET, της C#, της C++ κλπ. Αυτά τα Server Controls, μπορεί να περικλείονται σε ένα αρχείο με επέκταση .ascx ή να ενσωματωθούν σε μια .NET assembly, δηλαδή σε ένα αρχείο .dll.

Στο Σχήμα 16 φαίνεται ένα Server Control που περιέχεται στο .NET Framework και αναλαμβάνει τη δημιουργία ενός πίνακα δεδομένων, χωρίς την συγγραφή κώδικα HTML από τον προγραμματιστή. Όπως κάθε Server Control, έτσι και αυτό περιέχει διάφορες ιδιότητες, όπως καθορισμός των χρωμάτων του φόντου, της γραμματοσειράς κ.α. και πάλι χωρίς την επέμβαση στον κώδικα HTML, παρά μόνο στο WYSIWYG περιβάλλον του Visual Studio .NET. Για την συμπλήρωση του πίνακα αυτού με δεδομένα, αρκεί να δημιουργηθεί ένα ADO.NET DataSet, όπως ακριβώς θα γινόταν για παράδειγμα σε μια Windows .NET εφαρμογή, και να οριστεί η ιδιότητα DataSource, όπως φαίνεται και στο Παράδειγμα 7. Σε αυτό, τα περιεχόμενα ενός DataSet φορτώνονται από ένα αρχείο XML, όπως και στο Παράδειγμα 5. Συγκεκριμένα, θεωρώντας ότι το XML περιέχει τα δεδομένα ενός πίνακα, αυτά φορτώνονται στο πρώτο<sup>35</sup> Table του Dataset – *cDataSet.Tables(0)* - και στη συνέχεια αυτό το Table χρησιμοποιείται ως πηγή δεδομένων του DataGrid – *DataGrid1.DataSource*. Η εκτέλεση του κώδικα γίνεται κατά το φόρτωμα της σελίδας (event Load της σελίδας).

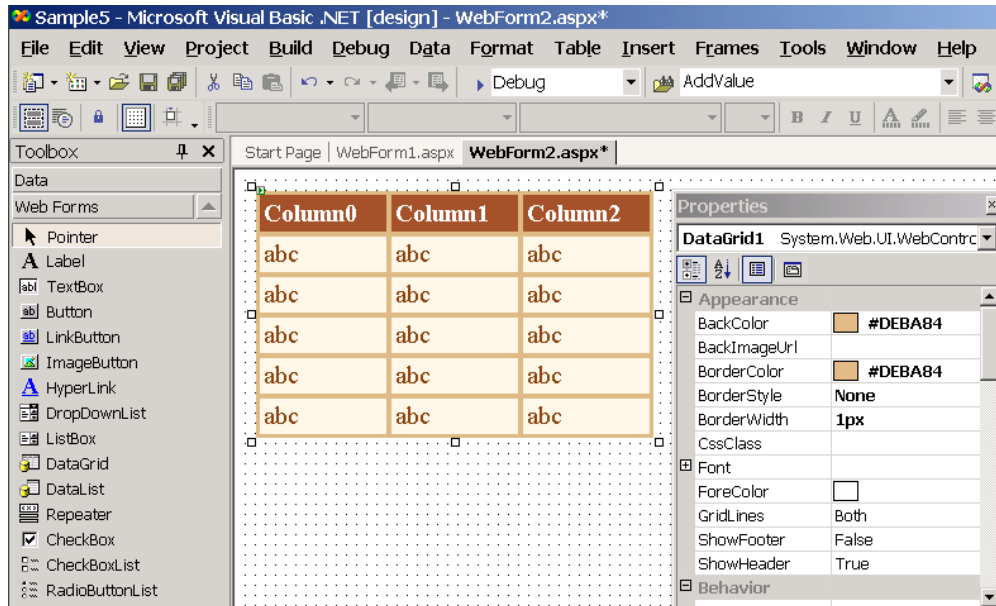
- **Server Caching**

Υπάρχει μια πολύ σημαντική διαφορά μεταξύ μιας Web εφαρμογής και μιας τυπικής Windows εφαρμογής. Αυτή είναι το γεγονός ότι, στην πρώτη περίπτωση, σχεδόν κάθε κλικ του τελικού χρήστη αντιστοιχεί στην επαναδημιουργία κάποιας html σελίδας από τον διακομιστή, με ότι αυτό μπορεί να συνεπάγεται ανάλογα με τα περιεχόμενά της. Λόγου χάρη, στο προηγούμενο παράδειγμα, κάθε

---

<sup>35</sup> Όπως αναφέρεται και στην παράγραφο 5.3, ένα DataSet μπορεί να περιέχει πολλούς πίνακες δεδομένων.

κλήση του χρήστη θα συνοδεύεται από άνοιγμα του αρχείου XML και επαναδημιουργία του πίνακα δεδομένων.



**Σχήμα 16 - Πίνακας Δεδομένων σε μορφή Server Control**

Η ASP.NET περιλαμβάνει την δυνατότητα Caching στοιχείων που επιλέγει ο προγραμματιστής, τα οποία δεν επαναδημιουργούνται σε κάθε κλήση. Πρόκειται για μια δυνατότητα που δεν υπήρχε στην προηγούμενη έκδοση της ASP. Αν για παράδειγμα είναι γνωστό εκ των προτέρων ότι τα περιεχόμενα κάποιου Server Control ή κάποιας ολοκληρωμένης σελίδας, δεν αλλάζουν παρά μόλις μια φορά την ημέρα, ο προγραμματιστής μπορεί να επιλέξει την αυτόματη προσωρινή αποθήκευση και καθημερινή ενημέρωση του συγκεκριμένου στοιχείου, έτσι ώστε αυτό να επαναχρησιμοποιείται έτοιμο σε κάθε κλήση οποιουδήποτε χρήστη. Ένα τέτοιο Server Control για παράδειγμα, μπορεί να εμφανίζει τα 5 προϊόντα με την μεγαλύτερη ζήτηση σε μια εφαρμογή e-shop. Προφανώς δεν θα επηρέαζε σημαντικά τη λειτουργικότητα της εφαρμογής, αν αυτή η λίστα ενημερωνόταν μια ή δυο φορές την ημέρα. Αντίθετα, θα βελτίωνε σημαντικά τη λειτουργία της, αφού κάτι τέτοιο θα μείωνε σημαντικά τις κλήσεις προς την βάση δεδομένων σε μόλις 1 ή 2 φορές την ημέρα, τουλάχιστον για τη συγκεκριμένη περιοχή της εφαρμογής.

### 8.3 Η εφαρμογή IBuySpy

Σε αυτή την παράγραφο θα γίνει η παρουσίαση μιας εφαρμογής ASP.NET, με σκοπό την καλύτερη κατανόηση των χαρακτηριστικών της, που αναφέρθηκαν και παραπάνω. Η εφαρμογή IBuySpy αποτελεί μια εφαρμογή επίδειξης της ASP.NET από την Microsoft. Πρόκειται για ένα Web Site που δίνει τη δυνατότητα αγοράς εξοπλισμού για κατασκόπους (!), όπως μυστικά όπλα, υλικά μεταμφίεσης κ.α.. Η πλοήγηση στην εφαρμογή γίνεται με τον τυπικό τρόπο που συνηθίζεται σε ένα e-shop, δηλαδή προβολή των διαθέσιμων προϊόντων σε κατηγορίες, τη χρήση ενός καλάθιού αγορών (shopping basket) στο οποίο ο χρήστης προσθέτει τα επιθυμητά προϊόντα και μια διαδικασία check-out κατά την οποία αποστέλλει ή ακυρώνει την παραγγελία του.

Για την καλύτερη κατανόηση της παρακάτω παρουσίασης από τον αναγνώστη, προτείνεται η εγκατάσταση της εφαρμογής σε έναν υπολογιστή. Εάν αυτό δεν είναι δυνατό, προτείνεται η αποσυμπίεση των περιεχομένων της εφαρμογής και προβολή κάποιων αρχείων σε οποιοδήποτε επεξεργαστή κειμένου.<sup>36</sup>

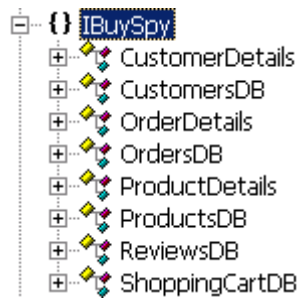
#### Σύνδεση με τη βάση δεδομένων – Business Objects

Η διαδικασία σύνδεσης με την βάση δεδομένων αναλαμβάνεται από μια σειρά από κλάσεις, οι οποίες δεν αποτελούν απαραίτητα τμήμα της ASP.NET εφαρμογής. Πολλές από τις εργασίες που εκτελούνται από αυτές τις κλάσεις θα μπορούσαν να χρησιμοποιηθούν και σε άλλες εφαρμογές. Η δομή των κλάσεων φαίνεται στο Σχήμα 17. Υπενθυμίζεται ότι μια κλάση στο .NET Framework δεν αποτελεί ένα αυτόνομο αρχείο, όπως ίσχυε στην περίπτωση της Visual Basic 6. Για παράδειγμα οι κλάσεις CustomerDetails και CustomersDB περιέχονται στο αρχείο CustomersDB.vb, όπου η επέκταση vb δηλώνει ότι ο περιεχόμενος κώδικας είναι σε Visual Basic.

---

<sup>36</sup> Ο κώδικας και όλα τα υπόλοιπα απαραίτητα στοιχεία (sql server database, documentation) για την εκτέλεση της εφαρμογής βρίσκονται στο συνοδευτικό CD-ROM αυτής της εργασίας, αλλά και στην επίσημη ιστοσελίδα της ASP.NET ([www.asp.net](http://www.asp.net)). Ο κώδικας είναι διαθέσιμος για VB και C#. Μια επίδειξη της εφαρμογής γίνεται και στη διεύθυνση <http://www.ibuyspystore.com>.

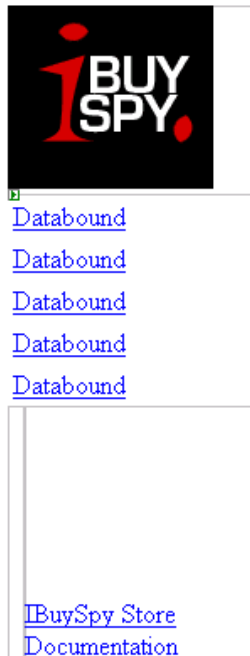
Σε γενικές γραμμές, οι κλάσεις αυτές κάνουν χρήση του ADO.NET (βλ. Κεφάλαιο 5), για τη λήψη και τροποποίηση δεδομένων σε μια βάση δεδομένων σε SQL Server.



Σχήμα 17 - Δομή κλάσεων IBuySpy

### Διάσπαση σε Server Controls

Ένα από τα πρώτα χαρακτηριστικά της εφαρμογής που διακρίνει κανείς, είναι η διάσπασή της σε server controls (αρχεία με επέκταση .ascx). Το καθένα από αυτά αναλαμβάνει την προβολή κάποιων συγκεκριμένων στοιχείων μέσα στις σελίδες της εφαρμογής. Για παράδειγμα, το server control `_popularItems.ascx` (το σύμβολο `_` στην αρχή του ονόματος αποτελεί απλά επιλογή των προγραμματιστών της εφαρμογής για όλα τα server



controls) αναλαμβάνει την παραγωγή του HTML κώδικα που θα εμφανίζει μια λίστα με τα πιο δημοφιλή προϊόντα. Αντίστοιχα, το server control `_menu.ascx` (εικόνα αριστερά) αναλαμβάνει την παραγωγή του HTML κώδικα που θα εμφανίσει το μενού επιλογών που περιέχεται στο αριστερό τμήμα όλων των σελίδων της εφαρμογής. Θα μπορούσε κανείς να χρησιμοποιήσει αυτά τα server controls χωρίς να γνωρίζει τον περιεχόμενο κώδικά τους, μια που είναι πιθανόν η ανάπτυξή τους να πραγματοποιήθηκε από τρίτους προγραμματιστές. Ενδεικτικά, φαίνεται παρακάτω ένα τμήμα του κώδικα του `_menu.ascx`, στο οποίο γίνεται η ανάθεση των περιεχομένων που θα έχει τελικά το μενού, σε ένα DataList server control

(`MyList.DataSource = products.GetProductCategories`). Ο παρακάτω κώδικας είναι άλλωστε όλος ο κώδικας που χρειάζεται να γράψει ο προγραμματιστής σε επίπεδο ASP.NET. Η υπόλοιπη προσπάθεια που

απαιτείται είναι σε επίπεδο σχεδιασμού της σελίδας (visual). Η συνάρτηση `GetProductCategories` απλά επιστρέφει ένα ADO.NET `DataTable` (βλ. παράγραφο 5.3) με τις κατηγορίες των προϊόντων, οι οποίες θέλουμε να αποτελούν τις επιλογές του μενού, όπως διαβάζονται από την βάση δεδομένων. Αυτή η συνάρτηση δεν απευθύνεται μόνο σε αυτή την εφαρμογή ASP.NET.

```
Private Sub Page_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

    ' Set the current selection of list
    Dim selectionId As String = Request.Params("selection")

    If Not selectionId Is Nothing Then
        MyList.SelectedIndex = CInt(selectionId)
    End If

    ' Obtain list of menu categories and databind to list control
    Dim products As IBuySpy.ProductsDB = New IBuySpy.ProductsDB()

    MyList.DataSource = products.GetProductCategories()
    MyList.DataBind()

End Sub
```

Το `DataList` Control που προαναφέρθηκε, είναι και αυτό από την πλευρά του ένα `Server Control`, το οποίο εκτελείται κατά την ακολουθία εκτέλεσης του `_menu.ascx`. Το εν λόγω control είναι ενσωματωμένο στις βιβλιοθήκες κλάσεων του `.NET Framework` και δεν δίνεται η δυνατότητα στον χρήστη να παρέμβει σε αυτό, ούτε να δει τον κώδικα που περιέχει.

Σε αντίθεση λοιπόν με την ASP, στην ASP.NET είναι πολύ πιθανόν, και αυτό είναι και η προτεινόμενη τακτική, να χρησιμοποιούνται `Server Controls` τα οποία θα δημιουργούν ένα σύνολο σελίδων στις οποίες δεν θα είναι πλήρως ορατός ούτε ο server κώδικας, αλλά ούτε και ο κώδικας HTML που μπορεί να περιέχουν. Στην ASP, εκτός εξαιρετικών περιπτώσεων, μια εφαρμογή περιείχε στον φάκελό της όλο τον κώδικα που θα μπορούσε να εκτελεστεί σε αυτή.

### Server Caching

Άλλη μια παρατήρηση που μπορεί να κάνει κανείς μελετώντας τον κώδικα της εφαρμογής, είναι η γραμμές `@OutputCache` στην αρχή των αρχείων `.ascx`, δηλαδή των `Server Controls`. Η γραμμή αυτή δηλώνει κάθε πότε θα

πρέπει να εκτελείται το κάθε Server Control, παράγοντας τον HTML κώδικα ο οποίος θα αποσταλεί στον web browser του επισκέπτη της εφαρμογής.

Μια πλήρης μορφή της γραμμής αυτής είναι η ακόλουθη:

```
<%@ OutputCache Duration="#ofseconds" Location="Any | Client |  
Downstream | Server | None" VaryByControl="controlname"  
VaryByCustom="browser | customstring" VaryByHeader="headers"  
VaryByParam="parametername" %>
```

Οι παράμετρος *duration* καθορίζει κάθε πόσα δευτερόλεπτα θα εκτελείται ο κώδικας του Server Control, ενώ η παράμετρος *Location* καθορίζει την τοποθεσία που θα γίνεται η προσωρινή αποθήκευση των αποτελεσμάτων της εκτέλεσης. Στην περίπτωση που δεν έχει παρέλθει ο χρόνος *duration*, το Server Control δεν θα εκτελείται, αλλά θα χρησιμοποιούνται απευθείας τα προσωρινά αποθηκευμένα αποτελέσματα από την τοποθεσία που αποθηκεύτηκαν κατά την προηγούμενη εκτέλεση.

Οι υπόλοιπες παράμετροι προσδιορίζουν με περισσότερες λεπτομέρειες πότε πρέπει να γίνεται η εκτέλεση του Server Control, ακόμα και αν δεν έχει παρέλθει ο απαιτούμενος χρόνος. Μπορούμε με αυτό τον τρόπο να παραμετροποιήσουμε περισσότερο τη στιγμή εκτέλεσης. Για παράδειγμα, σε περίπτωση που τα αποτελέσματα της εκτέλεσης του Server Control διαφέρουν ανάλογα με τον Web Browser του χρήστη (π.χ. Internet Explorer 5, Internet Explorer 6, Mozilla 2 κλπ.), η προσωρινή αποθήκευση θα πρέπει να γίνεται για κάθε browser ξεχωριστά. Κάτι τέτοιο επιτυγχάνεται θέτοντας την τιμή "browser" στην παράμετρο *VaryByCustom*.

Περισσότερες πληροφορίες για τη χρήση της γραμμής @OutputCache, βλ. εγχειρίδιο .NET Framework.

## Web Services

Η εφαρμογή IBuySpry περιλαμβάνει δυο συναρτήσεις σε μορφή Web Service, τις οποίες μπορεί να χρησιμοποιήσει κάποια απομακρυσμένη εφαρμογή. Δίνεται με αυτό τον τρόπο η δυνατότητα χρήσης της εφαρμογής χωρίς την πρόσβαση στο Web περιβάλλον της. Οι δυο συναρτήσεις περιέχονται στο αρχείο InstantOrder.aspx, το οποίο αποτελεί και την μοναδική προϋπόθεση για την λειτουργία του Web Service. Όπως αναφέρθηκε και στην παράγραφο 6.3, όλα τα πρωτόκολλα των Web

Services ικανοποιούνται δυναμικά από το .NET Framework. Για παράδειγμα, ο χρήστης του Web Service που θέλει να πάρει ένα αντίγραφο του εγγράφου WSDL που χαρακτηρίζει την υπηρεσία, αρκεί να γράψει στον web browser του τη διεύθυνση:

`http://www.myserver.gr/StoreVBVS/InstantOrder.asmx?WSDL,`

σε περίπτωση βέβαια όπου η διεύθυνση του Web Site είναι η `http://www.myserver.gr/StoreVBVS`.

Οι δυο συναρτήσεις που περιέχονται εν τέλει στην εφαρμογή, είναι η συνάρτηση `OrderItem`, με τη βοήθεια της οποίας μπορεί κανείς να αποστείλει μια παραγγελία και η συνάρτηση `CheckStatus` η οποία επιστρέφει την κατάσταση μιας υπάρχουσας παραγγελίας.

### **To αρχείο Web.Config**

Πρόκειται για ένα αρχείο XML, το οποίο περιέχει διάφορες ρυθμίσεις που αφορούν στην εκτέλεση της εφαρμογής. Το αρχείο αυτό, αν και βρίσκεται στον φάκελο της εφαρμογής μαζί με τα υπόλοιπα αρχεία της, δεν είναι ορατό από τον τελικό χρήστη, όπως βέβαια και άλλοι τύποι αρχείων. Στην περίπτωση που ο χρήστης πληκτρολογήσει τη διεύθυνση `http://localhost/StoreVBVS/web.config`, τότε θα λάβει το μήνυμα 'This type of page is not served'.

Μερικές από τις ρυθμίσεις που μπορεί περιέχονται σε αυτό το αρχείο είναι οι παρακάτω:

- Authentication

```
<authentication mode="Forms">
  <forms name="IBuySpyStoreAuth" loginUrl="login.aspx" protection="All" path="/" />
</authentication>
```

Καθορίζει τον τρόπο με τον οποίο θα γίνεται η πιστοποίηση των χρηστών σε περιοχές της εφαρμογής στις οποίες δεν επιτρέπεται η ανώνυμη πρόσβαση. Η μέθοδος πιστοποίησης που έχει επιλεγεί για την εφαρμογή `IBuySpy` είναι η μέθοδος `Forms Authentication`, κατά την οποία η πιστοποίηση γίνεται από την ίδια την εφαρμογή. Σε αυτή την περίπτωση η εφαρμογή περιέχει μια σελίδα (στην περίπτωση μας την `login.aspx`) η οποία αναλαμβάνει να ζητήσει ένα e-mail και έναν κωδικό πρόσβασης. Στη συνέχεια η ίδια σελίδα ελέγχει την εγκυρότητα των στοιχείων που δίνονται σε σχέση με τα στοιχεία κάποιας βάσης δεδομένων. Αν τα στοιχεία είναι έγκυρα, αποθηκεύει στον

υπολογιστή του χρήστη ένα cookie, το οποίο πιστοποιεί ότι ο χρήστης δεν είναι πλέον ανώνυμος. Προφανώς κάποια άλλη σελίδα θα πρέπει αναλαμβάνει την εγγραφή νέων χρηστών στην εφαρμογή, η οποία στην περίπτωση της IBuySpy είναι η σελίδα register.aspx.

Άλλες μέθοδοι πιστοποίησης είναι η μέθοδος Windows και η μέθοδος Passport Authentication οι οποίες χρησιμοποιούν την ασφάλεια των Windows NT και το σύστημα Microsoft Passport για την αναγνώριση των επισκεπτών του Web Site. Περισσότερες πληροφορίες γι αυτές τις μεθόδους βλ. 'ASP.NET Authentication' στο εγχειρίδιο του .NET Framework.

- CustomErrors

```
<!-- enable custom errors for the application -->  
<customErrors mode="RemoteOnly" defaultRedirect="ErrorPage.aspx" />
```

Καθορίζει τον τρόπο αντιμετώπισης μη αναμενόμενων σφαλμάτων της εφαρμογής, δηλαδή σφαλμάτων που δεν προέβλεψε ο προγραμματιστής. Στην περίπτωση του IBuySpy, η σελίδα ErrorPage.aspx αναλαμβάνει να εμφανίζει ένα φιλικό μήνυμα στο χρήστη. Στο αρχείο web.config ορίζεται επίσης ότι αυτή η σελίδα θα χρησιμοποιείται μόνο για επισκέπτες της εφαρμογής από άλλους υπολογιστές. Με αυτό τον τρόπο ο προγραμματιστής ή διαχειριστής της εφαρμογής, θα μπορεί να λαμβάνει ολοκληρωμένες τις περιγραφές των σφαλμάτων, την ώρα που οι τελικοί χρήστες θα βλέπουν μια πιο φιλική μορφή των προβλημάτων που μπορεί να προκύψουν.

- Authorization

Η διαδικασία authorization καθορίζει σε ποιες σελίδες θα έχει πρόσβαση ένας ανώνυμος χρήστης, καθώς στις υπόλοιπες θα πρέπει να ζητηθεί η πιστοποίησή του.

Για παράδειγμα, οι παρακάτω γραμμές του αρχείου web.config, ορίζουν ότι η σελίδα Checkout.aspx, δηλαδή η σελίδα κατά την οποία ολοκληρώνεται μια παραγγελία, δεν θα πρέπει να είναι προσβάσιμη από ανώνυμους χρήστες. Αυτό δηλώνεται από την γραμμή <deny users="?">

```
<location path="Checkout.aspx">
  <system.web>
    <authorization>
      <deny users="?" />
    </authorization>
  </system.web>
</location>
```

- AppSettings

Η περιοχή αυτή προσφέρεται για την προσθήκη διάφορων ρυθμίσεων από τον προγραμματιστή, οι οποίες πρέπει να είναι προσβάσιμες χωρίς να χρειάζεται κανείς να ανατρέξει στον κώδικα της εφαρμογής. Μπορούν να οριστούν παράμετροι σύνδεσης με κάποια βάση δεδομένων, τοποθεσίες αρχείων στον δίσκο του διακομιστή κ.α.

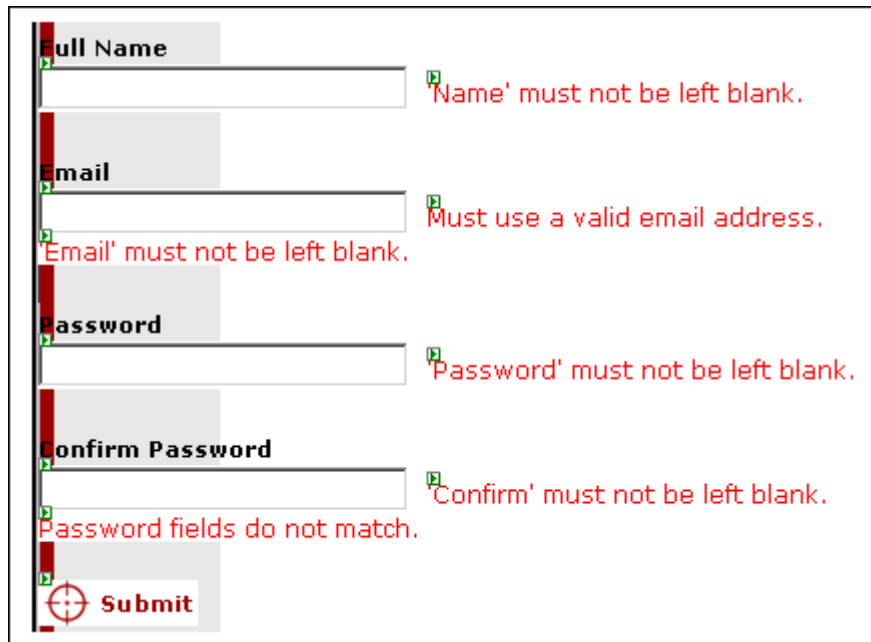
Περισσότερα για τα αρχεία Web.Config, βλ. 'Application Root Directory Configuration File' στο εγχειρίδιο του .NET Framework.

### User Interface

Η σχεδίαση του περιβάλλοντος διάδρασης με τον χρήστη στην ASP.NET φέρει μερικά κοινά στοιχεία με την ASP, ως προς το γεγονός της χρήσης HTML Controls, όπως πλαίσια κειμένου (input text), πλήκτρα λειτουργιών (input command) κ.α. Εκτός από αυτά, στην ASP.NET, όπως προαναφέρθηκε αρκετές φορές, είναι δυνατή η χρήση των Server Controls, τα οποία μπορούν να εκτελέσουν τις ίδιες ή περισσότερο σύνθετες εργασίες. Παρουσιάστηκαν διάφορες περιπτώσεις server controls, όπως αυτά της εφαρμογή IBuySpy αλλά και άλλα ενσωματωμένα στο .NET Framework. Μια κατηγορία Server Controls, ενσωματωμένων στο .NET Framework, που χρησιμοποιούνται και στην εφαρμογή IBuySpy, είναι τα Validation Controls (στοιχεία ελέγχου εγκυρότητας).

Στο Σχήμα 18 φαίνεται τμήμα της σελίδας register.aspx σε κατάσταση σχεδίασης. Δίπλα από κάθε Server Control, στο οποίο αναμένονται τα στοιχεία του χρήστη, έχει τοποθετηθεί ένα Validation Control το οποίο αναλαμβάνει τον έλεγχο της τιμής που θα δοθεί. Σε κάθε ένα από αυτά ορίζεται ο αντίστοιχο Server Control του οποίου την τιμή αναλαμβάνει να ελέγξει, το μήνυμα το οποίο θα εμφανιστεί σε περίπτωση λάθους, καθώς και το είδος του ελέγχου που θα γίνει. Είναι δυνατός ο έλεγχος αριθμητικών τιμών, έτσι ώστε να είναι μεταξύ συγκεκριμένων ορίων, η εξασφάλιση πληκτρολόγησης οποιασδήποτε τιμής (not null), ο έλεγχος εγκυρότητας

ημερομηνιών καθώς και ο έλεγχος πολυπλοκότερων προϋποθέσεων, όπως κάποια έγκυρη διεύθυνση e-mail κ.α. Ο έλεγχος των τιμών γίνεται μετά το πάτημα του πλήκτρου 'Submit' από τον τελικό χρήστη.



The image shows a web form with four input fields and a 'Submit' button. Each field has a red error message next to it. The fields and their messages are:

- Full Name**: 'Name' must not be left blank.
- Email**: 'Email' must not be left blank. Must use a valid email address.
- Password**: 'Password' must not be left blank.
- Confirm Password**: 'Confirm' must not be left blank. Password fields do not match.

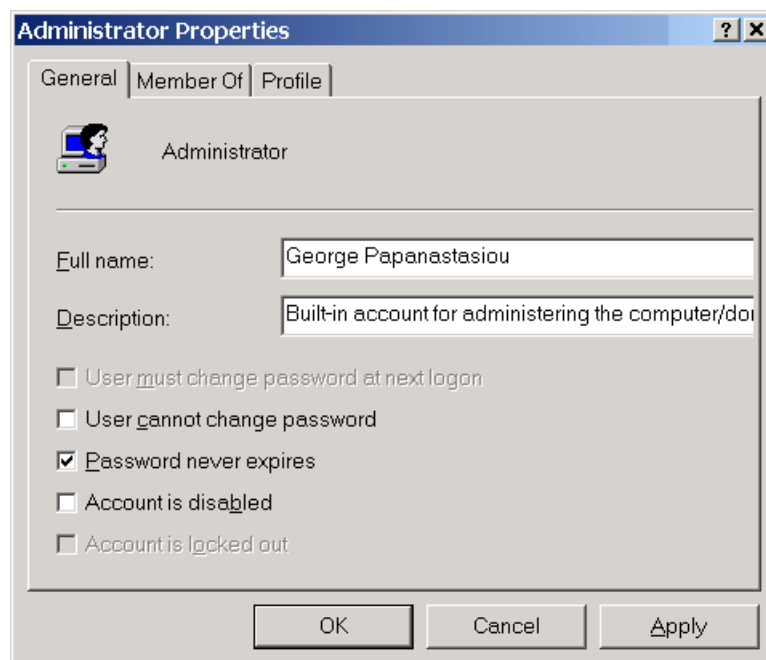
The 'Submit' button is located at the bottom left of the form area.

**Σχήμα 18 - Validation Controls**

Ανάλογοι έλεγχοι στην περίπτωση της ASP, απαιτούσαν σημαντική ποσότητα κώδικα, συνήθως σε Javascript. Αντίθετα, στην ASP.NET, τα Validation Controls προσφέρουν πολύπλοκους ελέγχους σε πεδία τιμών, χωρίς την σύνταξη κώδικα, μια που αυτός δημιουργείται δυναμικά κατά την εκτέλεση των σελίδων, ανάλογα με τους ελέγχους που επιλέχθηκε να γίνουν.

## 9. Active Directory

Το Active Directory αποτελεί το υπόβαθρο για το λογισμικό των δικτύων που βασίζονται στους Windows 2000 και Windows .NET Server 2003 domain controllers. Καταγράφει και παρέχει μια ιεραρχική δομή<sup>37</sup> των πόρων που περιλαμβάνει το δίκτυο μιας επιχείρησης, σε επίπεδο χρηστών, υπολογιστών κ.α. Με τη βοήθεια του Active Directory μπορεί κανείς να εντοπίσει και να εκμεταλλευτεί όλη την καταχωρημένη πληροφορία για αυτούς τους πόρους. Το Active Directory, παρέχει επίσης στους διαχειριστές του δικτύου τη δυνατότητα να αυτοματοποιήσουν εργασίες που εκτελούν συχνά.



**Σχήμα 19 - Παράθυρο επεξεργασίας χρήσης από το γραφικό περιβάλλον των Windows 2000**

Το ενδιαφέρον όμως για την ανάπτυξη ενός προγράμματος και το χτίσιμο ενός πληροφοριακού συστήματος που εκμεταλλεύεται το Active Directory, είναι το γεγονός ότι αμέσως μετά την εγκατάστασή του, μπορεί ουσιαστικά να αντιληφθεί αρκετά στοιχεία του περιβάλλοντος στο οποίο πρόκειται να λειτουργήσει. Για παράδειγμα, μια από τις πιο γνωστές λειτουργίες ενός συστήματος είναι η απόδοση ενός κωδικού ονόματος και ενός μυστικού

---

<sup>37</sup> Ιεραρχική είναι τόσο η δομή όσο και η μέθοδος αποθήκευσης των πληροφοριών του active directory

κωδικού πρόσβασης μέσω των οποίων θα γίνεται η πιστοποίηση των χρηστών του. Κάτι τέτοιο θα απαιτούσε τη δημιουργία μιας λίστας με τους χρήστες του συστήματος, την οργάνωση μιας διαδικασίας όπου οι χρήστες θα πρέπει να επιλέξουν ένα κωδικό όνομα και να παραλάβουν έναν αντίστοιχο κωδικό πρόσβασης, την παροχή της δυνατότητας να αλλάζουν αυτόν τον κωδικό κ.α.

Κάτι ανάλογο θα πρέπει να γίνει για όλα τα συστήματα που εγκαθίστανται στην επιχείρηση. Αυτή η διαδικασία, απαιτεί σημαντικό χρόνο για να ολοκληρωθεί, ενώ το ίδιο ή περισσότερο χρονοβόρες είναι και οι διαδικασίες συγχρονισμού, μεταξύ των συστημάτων, του συνόλου των πληροφοριών, σε περίπτωση αλλαγών του περιβάλλοντος της επιχείρησης. Αποτέλεσμα αυτών των διαδικασιών είναι η αύξηση του λειτουργικού κόστους των επιμέρους συστημάτων που χρησιμοποιούνται. Το μοντέλο που περιγράφηκε είναι μάλλον γνωστό και αναπαριστάται στο Σχήμα 20.



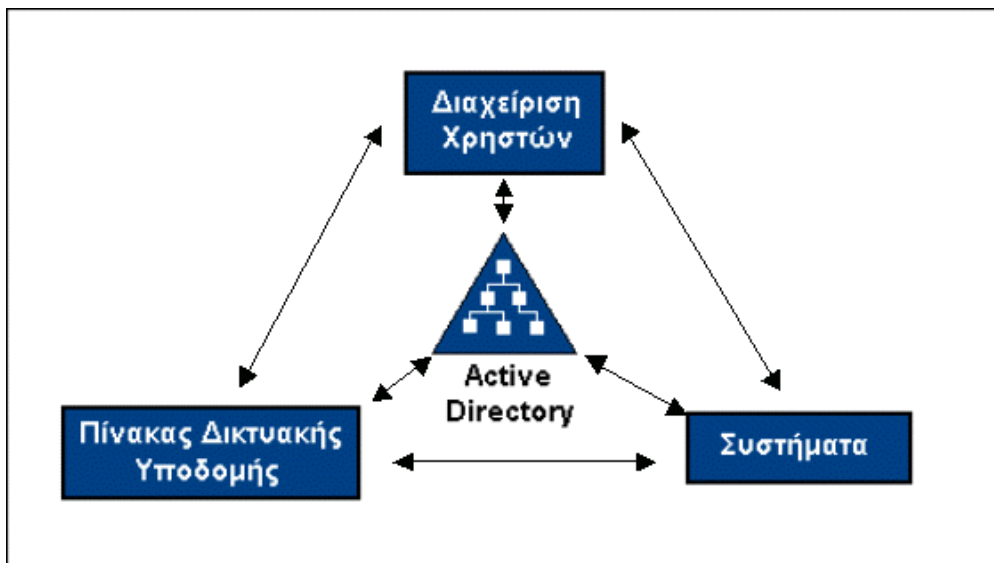
**Σχήμα 20 - Μοντέλο Λειτουργίας Επιχείρησης χωρίς Active Directory**

Επιστρέφοντας στο προηγούμενο παράδειγμα, αυτό που χρειάζεται ένα σύστημα πριν επιτρέψει ένα χρήστη να το χρησιμοποιήσει, είναι να αντιληφθεί την ταυτότητά του, έτσι ώστε στη συνέχεια να του επιτρέψει να εκτελέσει κάποιες εργασίες σε δεδομένα στα οποία έχει πρόσβαση ή να μην του επιτρέψει καθόλου οποιαδήποτε χρήση. Κάτι τέτοιο, όμως, στο περιβάλλον ενός Windows Domain<sup>38</sup>, μπορεί να γίνει από το εν λόγω

---

<sup>38</sup> Λογικό σύνολο υπολογιστών, συνήθως ενός τοπικού δικτύου, το οποίο ελέγχεται από ένα κεντρικό υπολογιστή (domain controller)

σύστημα χωρίς να ζητήσει κάποια πληροφορία από τον τελικό χρήστη, επομένως χωρίς να χρειάζεται ο χρήστης να θυμάται ένα κωδικό πρόσβασης, άρα και χωρίς να χρειάζεται ένας διαχειριστής του δικτύου να τον παρέχει. Το Active Directory προσφέρει στο σύστημα τα στοιχεία του χρήστη (βλ. Σχήμα 19), τους διαθέσιμους εκτυπωτές στο χώρο του κ.α. καθιστώντας το σύστημα άμεσα λειτουργικό από την πρώτη στιγμή της λειτουργίας του χωρίς κάποια ιδιαίτερη παραμετροποίηση από τους διαχειριστές του, τουλάχιστον στο επίπεδο αυτής της πληροφορίας. Αυτό το μοντέλο λειτουργίας φαίνεται στο Σχήμα 21.



**Σχήμα 21 – Μοντέλο Λειτουργίας Επιχείρησης με χρήση Active Directory**

Τρεις είναι οι τρόποι με τους οποίους μπορεί κανείς να έχει πρόσβαση στην πληροφορία του Active Directory:

- μέσω του γραφικού περιβάλλοντος διαχείρισης των Windows 2000 και .NET Server 2003<sup>39</sup> για τη διαχείριση των πληροφοριών από τους διαχειριστές ενός δικτύου,

---

<sup>39</sup> Η διαχείριση αυτή γίνεται μέσω των Snap-In του Management Console των Windows, ένα σύστημα που αναλαμβάνει να παρουσιάσει σχεδόν όλες τις επιλογές των Windows σε ένα ενιαίο γραφικό περιβάλλον, γεγονός που καθιστά τη διαχείριση των εκατοντάδων ρυθμίσεων των Windows αρκετά φιλική. Στο περιβάλλον αυτό στηρίζεται τόσο η διαχείριση του Active Directory, όσο και άλλων συστημάτων, όπως ο Microsoft SQL Server, ο IIS, το Computer Management των Windows 2000 PRO και Windows XP κ.α.

- μέσω του ADSI (Active Directory Service Interface) το οποίο παρέχει δυνατότητα πρόσβασης σε γλώσσες προγραμματισμού όπως η C++, η Visual Basic 6 ή ακόμα και στις VB-Script και J-Script, χρησιμοποιώντας την τεχνολογία COM και τέλος,
- μέσω του System.DirectoryServices namespace του .NET Framework, ο τρόπος χρήσης του οποίου περιγράφεται στην επόμενη παράγραφο.

Πολύ σημαντικό είναι το γεγονός ότι τα δυο τελευταία μπορούν να χρησιμοποιηθούν για την λήψη αντίστοιχης πληροφορίας από πηγές εκτός του Microsoft Active Directory. Συγκεκριμένα, το πακέτο εργαλείων του ADSI παρέχει τη δυνατότητα για λήψη αυτής της πληροφορίας και από τα συστήματα Novell NDS, Novell 3x και LDAP. Αυτό ουσιαστικά σημαίνει ότι οι εφαρμογές που θα σχεδιαστούν έτσι ώστε να αντιλαμβάνονται το περιβάλλον των Windows NT στο οποίο εγκαθίστανται, θα μπορούν να διαβάσουν αντίστοιχη πληροφορία ακόμα και αν σε αυτό το περιβάλλον δεν υπάρχει ένα ολοκληρωμένο NT domain στηριζόμενο σε έναν διακομιστή Windows, αλλά ένα από τα προηγούμενα περιβάλλοντα. Πιο συγκεκριμένα, μέσω του πρωτοκόλλου LDAP και με τη βοήθεια του ADSI ή του System.DirectoryServices του .NET Framework, μπορούμε να έχουμε πρόσβαση στους καταλόγους των παρακάτω συστημάτων:

- Windows NT
- Netscape Directory Server 1.0
- Microsoft Exchange 5.0
- Microsoft Internet Information Server (IIS)
- Microsoft Commercial Internet System
- University of Michigan SLAPD server
- Novell's LDAP-enabled NDS

## **9.1 Χαρακτηριστικά και πλεονεκτήματα του Microsoft Active Directory**

Σε αυτή την παράγραφο απαριθμούνται χαρακτηριστικά και πλεονεκτήματα που απορρέουν από την εφαρμογή και χρήση του Microsoft Active Directory

στα πλαίσια μιας εταιρείας ή οργανισμού<sup>40</sup>. Μερικά από αυτά αναφέρθηκαν εκτενέστερα στην προηγούμενη παράγραφο.

- Ενοποιημένη διαχείριση, από λιγότερα άτομα που έχουν τον έλεγχο καταλόγων υπολογιστών από ένα ή περισσότερα κεντρικά σημεία, ανάλογα με το μέγεθος του Active Directory.
- Ομαδοποίηση χρηστών (αλλά και ομάδων χρηστών σε νέες ομάδες<sup>41</sup>) παραμετροποιώντας καλύτερη την πρόσβασή τους στους πόρους του Directory.
- Δημιουργία διάφορων τύπων αντικειμένων εκτός από τα προκαθορισμένα (χρήστες, ομάδες χρηστών, υπολογιστές, domains κ.α.)
- Δημιουργία σχέσεων εμπιστοσύνης (trust<sup>42</sup>) μεταξύ domain, με μεταβατικό τρόπο. Π.χ. αν ένα domain A εμπιστεύεται ένα domain B και το τελευταίο εμπιστεύεται ένα domain Γ, τότε το A θα εμπιστεύεται και το Γ.
- Ενσωμάτωση του πρωτοκόλλου ασφάλειας Kerberos.
- Αντιγραφή και εγκατάσταση εφαρμογών σε υπολογιστές από τους χρήστες, ανεξάρτητα από τον υπολογιστή στον οποίο βρίσκονται<sup>43</sup>.
- Δυνατότητα μεταφοράς του Active Directory σε άλλο υπολογιστή-server, χωρίς να απαιτείται επανεγκατάσταση του τελευταίου.

Μερικές ποιο συγκεκριμένα χαρακτηριστικά του Active Directory είναι τα παρακάτω:

- Δυνατότητα πρόσβασης, όπως προαναφέρθηκε, με εύκολα στη χρήση APIs<sup>44</sup> και με διάφορες γλώσσες προγραμματισμού (VB, .NET Framework, VBScript/ Jscript, Perl κ.α.)

---

<sup>40</sup> Ken Spencer - Installing and Configuring Active Directory, 2000, Prentice Hall PTR

<sup>41</sup> η προσθήκη ομάδων χρηστών σε νέες ομάδες δεν ήταν δυνατή σε έναν Windows Server πριν το Active Directory.

<sup>42</sup> Τα trusts μεταξύ domain καθορίζουν αν οι χρήστες ενός domain θα έχουν πρόσβαση στο άλλο domain ή/ και στο αντίθετο.

<sup>43</sup> Active Directory's Microsoft Installer

- Διαλειτουργικότητα με περιβάλλοντα NetWare
- Υποστήριξη πολλών μορφών ονοματολογίας όπως RFC822 (π.χ. username@domain.com), RFC1779 (π.χ. CN=username, O=domain, C=gr) και άλλες παραλλαγές
- Ενσωμάτωση DNS (Domain Name Server)

## 9.2 Active Directory στο .NET Framework – System.DirectoryServices

Όπως αναφέρθηκε και προηγουμένως, η πρόσβαση στις πληροφορίες ενός Active Directory μέσω του .NET Framework, γίνεται με τη χρήση του System.DirectoryServices namespace, προσθέτοντας την αντίστοιχη παραπομπή σε ένα VB, C# ή άλλο .NET project.

```

Dim objDirEnt As DirectoryEntry
objDirEnt = new _
    DirectoryEntry("WinNT://domainname/computername/username")
Console.WriteLine("Name           = " & objDirEnt.Name)
Console.WriteLine("Path           = " & objDirEnt.Path)
Console.WriteLine("ClassName = " & objDirEnt.SchemaClassName)
Console.WriteLine("")
Console.WriteLine("Properties:")

Dim tab As string = "    "
Dim Key As String
Dim objValue as object

For Each Key In objDirEnt.Properties.PropertyNames
    Console.Write(tab & Key & " = ")
    Console.WriteLine("")
    For Each objValue In objDirEnt.Properties(Key)
        Console.WriteLine(tab & tab & objValue.ToString())
    next objValue
Next

```

### Παράδειγμα 8 - Λήψη πληροφοριών του ActiveDirectory

Στο Παράδειγμα 8 γίνεται χρήση της κλάσης DirectoryEntry για τη λήψη των ιδιοτήτων ενός αντικειμένου από ένα ActiveDirectory. Στην τρίτη γραμμή αυτού του παραδείγματος διακρίνεται η διαδρομή

---

<sup>44</sup> Application Programming Interface: οι βιβλιοθήκες που προσφέρει ένα σύστημα για την πρόσβαση σε αυτό μέσω γλωσσών προγραμματισμού.

"WinNT://domainname/computername/username", μέσω της οποίας γίνεται η λήψη ενός αντικειμένου-χρήστη ενός Windows Domain. Η εκτέλεση του παραπάνω παραδείγματος θα γράψει στην κονσόλα των Windows (μέσω της εντολής Console.WriteLine) της ιδιότητες ενός χρήστη, όπως το πλήρες όνομά του κ.α.

Άλλα παραδείγματα διαδρομών/paths που μπορούν να χρησιμοποιηθούν είναι τα παρακάτω, ανάλογα με το Directory το οποίο εξετάζεται:

- WinNT
  - Ομάδα χρηστών υπολογιστή: "WinNT://domain/computer/group" ή αν πρόκειται για τοπικό υπολογιστή: "WinNT://computer/group".
  - Χρήστης υπολογιστή: "WinNT://domain/computer/user" ή αν πρόκειται για τοπικό υπολογιστή: "WinNT://computer/user".
  - Service υπολογιστή: "WinNT://domain/computer/service" ή αν πρόκειται για τοπικό υπολογιστή: "WinNT://computer/service".
- LDAP
  - Ομάδα χρηστών ενός domain: "LDAP://CN=group-name, CN=Users, DC=domain-controller1, DC=domain-controller2,...".
  - Χρήστης ενός domain: "LDAP://CN=full-user-name, CN=Users, DC=domain-controller1, DC=domain-controller2,...".
  - Υπολογιστές ενός domain: "LDAP://CN= computer-name, CN=Computers, DC=domain-controller1, DC=domain-controller2,...".
- IIS
  - Web directory του IIS:, "IIS://LocalHost/W3SVC/1/ROOT/web-directory-name".

## 10. Εφαρμογή στο χώρο του e-government

Για την ολοκλήρωση της μελέτης της πλατφόρμας του .NET και την καλύτερη κατανόηση των τεχνολογιών που αυτό φέρει, παρουσιάζεται παρακάτω ο σχεδιασμός και ανάπτυξη ενός πραγματικού συστήματος. Το σύστημα αυτό έχει ως σκοπό του την αυτοματοποίηση της ανταλλαγής πιστοποιητικών σε ηλεκτρονική μορφή, μεταξύ υπηρεσιών και οργανισμών του δημόσιου ή ιδιωτικού τομέα<sup>45</sup>.

Είναι γνωστή σε όλους η διαδικασία υποβολής δικαιολογητικών για την διεκπεραίωση μιας διαδικασίας κυρίως σε δημόσιους φορείς, αλλά και στον ιδιωτικό τομέα. Μερικά μόνο από αυτά τα δικαιολογητικά μπορεί να είναι:

- επικυρωμένο αντίγραφο της αστυνομικής ταυτότητας,
- αντίγραφα πτυχίων, αναλυτικές βαθμολογίες,
- πιστοποιητικά φορολογικής ή ασφαλιστικής ενημερότητας,
- εκκαθαριστικά σημειώματα της εφορίας,
- βεβαιώσεις προϋπηρεσίας κ.α.

Το σύστημα CertServer που προτείνεται παρακάτω, προσπαθεί να αυτοματοποιήσει τη διαδικασία έκδοσης πιστοποιητικών και την ανταλλαγή τους μεταξύ των ενδιαφερόμενων υπηρεσιών, χωρίς την παρέμβαση του ενδιαφερόμενου πολίτη. Ένα παράδειγμα θα ήταν το πληροφοριακό σύστημα μιας νομαρχίας να λαμβάνει αυτόματα ένα πιστοποιητικό φορολογικής ενημερότητας του πολίτη με τον οποίο συναλλάσσεται, από το αντίστοιχο σύστημα του υπουργείου οικονομικών.

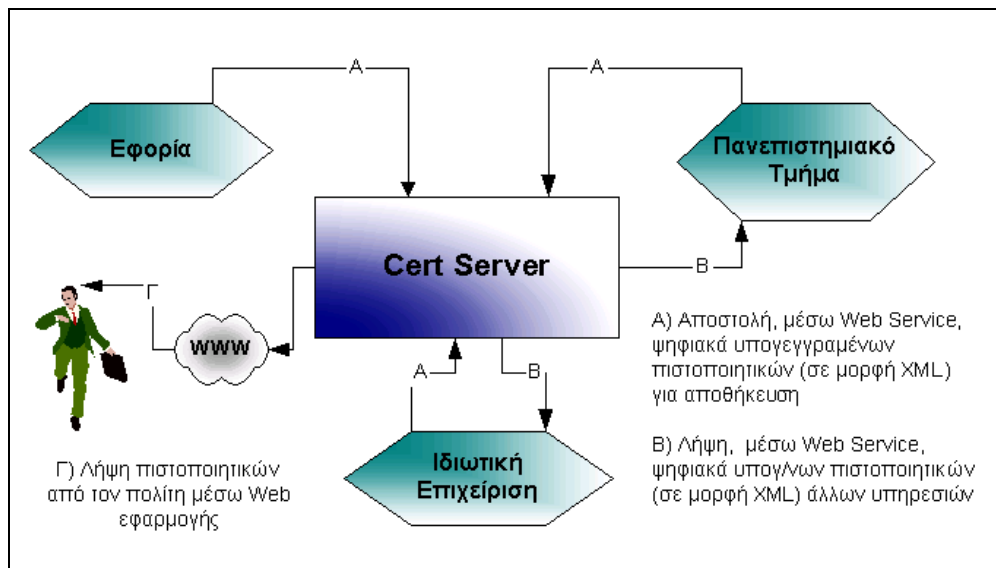
Για την υλοποίηση ενός τέτοιου συστήματος είναι απαραίτητη η χρήση των παρακάτω τεχνολογιών:

- Web Services, ως το πρωτόκολλο για την ανταλλαγή των πιστοποιητικών (βλ. κεφάλαιο 6),

---

<sup>45</sup> Όλος ο κώδικας του συστήματος CertServer, καθώς και συνοδευτικά αρχεία, όπως script για τη δημιουργία της βάσης δεδομένων, έγγραφα XML Schema, βρίσκονται στο CD-ROM που συνοδεύει αυτή την εργασία. Για την προβολή του κώδικα απαιτείται η ύπαρξη του Visual Studio .NET 2002.

- XML και XML Schema για την αναπαράσταση και την περιγραφή της πληροφορίας που φέρει το κάθε πιστοποιητικό (βλ. κεφάλαιο 7),
- SignedXML (W3C Recommendation) για την υλοποίηση ψηφιακών υπογραφών που θα πιστοποιούν την αυθεντικότητα των πιστοποιητικών,
- Windows .NET Forms, για τη δημιουργία εφαρμογής διαχείρισης των πιστοποιητικών,
- ASP.NET Forms, για τη δημιουργία WEB εφαρμογής μέσω της οποίας πολίτες και υπηρεσίες θα μπορούν να έχουν οπτική πρόσβαση στα πιστοποιητικά (βλ. κεφάλαιο 8),
- SQL Server, ADO.NET για την διαχείριση των δεδομένων (βλ. κεφάλαιο 5).



**Σχήμα 22 - Σύνδεση ενδιαφερόμενων με τον CertServer**

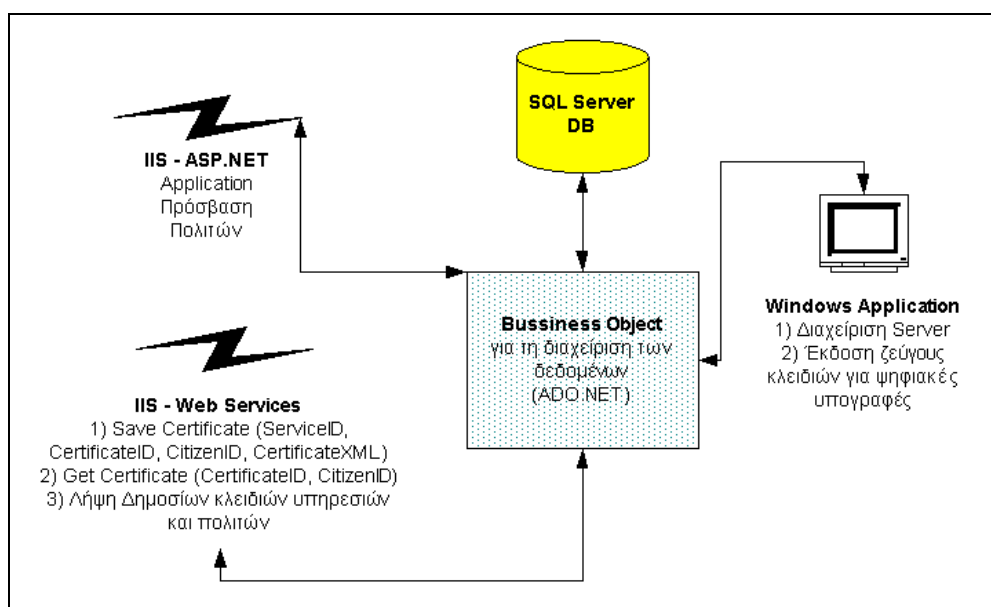
## 10.1 Ανάλυση του συστήματος CertServer

Ένα γενικευμένο διάγραμμα επικοινωνίας των ενδιαφερόμενων υπηρεσιών και πολιτών με το σύστημα CertServer φαίνεται στο Σχήμα 22. Σύμφωνα με αυτό, οι ενδιαφερόμενες υπηρεσίες στέλνουν τα πιστοποιητικά που εκδίδουν στον CertServer, κατάλληλα διαμορφωμένα σε έγγραφο XML (ένα τέτοιο έγγραφο φαίνεται στο 1<sup>ο</sup> μέρος, στο Σχήμα 25) που περιγράφονται από αντίστοιχα XML Schema (Σχήμα 26). Η αποστολή γίνεται με χρήση Web Services που βρίσκονται στον CertServer. Οι ενδιαφερόμενες υπηρεσίες

μπορούν επίσης να λαμβάνουν πιστοποιητικά που έχουν αποστείλει στον CertServer άλλες υπηρεσίες.

Στο Σχήμα 23 φαίνεται η εσωτερική δομή του CertServer. Αυτός περιλαμβάνει τα παρακάτω στοιχεία:

- Βάση δεδομένων σε SQL Server για την αποθήκευση δεδομένων όπως πιστοποιητικά, υποστηριζόμενες υπηρεσίες έκδοσης πιστοποιητικών (π.χ. εφορία, Τμήμα MIS, ΙΚΑ κλπ.), λίστα καταχωρημένων πολιτών κ.α.
- Windows εφαρμογή, για τη διαχείριση των πιστοποιητικών (διαγραφή, προβολή κλπ.), την έκδοση ζεύγους κλειδιών για κάθε υπηρεσία και πολίτη (για την υλοποίηση ψηφιακών υπογραφών),



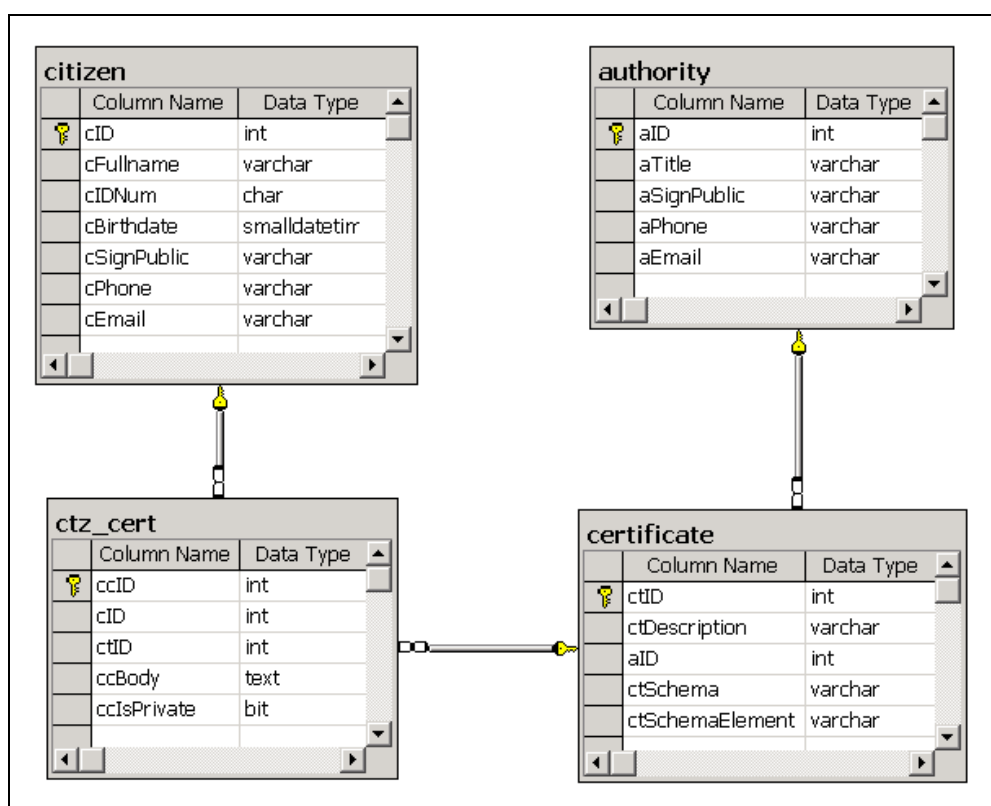
**Σχήμα 23 - Εσωτερική δομή του CertServer**

- ASP.NET εφαρμογή, για την πρόσβαση πολιτών και υπηρεσιών στο σύστημα μέσω ενός Web Browser και την διαχείριση των προσωπικών/ υπηρεσιακών στοιχείων τους, υποβολή αιτήσεων για έκδοση πιστοποιητικών οι οποίες θα μεταβιβάζονται στην αρμόδια υπηρεσία κλπ,
- Web Services, για την αποστολή και λήψη πιστοποιητικών από της υπηρεσίες,
- Class Library (.NET Assembly) για την πρόσβαση των προηγούμενων στοιχείων στη βάση δεδομένων.

## 10.2 Η βάση δεδομένων

Στο Σχήμα 24 φαίνεται το διάγραμμα της βάσης δεδομένων (όπως εξάγεται από τον αντίστοιχο επεξεργαστή του SQL Server). Η σχεδίαση αυτή έγινε με τα λιγότερα δυνατά πεδία για την απλούστευση της μελέτης.

Ο πίνακας citizen περιλαμβάνει στοιχεία πολιτών. Είναι προφανές ότι τα περιεχόμενα αυτού του πίνακα θα πρέπει να εισαχθούν από κάποια άλλο σύστημα, ενδεχομένως από πίνακες δημοτολογίων. Το πεδίο cSignPublic περιλαμβάνει το δημόσιο κλειδί που αντιστοιχεί σε κάθε πολίτη. Αυτό το κλειδί, με το αντίστοιχο ιδιωτικό, το οποίο διαθέτει μόνο ο πολίτης, αποτελούν το ζεύγος για την υλοποίηση ψηφιακών υπογραφών.



Σχήμα 24 – Διάγραμμα της βάσης δεδομένων

Ο πίνακας authority περιλαμβάνει στοιχεία των υπηρεσιών που υποστηρίζονται από το σύστημα, όπως εφορίες, πανεπιστημιακά τμήματα κλπ. Οι υπηρεσίες αυτές εκδίδουν πιστοποιητικά τα οποία αποστέλλουν στον CertServer για αποθήκευση, ενώ μπορούν να παραλαμβάνουν πιστοποιητικά που έχουν στείλει άλλες υπηρεσίες.

Ο πίνακας certificate περιλαμβάνει μια λίστα με τα ονόματα και άλλες πληροφορίες των πιστοποιητικών που εκδίδει η κάθε υπηρεσία. Για παράδειγμα, αν ένα πανεπιστημιακό τμήμα μπορεί να εκδίδει αντίγραφα πτυχίου και αναλυτικές βαθμολογίες, τότε θα υπάρχουν δυο εγγραφές γι αυτό, στον πίνακα certificate. Τα πεδία ctSchema και ctSchemaElement προσδιορίζουν το XML Schema που περιγράφει το κάθε πιστοποιητικό. Συγκεκριμένα, το πεδίο ctSchema περιλαμβάνει το namespace του XML Schema, ενώ το πεδίο ctSchemaElement, το οποίο είναι προαιρετικό, μπορεί να περιλαμβάνει κάποιο συγκεκριμένο element του XML Schema, σε περίπτωση που το namespace περιγράφει περισσότερα από ένα πιστοποιητικά.

```

<xs:element name="ptyxio">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="student_first" type="xs:string" />
      <xs:element name="student_last" type="xs:string" />
      <xs:element name="student_am" type="xs:string" />
      <xs:element name="institution" type="xs:string" />
      <xs:element name="department" type="xs:string" />
      <xs:element name="mesos_oros" type="xs:short" />
      <xs:element name="degree_title" type="xs:string" />
      <xs:element name="date_acquired" type="xs:dateTime" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
    <Reference URI="#Certificate">
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
      <DigestValue>VYbCy/2mhV/DjrJnDUBfOKRfTQg=</DigestValue>
    </Reference>
  </SignedInfo>
  <SignatureValue>bADFKwmN6GE3E6f0e6+kz0q19IZ3GQ2H9cgqXzSQck5pCYBspfkp/PuFWGfLx:
  <ptyxio xmlns="">
    <student_first>Παναδόπουλος</student_first>
    <student_last>Κωνσταντίνος</student_last>
    <student_am>3412/02</student_am>
    <institution>Πανεπιστήμιο Μακεδονίας</institution>
    <department>Τμήμα MIS</department>
    <mesos_oros>8.31</mesos_oros>
    <degree_title>Μεταπτυχιακό στα I.S.</degree_title>
    <date_acquired>04/03/2003</date_acquired>
  </ptyxio>
</Signature>

```

**Σχήμα 25 - XML Schema ενός αντιγράφου πτυχίου και ένα παράδειγμα ψηφιακά υπογεγραμμένου πιστοποιητικού για έναν υποθετικό φοιτητή**

Τέλος, ο πίνακας ctz\_cert, περιλαμβάνει τα πιστοποιητικά που αποστέλλονται από τις υπηρεσίες και αφορούν σε συγκεκριμένους τύπους πιστοποιητικών και συγκεκριμένους πολίτες. Το πεδίο ccBody περιλαμβάνει το ψηφιακά υπογεγραμμένο έγγραφο XML που αντιστοιχεί σε ένα πιστοποιητικό, ενώ τα πεδία ciD και ctID περιέχουν τους κωδικούς του

πολίτη και του πιστοποιητικού που αναπαριστά η εγγραφή. Τέλος, το πεδίο ccIsPrivate παίρνει την τιμή 1 όταν το πιστοποιητικό (περιεχόμενο του πεδίου ccBody) είναι κρυπτογραφημένο με το δημόσιο κλειδί του πολίτη που αντιστοιχεί. Όπως περιγράφεται και παρακάτω, αυτή η μέθοδος εφαρμόζεται όταν το πιστοποιητικό είναι ιδιωτικού χαρακτήρα και μόνο ο πολίτης μπορεί να έχει πρόσβαση σε αυτό.

Στο Σχήμα 25 φαίνεται ένα παράδειγμα ενός XML Schema που αντιστοιχεί σε ένα αντίγραφο πτυχίου, καθώς και ένα αντίστοιχο XML, ψηφιακά υπογεγραμμένο από την αντίστοιχη υπηρεσία. Η διαδικασία υπογραφής περιγράφεται στην παράγραφο 10.4.

```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema targetNamespace="http://uom.gr/dipapadi/masters.xsd"
  elementFormDefault="qualified"
  xmlns="http://uom.gr/masters.xsd"
  xmlns:mstns="http://uom.gr/masters.xsd"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="ptyxio">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="student_first" type="xs:string" />
        <xs:element name="student_last" type="xs:string" />
        <xs:element name="student_am" type="xs:string" />
        <xs:element name="institution" type="xs:string" />
        <xs:element name="department" type="xs:string" />
        <xs:element name="mesos_oros" type="xs:short" />
        <xs:element name="degree_title" type="xs:string" />
        <xs:element name="date_acquired" type="xs:dateTime" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Σχήμα 26 - XML Schema Πτυχίου Μεταπτυχιακών Σπουδών

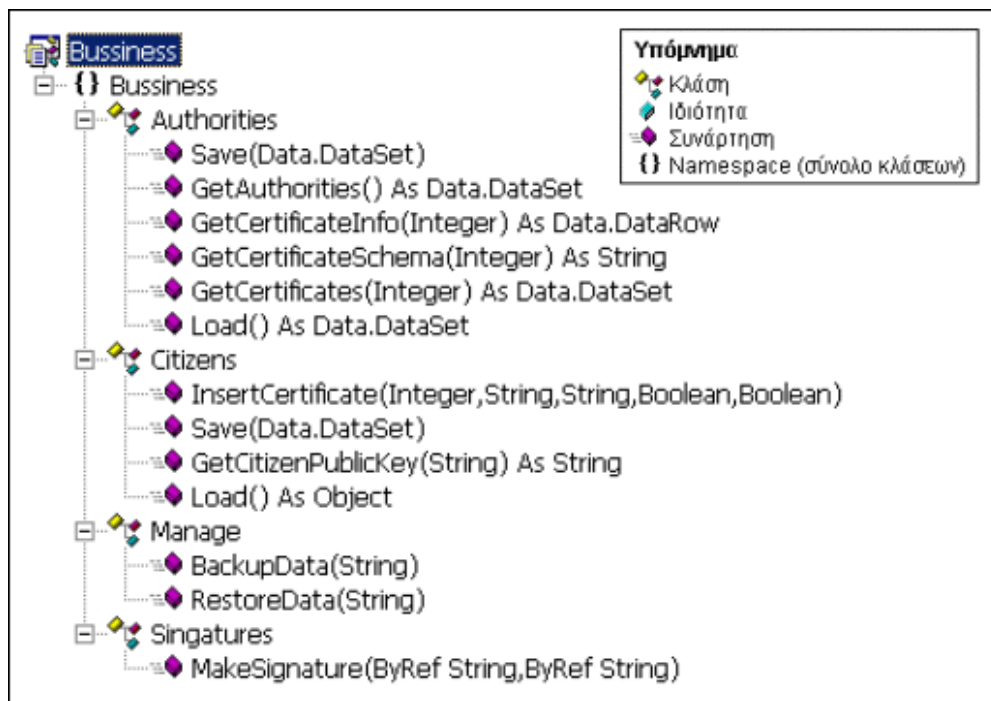
### 10.3 Πρόσβαση στη βάση δεδομένων - .NET Assembly και ADO.NET

Συνηθίζεται σε πληροφοριακά συστήματα μεσαίας ή μεγάλης κλίμακας, ο κώδικας που γράφεται για την διάδραση με τον χρήστη (UI: User Interface) να μην έχει άμεση πρόσβαση με τη βάση δεδομένων, να μην εκτελεί δηλαδή άμεσα ερωτήματα SQL που μεταφράζουν τις επιλογές του χρήστη άμεσα σε εντολές προς τη βάση δεδομένων.

Αυτό είναι επιθυμητό για διάφορους λόγους. Υπάρχουν περιπτώσεις που τα ίδια ερωτήματα SQL πρέπει να χρησιμοποιηθούν σε διάφορα σημεία και

άλλες περιπτώσεις όπου ένα σύνολο ερωτημάτων να πρέπει να εκτελεστεί για την ολοκλήρωση μιας διαδικασίας. Πολλές φορές επίσης δεν είναι επιθυμητό ο προγραμματιστής που δημιουργεί το UI, να καθορίζει τον τρόπο χρήσης της βάσης δεδομένων, αλλά κάποιος άλλος προγραμματιστής. Για τους λόγους αυτούς, συνηθίζεται η δημιουργία ενός επιπέδου μεταξύ της βάσης δεδομένων και του UI, το οποίο συνήθως ονομάζεται Business Layer και αναλαμβάνει τη ροή δεδομένων μεταξύ των δυο προηγούμενων.

Στην περίπτωση του CertServer, προτείνεται η δημιουργία μιας .NET Assembly (δηλαδή ενός dll) ως Business Layer, με τη δομή που φαίνεται στο Σχήμα 27.



**Σχήμα 27 - Μοντέλο Αντικειμένων (Object Model) του Business Layer του CertServer**

Οι κλάσεις που φαίνονται στο σχήμα μπορούν να εκτελέσουν όλες τις λειτουργίες του CertServer. Προσθέτοντας την παραπομπή αυτής της assembly σε κάποια εφαρμογή, είναι δυνατή η χρήση των κλάσεων που περιέχει. Στον παρακάτω κώδικα γίνεται χρήση της συνάρτησης Load της κλάσης Citizens:

```
Dim cCitizen As New Bussiness.Citizens()  
dataSet = cCitizen.Load()
```

Στο εσωτερικό της assembly, γίνεται χρήση του ADO.NET για την πρόσβαση στη βάση δεδομένων που περιγράφηκε στην προηγούμενη παράγραφο. Για παράδειγμα, ο κώδικας που εκτελείται στην συνάρτηση Load που προαναφέρθηκε είναι ο παρακάτω:

```
Public Function Load()  
    'load from datasource  
    Dim datDataset As New DataSet()  
    Dim datAdapter As New SqlClient.SqlDataAdapter()  
    datAdapter.SelectCommand = New SqlClient.SqlCommand()  
    datAdapter.SelectCommand.Connection = New  
Data.SqlClient.SqlConnection(General.ConnectionString)  
  
    datAdapter.SelectCommand.CommandText = "SELECT * FROM  
citizen"  
    datAdapter.Fill(datDataset, 0, 200, "citizen")  
  
    datAdapter.SelectCommand.CommandText = "SELECT c.*,  
b.ctDescription FROM ctz_cert AS c INNER JOIN certificate AS b ON  
c.ctID=b.ctID"  
    datAdapter.Fill(datDataset, "ctz_cert")  
  
    'declare the relation between the two tables  
    datDataset.Relations.Add( _  
        "Πιστοποιητικά Πολίτη", _  
        datDataset.Tables("citizen").Columns("cID"), _  
        datDataset.Tables("ctz_cert").Columns("cID"), True)  
  
    datAdapter.Dispose()  
  
    'return result  
    Return datDataset  
End Function
```

Στις πρώτες γραμμές του κώδικα αρχικοποιείται ένα αντικείμενο τύπου SqlDataAdapter και ένα αντικείμενο τύπου SqlCommand, το οποίο θα εκτελέσει το ερώτημα SQL προς τη βάση δεδομένων. Η τοποθεσία της βάσης δεδομένων ορίζεται μέσω της μεταβλητής General.ConnectionString. Η τιμή αυτής της μεταβλητής είναι της μορφής που φαίνεται παρακάτω, στην περίπτωση του SQL Server, όπου διακρίνεται το όνομα του υπολογιστή που βρίσκεται η βάση δεδομένων (myServerName) και το όνομα της βάσης δεδομένων (certServerDB) :

```
General.ConnectionString = "data source=myServerName;initial  
catalog=certServerDB;integrated security=SSPI;persist security  
info=False; packet size=4096"
```

Περισσότερα για την μορφή των Connection Strings, βλ. εγχειρίδιο ADO.NET - .NET Framework.

Μετά τη δημιουργία των παραπάνω αντικειμένων εκτελείται η εντολή Fill του Data Adapter, για το φόρτωμα των επιστρεφόμενων αποτελεσμάτων δυο ερωτημάτων SQL σε ένα αντικείμενο DataSet. Τέλος ορίζεται η σχέση μεταξύ των δυο συνόλων δεδομένων που επιστρέφονται με τη μορφή ενός Foreign Key Constraint, μέσω της ιδιότητας Relations του DataSet. Περισσότερες πληροφορίες για τη χρήση του ADO.NET για την πρόσβαση σε βάσεις δεδομένων, βλ. Κεφάλαιο 5.

### Έκδοση ψηφιακών υπογραφών

Μεταξύ των λειτουργιών που εκτελούνται από την συγκεκριμένη assembly, είναι και η έκδοση ζευγών κλειδιών για την υλοποίηση ψηφιακών υπογραφών. Για το σκοπό αυτό, το .NET Framework περιλαμβάνει μια σειρά από κλάσεις, μέρος της assembly System.Security.Cryptography, οι οποίες ενσωματώνουν αλγόριθμους κρυπτογράφησης. Η κλάση που χρησιμοποιείται από τον CertSrv είναι η System.Security.Cryptography.RSACryptoServiceProvider, που ενσωματώνει τον αλγόριθμο κρυπτογράφησης RSA<sup>46</sup>. Παρακάτω φαίνεται ο κώδικας της συνάρτησης MakeSignature που διακρίνεται και στα περιεχόμενα της κλάσης Signatures στο Σχήμα 27.

```
Public Sub MakeSignature(ByRef all As String, ByRef onlyPublic As
                        String)
    Dim sPublicKey As String, sPublicAndPrivateKey As String
    Dim cCrypto As New Cryptography.RSACryptoServiceProvider()
    all = cCrypto.ToXmlString(True)
    onlyPublic = cCrypto.ToXmlString(False)
End Sub
```

### Παράδειγμα 9 – Έκδοση ζεύγους κλειδιών για ψηφιακές υπογραφές RSA

Ο παραπάνω κώδικας απλά δημιουργεί ένα τυχαίο ζεύγος κλειδιών, το οποίο περνάει στις μεταβλητές *all* και *onlyPublic* που επιστρέφονται ως

---

<sup>46</sup> Άλλοι αλγόριθμοι κρυπτογράφησης που υποστηρίζονται, με αντίστοιχες κλάσεις στην assembly System.Security.Cryptography είναι οι DSA, MD5, DES, Triple DES, RNG, SHA1 και RC2.

αποτελέσματα της συνάρτησης<sup>47</sup>. Περισσότερα για τις ψηφιακές υπογραφές, καθώς και για τον τρόπο που θα πρέπει να γίνεται η διανομή των κλειδιών που παράγονται, σχολιάζονται στην επόμενη παράγραφο.

#### 10.4 Διαχείριση πιστοποιητικών – έκδοση κλειδιών ψηφιακών υπογραφών (Windows Application)

Ο CertServer περιλαμβάνει μια τυπική Windows εφαρμογή (βλ. Σχήμα 23), μέσω της οποίας το προσωπικό της υπηρεσίας διαχειρίζεται τα περιεχόμενα της βάση δεδομένων. Στο Σχήμα 28 φαίνεται το παράθυρο μέσω του οποίου, ο αρμόδιος υπάλληλος της υπηρεσίας, μπορεί να επεξεργαστεί τα στοιχεία ενός πολίτη, καθώς και να δει τα πιστοποιητικά που είναι καταχωρημένα γι αυτόν. Για το φόρτωμα των σχετικών δεδομένων, χρησιμοποιείται η συνάρτηση `Bussiness.Citizens.Load`, που σχολιάστηκε στην προηγούμενη παράγραφο.

Περιγραφή	Ιδιωτικό
Δίπλωμα Μεταπτυχιακών Σπουδών	<input checked="" type="checkbox"/>
Φορολογική Ενημερότητα	<input checked="" type="checkbox"/>

Σχήμα 28 - Παράθυρο επεξεργασίας στοιχείων πολίτη και προβολή των καταχωρημένων, γι αυτόν, πιστοποιητικών

---

<sup>47</sup> Αυτό δηλώνει η λέξη *ByRef*, της Visual Basic, στο όρισμα της συνάρτησης, ότι δηλαδή οι τιμές των μεταβλητών αυτών θα επιστραφούν στον κώδικα που κάλεσε τη συνάρτηση. Η αντίστοιχη λέξη στην C# είναι η *Ref*.

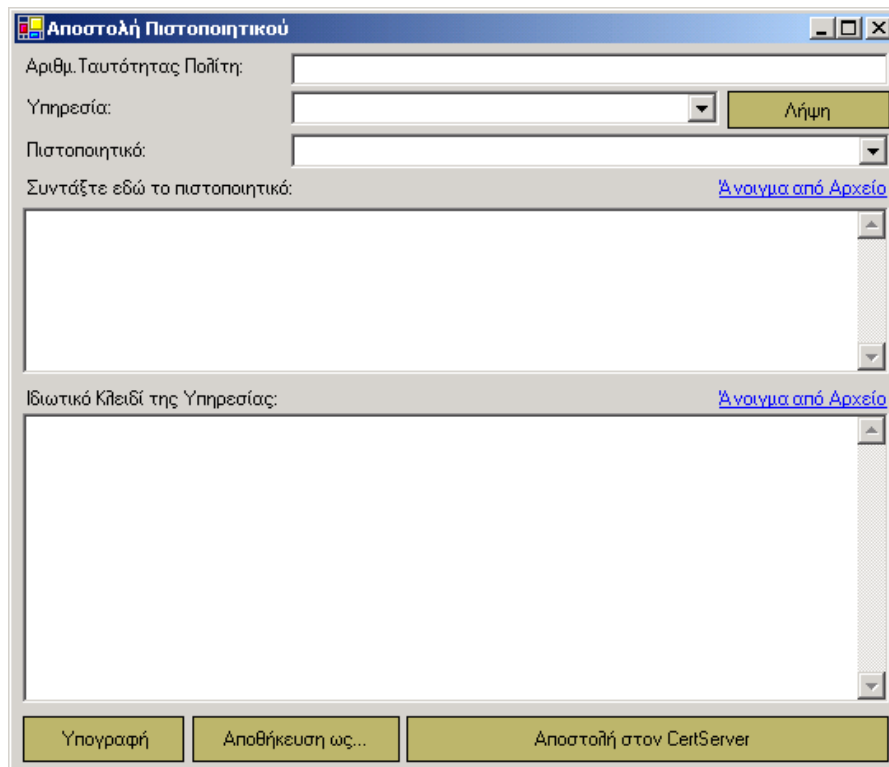
Η εφαρμογή έχει επίσης τη δυνατότητα έκδοσης ζεύγους κλειδιών που μπορούν να χρησιμοποιηθούν για την ψηφιακή υπογραφή εγγράφων. Στο ίδιο παράθυρο, κάνοντας κλικ στο πλήκτρο 'Έκδοση Νέας Υπογραφής', γίνεται χρήση της συνάρτησης `Bussiness.Signatures.MakeSignature`, για την έκδοση ενός ζεύγους κλειδιών για τον πολίτη. Το δημόσιο κλειδί αποθηκεύεται στη βάση δεδομένων του CertServer (πεδίο `cSignPublic` του πίνακα `citizen`), ενώ το ιδιωτικό κλειδί πρέπει να παραδοθεί με ασφαλή τρόπο στον πολίτη και μόνο σε αυτόν. Η συγκεκριμένη οθόνη εξάγει αυτό το κλειδί σε ένα αρχείο XML, θα ήταν όμως δυνατή η ενσωμάτωση του κλειδιού σε κάποια μαγνητική κάρτα (*smart card*).

Η τεχνολογία των ψηφιακών υπογραφών, η οποία αναμένεται σύντομα να αποκτήσει νομική υπόσταση και στην Ελλάδα, μπορεί να εξασφαλίσει την αυθεντικότητα ενός εγγράφου. Αυτό σημαίνει ότι ένα ψηφιακά υπογεγραμμένο έγγραφο (σε μορφή απλού κειμένου, XML, e-mail κ.α.) είναι πρακτικά αδύνατο να τροποποιηθεί από οποιονδήποτε, εκτός από τον αυθεντικό εκδότη του. Είναι επίσης δυνατό, ένα ψηφιακά υπογεγραμμένο έγγραφο να κρυπτογραφηθεί, έτσι ώστε μόνο συγκεκριμένος παραλήπτης να μπορεί να το διαβάσει. Αυτή η δυνατότητα μπορεί να αξιοποιηθεί από τον CertServer, στην περίπτωση που ένα πιστοποιητικό που εκδίδεται από κάποια υπηρεσία, πρέπει να είναι αναγνώσιμο μόνο από τον πολίτη τον οποίο αφορά, όπως πιστοποιητικά ασθενείας, αποφυλάκισης κ.α.

### **Διαδικασία Ψηφιακής Υπογραφής και Αποστολής Πιστοποιητικών**

Ένα δεύτερο πρόγραμμα που περιλαμβάνεται στο σύστημα, το οποίο όμως θα μπορούσε να αποτελεί και μέρος του πληροφοριακού συστήματος που χρησιμοποιεί η κάθε υπηρεσία, είναι το πρόγραμμα υπογραφής και αποστολής πιστοποιητικών στον CertServer. Στην απλουστευμένη του περίπτωση, το πρόγραμμα αυτό περιλαμβάνει την φόρμα που φαίνεται στο Σχήμα 29. Στο πρώτο πλαίσιο της φόρμας εισάγεται το πιστοποιητικό σε μορφή XML και στο δεύτερο πλαίσιο εισάγεται το ιδιωτικό κλειδί της υπηρεσίας. Πατώντας το πλήκτρο 'Υπογραφή', το XML του πρώτου πλαισίου υπογράφεται ψηφιακά, χρησιμοποιώντας το ιδιωτικό κλειδί που

δόθηκε και την τεχνολογία SignedXML του W3C<sup>48</sup>, όπως φαίνεται και στον παρακάτω κώδικα.



**Σχήμα 29 – Παράθυρο αποστολής ψηφιακά υπογεγραμμένου πιστοποιητικού στον CertServer**

```
'φόρτωμα του xml στο αντικείμενο Cryptography.Xml.DataObject
Dim cXML As New XmlDocument()
cXML.LoadXml(xmlDoc)
Dim cDataObject As New DataObject()
cDataObject.LoadXml(cXML.ChildNodes(0))

'δημιουργία αντικειμένου κρυπτογράφησης και φόρτωμα του κλειδιού
Dim cRSA As New RSACryptoServiceProvider()
cRSA.FromXmlString(key)

'δημιουργία αντικειμένου signedxml και προσθήκη του XML document
Dim cSignedXML As New SignedXml()
cSignedXML.SigningKey = cRSA
cSignedXML.AddObject(cDataObject)
'το reference προσδιορίζει την πληροφορία που προστίθεται, σε
'περίπτωση που προσθέσουμε άλλη πληροφορία στη συνέχεια
```

<sup>48</sup> Η τεχνολογία αυτή, η οποία παρακολουθείται από το W3C, ενσωματώνει σε έγγραφα XML, ψηφιακά υπογεγραμμένες πληροφορίες, στην περίπτωσή μας το XML πιστοποιητικό. Στο .NET Framework, τα SignedXML υλοποιούνται με την κλάση 'System.Security.Cryptography.Xml.SignedXml' (βλ. εγχειρίδιο χρήσης .NET Framework)

```
cSignedXML.AddReference (New Reference (xmlDocURI))
```

```
'υπολογισμός υπογραφής και επιστροφή αποτελέσματος
```

```
cSignedXML.ComputeSignature ()
```

```
Return cSignedXML.GetXml.OuterXml
```

Αυτό το τμήμα κώδικα βέβαια, δεν είναι ενσωματωμένο στην .NET Assembly που περιγράφηκε στην παράγραφο 10.3, μια που πρέπει να είναι προσβάσιμο από ένα πρόγραμμα που θα λειτουργεί εκτός του περιβάλλοντος του CertServer, για παράδειγμα στο γραφείο έκδοσης πιστοποιητικών κάποιας υπηρεσίας.

Με τη χρήση του πλήκτρου 'Αποστολή στον CertServer', γίνεται η αποστολή του πιστοποιητικού. Ο σχετικός κώδικας φαίνεται στην ακόλουθη παράγραφο.

## 10.5 Αποστολή και λήψη πιστοποιητικών - Web Services

Ο CertServer περιλαμβάνει Web Services μέσω των οποίων γίνεται η αποστολή των ψηφιακά υπογεγραμμένων πιστοποιητικών από και προς αυτόν, από τις ενδιαφερόμενες υπηρεσίες. Στο Παράδειγμα 10 φαίνεται μια από τις συναρτήσεις, που υλοποιείται ως Web Service, για την αποστολή ενός πιστοποιητικού από μια υπηρεσία (Sub InsertCertificate). Στην παράμετρο signedCertificate ορίζεται το ψηφιακά υπογεγραμμένο, από την αρμόδια υπηρεσία, XML, όπως αυτό στο 2<sup>ο</sup> μέρος στο Σχήμα 25. Οι παράμετροι certificateID και cIDNum καθορίζουν τον κωδικό του πιστοποιητικού και τον αριθμό ταυτότητας του πολίτη στον οποίο αντιστοιχεί, ενώ η παράμετρος ccIsPrivate δηλώνει αν το πιστοποιητικό που αποστέλλεται έχει κρυπτογραφηθεί με το δημόσιο κλειδί του πολίτη, έτσι ώστε μόνον αυτός να μπορεί να το διαβάσει.

```
<WebMethod()> _  
    Public Sub InsertCertificate(ByVal certificateID As Integer,  
ByVal cIDNum As String, ByVal signedCertificate As String, ByVal  
ccIsPrivate As Boolean)  
        Dim cCitizen As New Bussiness.Citizens()  
        cCitizen.InsertCertificate(certificateID, cIDNum,  
signedCertificate, ccIsPrivate, True)  
    End Sub
```

**Παράδειγμα 10 – Κώδικας υλοποίησης της συνάρτησης του Web Service για την καταχώρηση πιστοποιητικού που αποστέλλει κάποια υπηρεσία**

Στο Παράρτημα 4 φαίνεται το WSDL έγγραφο που περιγράφει τις λειτουργίες των Web Services του CertServer, ενώ ακολουθεί ο κώδικας που χρειάστηκε για την υλοποίησή τους στο Visual Studio .NET. Η κάθε μια από τις παρακάτω συναρτήσεις αποτελεί μια συνάρτηση η οποία μπορεί να εκτελεστεί αυτόνομα ως Web Service από κάποιο απομακρυσμένο χρήστη ή εφαρμογή. Οι συναρτήσεις χρησιμοποιούν λειτουργίες της .NET Assembly του CertServer, οι οποίες δεν έχουν δημιουργηθεί απαραίτητα για χρήση μόνο σε αυτά τα Web Services.

```
<WebMethod()> _
Public Function GetCertificates(authorityID As Integer) As DataSet
    Dim cAuth As New Bussiness.Authorities()
    Return cAuth.GetCertificates(authorityID)
End Function

<WebMethod()> _
Public Function GetAuthorities() As DataSet
    Dim cAuth As New Bussiness.Authorities()
    Return cAuth.GetAuthorities()
End Function

<WebMethod()> _
Public Function GetCitizenPublicKey(cIDNum As String) As String
    Dim cCitizen As New Bussiness.Citizens()
    Return cCitizen.GetCitizenPublicKey(cIDNum)
End Function

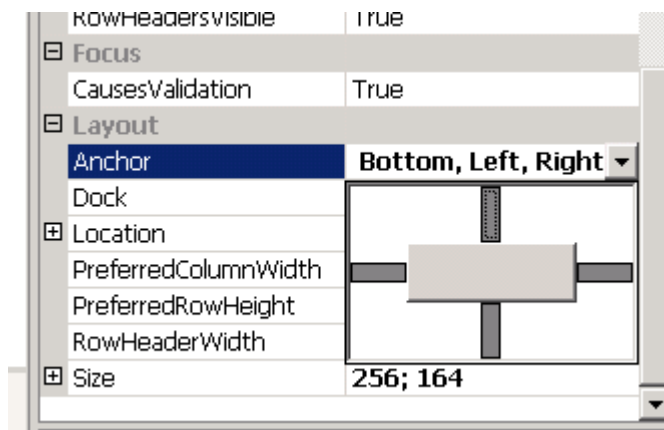
<WebMethod()> _
Public Sub InsertCertificate(ByVal certificateID As Integer, ByVal
cIDNum As String, ByVal signedCertificate As String, ByVal
ccIsPrivate As Boolean)
    Dim cCitizen As New Bussiness.Citizens()
    cCitizen.InsertCertificate(certificateID, cIDNum,
signedCertificate, ccIsPrivate, True)
End Sub
```

## 10.6 Διάφορες λειτουργίες του .NET Framework και του Visual Studio .NET

Εκτός από την πρακτική εφαρμογή των τεχνολογιών που μελετήθηκαν σε αυτή την εργασία, η σχεδίαση και ανάπτυξη του CertServer ανέδειξε διάφορες άλλες λειτουργίες του .NET Framework και του Visual Studio .NET. Αυτές οι λειτουργίες, αν και μάλλον μικρότερης σημασίας, βοηθούν σημαντικά στην ταχύτερη ανάπτυξη ενός συστήματος, σε διάφορα επίπεδα, όπως στην υλοποίηση της επιχειρησιακής λογικής του και την ανάπτυξη του

User Interface. Μερικές μόνο από αυτές τις λειτουργίες παρουσιάζονται σε αυτή την παράγραφο:

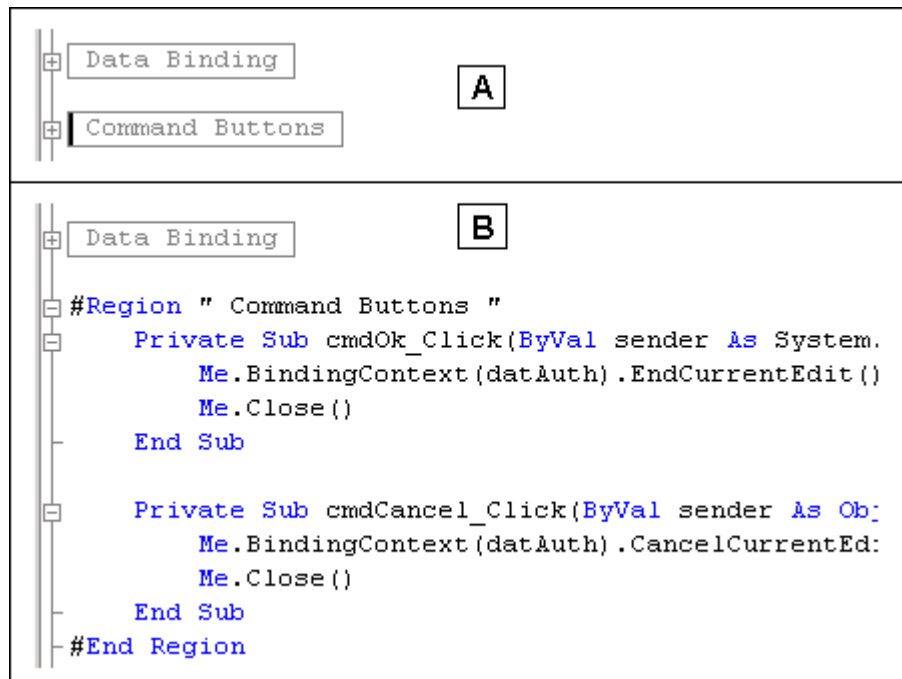
- **Σχεδίαση φορμών:** όλα τα στοιχεία που μπορούν να χρησιμοποιούν σε μια φόρμα για τη σχεδίαση του User Interface, μπορούν να έχουν προκαθορισμένη συμπεριφορά ως προς τις αλλαγές μεγέθους της φόρμας αυτής. Είναι δυνατόν δηλαδή, ένα πλαίσιο κειμένου να οριστεί ότι θα αλλάζει το μέγεθός του ως προς το αριστερό και το δεξί άκρο του, που σημαίνει ότι κάθε φορά που η φόρμα αλλάζει μέγεθος, το πλαίσιο κειμένου θα αλλάζει αναλογικά το μέγεθός του καταλαμβάνοντας το κενό χώρο που δημιουργείται. Αυτή η λειτουργία υλοποιείται μέσω της ιδιότητας *Anchor* που υπάρχει σε όλα τα στοιχεία UI. Στο Σχήμα 30 φαίνεται η ιδιότητα αυτή στον επεξεργαστή ιδιοτήτων (property editor) του Visual Studio .NET. Αυτή η λειτουργία βοηθά στη σχεδίαση πολύπλοκων φορμών χωρίς τη σύνταξη κώδικα ελέγχου των στοιχείων ως προς τη θέση και το μέγεθός τους.



Σχήμα 30 - Η ιδιότητα Anchor

- **Περιοχές Κώδικα:** χρησιμοποιώντας τις λέξεις #Region και #End Region, μπορούμε να διαχωρίσουμε περιοχές κώδικα οι οποίες είναι δυνατό να αποκρύπτονται από τον συνολικό αρχείο που είναι υπό επεξεργασία. Ο προγραμματιστής λοιπόν έχει τη δυνατότητα να οργανώσει τον κώδικά του σε λογικές ενότητες και να αποκρύπτει ή να εμφανίζει μόνο τον κώδικα που τον ενδιαφέρει ανάλογα με την περίπτωση. Η οργάνωση αυτή, η οποία φαίνεται και στο Σχήμα 31 στο οποίο γίνεται απόκρυψη και εμφάνιση της περιοχής Command

Buttons, γίνεται μόνο με σκοπό την οπτική οργάνωση του κώδικα και δεν έχει καμία επίπτωση στην εκτέλεσή του.



Σχήμα 31 - Περιοχές Κώδικα

- **Exception Handling:** Ο έλεγχος των σφαλμάτων που μπορεί να προκύψουν κατά την εκτέλεση ενός τμήματος του κώδικα γίνεται μέσω των μπλοκ Try...Catch...Finally. Στον παρακάτω κώδικα φαίνεται το φόρτωμα των στοιχείων ενός πολίτη, μέσα σε ένα μπλοκ ελέγχου σφαλμάτων, το οποίο θα παραπέμψει την εκτέλεση του κώδικα στο μπλοκ Catch, σε περίπτωση που συμβεί κάποια σφάλμα στο μπλοκ Try. Περισσότερες πληροφορίες για τον έλεγχο των σφαλμάτων στο .NET Framework, βλ. Εγχειρίδιο Χρήσης .NET Framework.

```
Try  
    If datCitizens Is Nothing Then  
        Dim cCit As New Bussiness.Citizens()  
        datCitizens = cCit.Load()  
    End If  
Catch exc As Exception  
    ShowException("Σφάλμα κατά τη λήψη των δεδομένων", exc)  
    Return False  
End Try
```

- **Events:** Κάθε αντικείμενο, όπως ίσως είναι γνωστό, περιλαμβάνει μια σειρά από ενέργειες τις οποίες μπορεί να κάνει ο χρήστης με αυτό. Αν πρόκειται για αντικείμενο σχεδίασης UI (User Interface), προφανώς αυτός ο χρήστης είναι ο τελικός χρήστης του προγράμματος, ενώ αν πρόκειται για κοινές κλάσεις, ο χρήστης των ενεργειών είναι ο κώδικας που καλεί την κλάση. Για παράδειγμα, ένα πλήκτρο μπορεί να έχει events όπως click, mouseOver, keyPress κλπ. οι οποίες αντιστοιχούν στις ενέργειες πατήματος του πλήκτρου, περάσματος του ποντικιού από επάνω του ή πατήματος ενός πλήκτρου του πληκτρολογίου αντίστοιχα. Η υλοποίηση των events στο .NET Framework και την Visual Basic .NET γίνεται με τον παρακάτω τρόπο:

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles Button1.Click
    MessageBox.Show("Πάτημα πλήκτρου!")
End Sub
```

Το όρισμα της συνάρτησης που διαχειρίζεται ένα event είναι πάντα σταθερό και περιλαμβάνει τα αντικείμενα sender και e. Αυτά περιέχουν πληροφορίες για το αντικείμενο που προκάλεσε το event (στην περίπτωσή μας το αντικείμενο Button1) και παραμέτρους σχετικά με το event (π.χ. στην περίπτωση του Button1.Click event, ποιο πλήκτρο του ποντικιού πατήθηκε) αντίστοιχα. Η λέξη Handles στο τέλος του ορίσματος της συνάρτησης ορίζει ποιο event θα ενεργοποιεί τη συγκεκριμένη συνάρτηση. Είναι δυνατόν μια συνάρτηση να ενεργοποιείται για περισσότερα από ένα αντικείμενα. Για παράδειγμα, ο παρακάτω κώδικας διαχειρίζεται τα click events δυο πλήκτρων. Ελέγχοντας τα περιεχόμενα της παραμέτρου sender μπορούμε να ξέρουμε ποιο πλήκτρο πατήθηκε κάθε φορά.

```
Private Sub Buttons_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles Button1.Click, Button2.Click
    MessageBox.Show("Πάτημα πλήκτρου!")
End Sub
```

- **Data Binding:** DataBinding ονομάζεται η διαδικασία αντιστοίχισης στοιχείων του UI με πεδία μιας πηγής δεδομένων. Ενεργοποιώντας

αυτή τη λειτουργικότητα, οι τιμές των αντιστοιχισμένων οντοτήτων θα διατηρούνται συγχρονισμένες. Είναι δυνατή η αντιστοίχιση στοιχείων όπως πλαίσια κειμένου, λίστες επιλογής κ.α. με στήλες ενός πίνακα που περιλαμβάνεται σε ένα DataSet, αλλά και με ιδιότητες ενός αντικειμένου. Στην ανάπτυξη του CertServer, απαιτήθηκε η υλοποίηση της πρώτης περίπτωσης. Παρακάτω φαίνεται ο κώδικας που υλοποιεί το DataBinding της φόρμας επεξεργασίας των στοιχείων ενός πολίτη (βλ. Σχήμα 28):

```
Public Function Edit(citizen As DataSet, newRecord As Boolean)
Try
    Dim dbn As Binding
    datCitizen = citizen.Tables("citizen")

    'bind datatable with textboxes
    txtFullname.DataBindings.Add("Text", datCitizen, "cFullname")
    txtIDNum.DataBindings.Add("Text", datCitizen, "cIDNum")
    txtPhone.DataBindings.Add("Text", datCitizen, "cPhone")
    txtEmail.DataBindings.Add("Text", datCitizen, "cEmail")

    'bind birtdate field. add custom handlers to format the date
    dbn = New Binding("Text", datCitizen, "cBirthdate")
    AddHandler dbn.Format, AddressOf FormatDate
    AddHandler dbn.Parse, AddressOf ParseDate
    txtBirthdate.DataBindings.Add(dbn)

    'bind signpublic field. add custom handlers to format the value
    dbn = New Binding("Text", datCitizen, "cSignPublic")
    AddHandler dbn.Format, AddressOf FormatSignature
    txtSignature.DataBindings.Add(dbn)
    txtSignature.DataBindings.Add("Tag", datCitizen, "cSignPublic")

    'bind citizen's certificate with datagrid
    datCertificates = citizen.Tables("ctz_cert")
    DataGridView1.DataSource = datCertificates

    'add a new record into the datasource if new record was
    'requested to be added
    If newRecord Then
        Me.BindingContext(datCitizen).AddNew()
    End If

    Me.ShowDialog()
Catch exc As Exception
    ShowException("Σφάλμα κατά την επεξεργασία των στοιχείων
του πολίτη", exc)
    Me.BindingContext(datCitizen).CancelCurrentEdit()
End Try
End Function
```

Τα στοιχεία UI που έχουν τη δυνατότητα DataBinding περιλαμβάνουν την ιδιότητα DataBindings μέσω της οποίας προσθέτουμε μια αντιστοίχιση κάποιας ιδιότητάς τους με μια στήλη του πίνακα datCitizen. Για παράδειγμα, στο στοιχείο txtFullname (το στοιχείο εισαγωγής του ονοματεπώνυμου του πολίτη – βλ. Σχήμα 28) γίνεται αντιστοίχιση της ιδιότητάς του Text με τη στήλη cfullname του πίνακα citizen (βλ. Σχήμα 24 – Διάγραμμα της βάσης δεδομένων). Μεταξύ άλλων, σε αυτή τη συνάρτηση μπορείτε να διακρίνετε τον έλεγχο σφαλμάτων (exception handling) με τη βοήθεια του μπλοκ Try...Catch.

## 10.7 Ένα σενάριο λειτουργίας του CertServer

Μια περίπτωση χρήσης του CertServer είναι η εγγραφή ενός φοιτητή σε ένα πανεπιστημιακό τμήμα. Συγκεκριμένα, η εγγραφή ενός φοιτητή σε ένα μεταπτυχιακό πρόγραμμα μπορεί να απαιτεί την προσκόμιση των παρακάτω δικαιολογητικών:

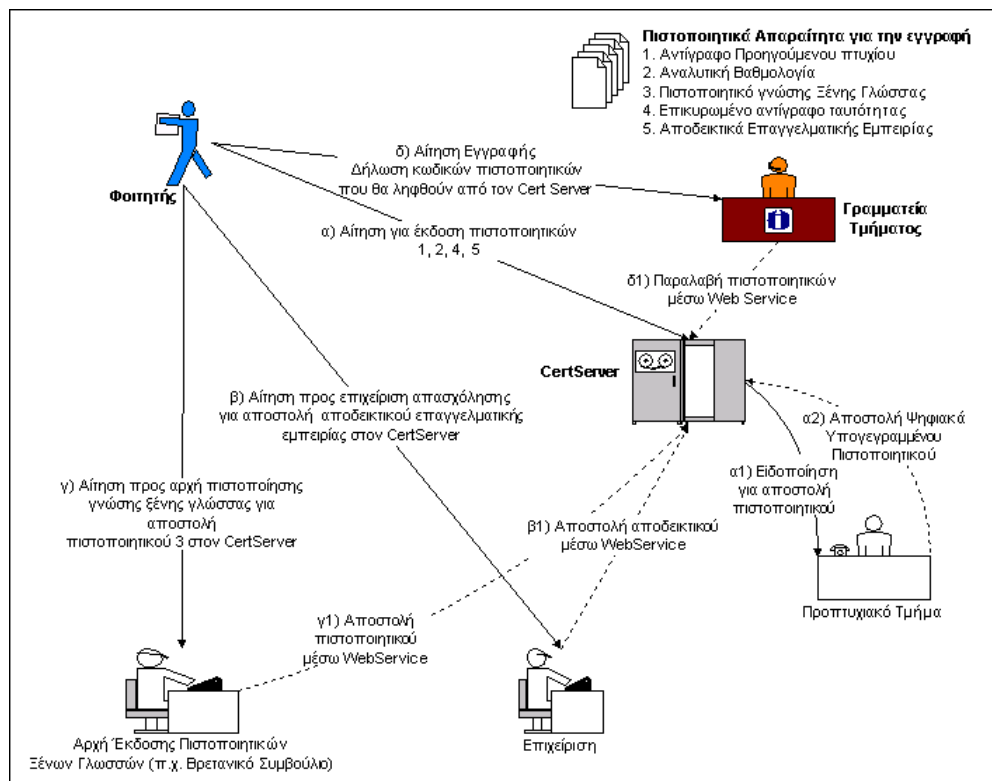
1. Αντίγραφο πτυχίου προπτυχιακών σπουδών
2. Αναλυτική βαθμολογία προπτυχιακών σπουδών
3. Πιστοποιητικό γνώσης ξένης γλώσσας
4. Επικυρωμένο αντίγραφο αστυνομικής ταυτότητας
5. Αποδεικτικά επαγγελματικής εμπειρίας

Τα βήματα που θα πρέπει να ακολουθήσει ο ενδιαφερόμενος φοιτητής για να ολοκληρώσει την εγγραφή του, τα οποία φαίνονται και στο Σχήμα 32, είναι τα εξής:

α) Θα πρέπει να υποβάλλει μια ηλεκτρονική αίτηση μέσω του Web Site του CertServer για την έκδοση των πιστοποιητικών 1, 2 και 4. Ο CertServer θα είναι υπεύθυνος για την ύπαρξη αυτών των πιστοποιητικών στη βάση δεδομένων του. Σε περίπτωση που αυτά τα πιστοποιητικά δεν υπάρχουν, λόγω κάποιας προηγούμενης συναλλαγής του ενδιαφερόμενου πολίτη, ο CertServer θα πρέπει να έρθει σε επαφή με τις αρμόδιες αρχές (α1). Κάθε μια από αυτές θα πρέπει να δημιουργήσει το πιστοποιητικό (έγγραφο XML), να το υπογράψει ψηφιακά χρησιμοποιώντας το ιδιωτικό της κλειδί και να αποστείλει το ψηφιακά υπογεγραμμένο XML, μέσω Web Service στον CertServer (α2).

Είναι προφανές ότι το πιστοποιητικό Νο 4, δηλαδή η αστυνομική του ταυτότητα, θα πρέπει να υπάρχει στην βάση δεδομένων του CertServer από τη στιγμή της εγγραφής του πολίτη σε αυτή.

β και γ) Ταυτόχρονα, ο ενδιαφερόμενος προς εγγραφή φοιτητής, γνωρίζοντας ότι τα υπόλοιπα πιστοποιητικά (3 και 5) δεν υπάρχουν ήδη στον CertServer – αυτή την πληροφορία μπορεί να την πάρει από το Web Site, από όπου μπορεί να δει μια λίστα με τα καταχωρημένα γι αυτόν πιστοποιητικά – ειδοποιεί τους αρμόδιους φορείς ή επιχειρήσεις για την έκδοση και αποστολή τους στον CertServer.



**Σχήμα 32 – Σενάριο λειτουργίας του Cert Server**

Μετά την σύνταξη των πιστοποιητικών και την υπογραφή τους ψηφιακά, τα ψηφιακά υπογεγραμμένα XML αποστέλλονται από τον κάθε φορέα ή επιχείρηση στον CertServer (β1 και γ1).

δ) Μετά την εξασφάλιση της ύπαρξης των απαραίτητων πιστοποιητικών στον CertServer, ο ενδιαφερόμενος φοιτητής μπορεί να υποβάλλει την αίτηση εγγραφής του στο Μεταπτυχιακό Πρόγραμμα, επισυνάπτοντας τις περιγραφές των πιστοποιητικών που θα πρέπει να ληφθούν από τον Cert Server (δ1).

Η αίτηση εγγραφής μπορεί να περιλαμβάνει τα παρακάτω στοιχεία:

- Ονοματεπώνυμο
- Αριθμός Ταυτότητας (θα χρησιμοποιηθεί ως παράμετρος στη συνάρτηση λήψης των πιστοποιητικών από τον CertServer)
- Υποβαλλόμενα Πιστοποιητικά:
  - <http://topo.auth.gr/certificates/ptyxio.xsd>
  - <http://topo.auth.gr/certificates/analitiki.xsd>
  - <http://www.britishcouncil.gr/certificates/profficiency.xsd>
  - <http://www.ydt.gr/certificates/id.xsd>
  - <http://www.company1.gr/apodeiktika/empeiria.xsd>
  - <http://www.company2.gr/apodeiktika/empeiria.xsd>

Οι παραπάνω διευθύνσεις (URL) αποτελούν τα XML Schema namespaces (βλ. παράγραφο 7.3) που προσδιορίζουν το κάθε πιστοποιητικό και όχι φυσικά κάποιες πραγματικές διευθύνσεις στο Internet. Εκτελώντας το Web Service λήψης πιστοποιητικών από τον CertServer και δίνοντας ως παραμέτρους αυτά τα URL's και τον αριθμό ταυτότητας του πολίτη, η ενδιαφερόμενη υπηρεσία - στην προκειμένη περίπτωση το Μεταπτυχιακό Πρόγραμμα – μπορεί να λάβει τα ψηφιακά υπογεγραμμένα XML, των οποίων την ύπαρξη εξασφάλισε ο αιτών.

# 11. Συμπεράσματα

## 11.1 Microsoft .NET και Sun J2EE

Δύσκολα θα μπορούσε κανείς να βρει δυσκολότερη εργασία από τη σύγκριση δυο πλατφόρμων ανάπτυξης όπως το Microsoft .NET και η Sun J2EE (Java 2 Enterprise Edition). Τουλάχιστον αν θα ήθελε το αποτέλεσμα της σύγκρισης να είναι αντικειμενικό, έστω σε μεγάλο βαθμό - αν όχι απόλυτα.

Η J2EE αποτελεί την πλατφόρμα ανάπτυξης εφαρμογών της Sun Microsystems, μεγάλου ανταγωνιστή της Microsoft. Πρόκειται για μια πλατφόρμα η οποία σε γενικές γραμμές φέρει όλα τα χαρακτηριστικά του .NET, ενώ και αυτή θεωρεί στυλοβάτη στην όλη φιλοσοφία της τα Web Services. Η δυσκολία στην σύγκριση μεταξύ των δυο συστημάτων οφείλεται κυρίως στο μέγεθός τους και στην πληθώρα των παραμέτρων που υπεισέρχονται σε μια τέτοια σύγκριση, μερικές από τις οποίες είναι δύσκολο να υπολογιστούν αντικειμενικά, για λόγους που περιγράφονται παρακάτω. Μερικές από τις παραμέτρους που θα μπορούσε κανείς να χρησιμοποιήσει είναι:

- η ταχύτητα εκτέλεσης απλού κώδικα (raw code),
- η ταχύτητα σχηματισμού δυναμικών σελίδων HTML,
- η καθυστέρηση στην εξυπηρέτηση μεγάλου αριθμού χρηστών,
- ο αριθμός γραμμών κώδικα που απαιτείται για αντίστοιχες εργασίες,
- η ταχύτητα πρόσβασης σε βάσεις δεδομένων,
- η πληθώρα των χαρακτηριστικών και των τεχνολογιών που καλύπτονται κ.α.

Τουλάχιστον σε ότι αφορά την τελευταία παράμετρο, η J2EE φέρει θεμελιώδεις τεχνολογίες που αναλύθηκαν σε αυτή την εργασία, όπως τα Web Services, η XML, η πρόσβαση σε βάσεις δεδομένων, ενώ υλοποιεί και άλλες, επίσης κοινές με το .NET, όπως η πιστοποίηση, η σχεδίαση γραφικών (2D/3D), η κρυπτογραφία, η παροχή βοήθειας στον χρήστη, τα multimedia και τεχνολογίες φωνής κ.α. Γενικά και οι δυο πλατφόρμες υλοποιούν όλες τις τελευταίες τεχνολογίες στο χώρο της πληροφορικής,

δίνονται και οι δυο έμφαση στο Internet, τα Web Services και την εφαρμογή των προτοτυποποιήσεων του W3C.

Χαρακτηριστικό παράδειγμα αποτελεί μια πρόσφατη προσπάθεια σύγκρισης των δυο όμοιων συστημάτων από τους ίδιους τους κατασκευαστές. Η ομάδα ανάπτυξης της πλατφόρμας της Java έχει δημιουργήσει ένα Web Site για λόγους επίδειξης με την ονομασία PetShop. Πρόκειται για ένα on-line σύστημα αγοράς κατοικίδιων και σχετικών με αυτά προϊόντων, για την υλοποίηση του οποίου χρησιμοποιήθηκαν διάφορες τεχνολογίες και προϊόντα όπως JSP, Java Beans, Servlets, Oracle. Ξεκινώντας ίσως ένα πόλεμο εντυπώσεων, μια αντίστοιχη ομάδα από την πλευρά της Microsoft ανέπτυξε το ίδιο Web Site χρησιμοποιώντας αντίστοιχες τεχνολογίες του .NET Framework (ASP.NET, .NET Assemblies, SQL Server). Μια τρίτη εταιρία, η Meddleware<sup>49</sup>, ανέλαβε την σύγκριση των δυο εφαρμογών, ως προς την απόδοση και την σχεδίασή τους.

Τα αποτελέσματα του συγκριτικού τεστ ήταν συντριπτικά υπέρ του .NET, σηκώνοντας θύελλα αντιδράσεων, όπως ήταν αναμενόμενο, από την κοινότητα των προγραμματιστών της Java. Η εταιρεία παραδέχτηκε ότι στο συγκριτικό τεστ δεν πήραν μέρος ειδικοί της Sun, ενώ ήταν παρόντες ειδικοί της Microsoft, αλλά προσπάθησε να αποκρούσει όλες τις κατηγορίες περί μεροληψίας υπέρ της τελευταίας. Το σχετικό μήνυμα-απάντηση παρατίθεται στο Παράρτημα 5. Ενδεικτικά αναφέρεται ότι ο αριθμός γραμμών κώδικα που μετρήθηκε στην εφαρμογή σε Java ήταν περίπου 14000, ενώ στην περίπτωση της εφαρμογής σε .NET ήταν περίπου 3500. Η Sun Microsystems έσπευσε να παρουσιάσει μια νέα έκδοση του PetShop, ανεπτυγμένη σε περιβάλλον Sun ASE<sup>50</sup>, η οποία περιείχε μόλις 224 γραμμές κώδικα! Ανάλογα ήταν και τα αποτελέσματα στο τομέα της ταχύτητας εκτέλεσης των δυο εφαρμογών και την καθυστέρηση εξυπηρέτησης μεγάλου αριθμού χρηστών. Τα αρχικά αποτελέσματα έδειξαν σαφή υπεροχή της εφαρμογής σε .NET, η οποία μπορούσε να εξυπηρετήσει

---

<sup>49</sup> <http://www.middleware-company.com>

<sup>50</sup> Το σύστημα ASE (<http://research.sun.com/projects/ace>) είναι ένα υπό ανάπτυξη έργο της Sun Microsystems το οποίο ενσωματώνει πολλές από τις τεχνολογίες που χρησιμοποιούνται για την ανάπτυξη ενός Web Site, μειώνοντας σημαντικά τον κώδικα που απαιτείται για την ανάπτυξή του.

ταυτόχρονα 4000 εικονικούς χρήστες με καθυστέρηση 20 milliseconds, ενώ η εφαρμογή σε Java με καθυστέρηση 69 milliseconds. Η συνέχεια ήρθε από την ίδια την εταιρεία που πραγματοποίησε τις μετρήσεις, η οποία ανακοίνωσε ότι επανασχεδιάζοντας της εφαρμογής της Java, κατάφερε να αυξήσει την ταχύτητά της κατά 17 φορές!

Σε κάθε περίπτωση, ακόμα και αν ήταν δυνατή η εξαγωγή μερικών αντικειμενικών δεικτών σύγκρισης, έναν για κάθε μια από τις παραπάνω παραμέτρους, το ερώτημα που αναδεικνύεται είναι αν κάτι τέτοιο θα μπορούσε να αποτελέσει κριτήριο για την επιλογή μεταξύ των δυο συστημάτων. Και αν ναι, σε ποιο βαθμό; Για να απαντήσει κανείς σε αυτό το ερώτημα, θα πρέπει να αντιπαραβάλλει άλλες παραμέτρους, καθαρά υποκειμενικές, όπως η διαθέσιμη αγορά εργασίας, η εταιρικές συνεργασίες και ο σημαντικότερος όλων, η εμπειρία και εξοικείωση με κάποιο από τα δυο συστήματα ή τους προκατόχους τους. Σταυροδρόμι για την επιλογή πλατφόρμας ανάπτυξης αποτελεί βέβαια και το περιβάλλον εκτέλεσης των υπό ανάπτυξη συστημάτων. Αν τα υπό ανάπτυξη συστήματα πρέπει να είναι ανεξάρτητα του περιβάλλοντος εκτέλεσης (platform independent), είναι σαφές ότι θα πρέπει να καταφύγουμε στη λύση της Java. Αν παρόλα αυτά, το περιβάλλον λειτουργίας θα είναι τα Windows, η λύση .NET θα ήταν σίγουρα περισσότερο αποδοτική, λόγω της καλύτερης συμβατότητας της πλατφόρμας .NET με αυτό το περιβάλλον.

## **11.2 Μετάβαση στο .NET Framework**

Όπως σε κάθε νέα πλατφόρμα ανάπτυξης λογισμικού, έτσι και στην περίπτωση του Microsoft .NET, πολλά από τα χαρακτηριστικά που αυτό ενσωματώνει, προέρχονται από την εμπειρία της ανάπτυξης, λειτουργίας και συντήρησης εφαρμογών που δημιουργήθηκαν με προγενέστερες πλατφόρμες. Απώτερος σκοπός είναι η εκμετάλλευση αυτής της εμπειρίας για τη δημιουργία του λογισμικού που θα καλύψει τις νέες ανάγκες που δημιουργούνται, οι οποίες χωρίς αμφιβολία είναι ολοένα και μεγαλύτερες. Πολλές συζητήσεις έχουν γίνει για το πότε θα ήταν η καλύτερη στιγμή για την μετάβαση ενός προγραμματιστή ή μιας εταιρείας παροχής λύσεων λογισμικού, στο Microsoft .NET. Η ίδια η Microsoft προσπαθεί να καθοδηγήσει τους πελάτες της με τον δικό της τρόπο, δημοσιεύοντας διάφορες μελέτες και προτάσεις γι αυτό το σκοπό.

Κάποιες από τις τεχνολογίες που φέρει το .NET Framework και υλοποιούνται με τη βοήθεια του Visual Studio είναι νέες τεχνολογίες που αναδύθηκαν από την ευρεία αποδοχή του Internet, η οποία σαφώς δεν υπήρχε στον ίδιο βαθμό την εποχή που κυκλοφόρησε η προηγούμενη έκδοση του Visual Studio (έκδοση 6, 1997). Για την απόδοση λύσεων στο χώρο των Web εφαρμογών και των Web Services, το .NET Framework είναι μάλλον μονόδρομος.

Άλλα εργαλεία και τεχνολογίες, όπως το ADO.NET, τα νέα χαρακτηριστικά της Visual Basic .NET, το XML Schema κ.α. φέρουν οπωσδήποτε ενδιαφέροντα χαρακτηριστικά, προσφέροντας πολλές νέες δυνατότητες. Το κρίσιμο ερώτημα είναι αν και πότε θα πρέπει κανείς να εγκαταλείψει την εμπειρία και την αποδοτικότητά του με κάποιο άλλο περιβάλλον ανάπτυξης λογισμικού και να μεταβεί στο .NET Framework. Πολλά από τα ζητήματα – προβλήματα που καλύπτει το τελευταίο, έχουν υπερβληθεί με διάφορους τρόπους, ορθόδοξους ή ανορθόδοξους, αρκετά ή λιγότερο αποδοτικούς, από προγραμματιστές. Για πολλές μάλιστα περιπτώσεις, έχει διαδοθεί πληθώρα λύσεων μέσω του Internet. Είναι λοιπόν εύλογο, ο χρόνος στον οποίο μπορεί να αναπτυχθεί μια εφαρμογή σε μια προγενέστερη πλατφόρμα να είναι σημαντικά μικρότερος, λόγω αυτής ακριβώς την εμπειρίας και τεχνογνωσίας σε αυτή.

Σε μια καινούρια όμως συνεργασία στα πλαίσια της ανάπτυξης ενός νέου συστήματος, η χρήση μεθόδων και τεχνικών για την κάλυψη παλαιότερων προβλημάτων, από διαφορετικούς προγραμματιστές, μάλλον θα δημιουργούσε ένα πύργο της Βαβέλ και όχι ένα σωστά σχεδιασμένο, εύκολα συντηρήσιμο και επεκτάσιμο πληροφοριακό σύστημα. Σε αυτή την περίπτωση, ο επιπλέον χρόνος που θα απαιτούνταν για την αφομοίωση και εκμετάλλευση του .NET Framework, μάλλον θα εξισορροπούσαν από την δημιουργία μιας καλύτερα οργανωμένης εφαρμογής, που κάνει χρήση των σύγχρονων προτοτυποποιήσεων στο χώρο της ανάπτυξης λογισμικού. Άλλωστε, η ονομασία “πλαίσιο εργασίας” είναι μάλλον το σημαντικότερο πλεονέκτημα του .NET Framework.

# Ευρετήριο

.NET Compact Framework.....	8
.NET Passport.....	60
.NET Server.....	74
Active Directory.....	60
ActiveX DLL.....	26
ADO.....	28
assemblies.....	13
assembly.....	18
attribute.....	50, 53
authentication.....	60
authentication tickets.....	40
authorization.....	60
C++ . 13, 15, 22, 26, 27, 61, 64, 77	
cardinality.....	51
Class Libraries.....	11, 12, 21
Class Library.....	16
CLR.....	58
COBOL.....	11
COM.....	12, 13, 26, 36, 54
COM Callable Wrapper.....	27
COM+.....	12
Common Language Runtime ... 11,	19
CORBA.....	36
Data Mining.....	29
data providers.....	28, 30
DataSet.....	29, 32, 57
DCOM.....	12, 36, 38
Delphi.....	27, 59, 61
e-government.....	81
element.....	50
Enterprise Manager.....	29
Εφορία.....	83
firewall.....	41
Foreign Key.....	33
Google.....	45
GUID.....	54
HTTP.....	41, 58
HTTP POST/GET.....	41, 59
IIS.....	42, 58, 77, 80
Inheritance.....	22
Intel x86.....	20
intermediate language.....	19
J#.....	11
J2EE.....	102
Java.....	103
Java RMI.....	36
JSP.....	59, 103
Just-In-Time Compiler.....	20
LDAP.....	77, 80
managed code.....	12, 13
metadata.....	33, 51
Microsoft Access.....	30
Microsoft BizTalk Server.....	8
Microsoft Exchange Server ...8,	77
moniker.....	55
MSDE.....	29
namespace.....	14, 18, 25, 53, 79
Netscape.....	77
Novell.....	77
Oracle.....	28, 31, 103
palmtop.....	11, 20
PHP.....	59
reference counting.....	25
runtime environment.....	11
SDK.....	11
Servers.....	8
SignedXML.....	82, 92
Smart Clients.....	8
SOAP.....	39
Solaris.....	34, 36
specification.....	51
SQL.....	30, 31, 54
SQL Server.....	28, 31, 82, 103
Stored Procedures.....	30
Style Sheets.....	60
Sun.....	34, 102, 103
Sun ASE.....	103
Sybase.....	28
tablet pc.....	20
TCP/IP.....	41
Threading.....	23
type library.....	27
UDDI.....	38, 45
Visual Basic 6.....	13, 15, 18, 26
W3C.....	35, 51, 52, 82, 92, 103
Web Services7, 11, 35, 58, 81,	102
Windows CE.....	11
Windows Domain.....	75
Windows NT.....	80
Windows NT.....	77
WSDL.....	38
XML Schema.....	39, 41, 82
Δημοτολόγιο.....	84
IKA.....	83
Τύπος Δεδομένων.....	51, 52
Ψηφιακές Υπογραφές40, 82, 83,	91

## Αναφορές

### Βιβλιογραφία – Αρθρογραφία

- [1] **Aaron Skonnard** - *Understanding XML Namespaces*, 2002
  - [2] **Aaron Skonnard** - *A Quick Guide to XML Schema*, 2002
  - [3] **Dino Esposito** – *XML Features in ADO.NET*, 2001
  - [4] **Duncan Mackenzie** - *Using System.DirectoryServices to Search the Active Directory*, 2002
  - [5] **Dulaney, Emmett – Sankar E., Sharon** – *Active Directory: An Overview*, 1999 – 29th Street Press.
  - [6] **Gordon Brown** – *Performance Optimization in Visual Basic .NET*, 2002
  - [7] **Middleware Company** - *Comparing Microsoft .NET Framework Performance and Scalability to J2EE Application Servers*, 2002
  - [8] **Microsoft Corp.** – *Active Directory Programmer's Guide*, 2000
  - [9] **Microsoft Corp** – *Application Architecture for .NET: Designing Applications and Services*, 2002
  - [10] **Microsoft Corp** – *Lowering Total Cost of Ownership with Active Directory-Enabled Applications*, 1998
  - [11] **Microsoft Press** – *Developing Web Applications with Microsoft Visual Basic .NET and Microsoft C# .NET*
  - [12] **Microsoft Development Network** - *The Transition from Visual Basic 6.0 to Visual Basic .NET*, 2002
  - [13] **Scott Short** – *Building XML Web Services for the Microsoft .NET Platform*, 2002
  - [14] **Sitaraman Lakshaminarayanan** - *XML Signatures in Microsoft .NET*, 2002
  - [15] **Sanders Kaufman, Jr.** – *The role of the .NET CLR in creating a global development framework*, 2002
  - [16] **Sanders Kaufman, Jr.** – *Under the covers of the .NET CLR*, 2002
-

- [17] **Spencer, Ken – Goncalves, Marcus** – *Installing and Configuring Active Directory*, 2000 – Prentice Hall PTR
- [18] **Diana Reichardt** - *A Field Guide to Services On Demand and Sun™ ONE*, Sun Microsystems, 2001
- [19] **Ted Pattison** – *Visual Basic NET – New Programming Model and Language Enhancements Boost Development Power*, 2001
- [20] **Γιάννης Ανδρουλάκης** - *.NET Framework*, Computer για Όλους 207, 2001
- [21] **Γιάννης Ηλιόπουλος, Γεώργιος Αθανασιάδης, Θανάσης Κοσμόπουλος** - *Ψηφιακή Υπογραφή*, Computer για Όλους 199, 2001

## Internet

- [22] <http://msdn.microsoft.com>  
MSDN Home: Η αρχική σελίδα του Microsoft Development Network, μέσω της οποίας μπορεί κανείς να έχει πρόσβαση σε όλα τα εργαλεία και τεχνολογίες ανάπτυξης της Microsoft.
- [23] <http://www.msdnaa.com>  
MSDN Academic Alliance: Κοινότητα της Microsoft για τις ανάγκες προγραμματιστών του ακαδημαϊκού χώρου και συγκεκριμένα για τα πεδία της Επιστήμης Υπολογιστών και των Πληροφοριακών Συστημάτων.
- [24] <http://www.asp.net>  
Επίσημη ιστοσελίδα της ASP.NET
- [25] <http://msdn.microsoft.com/webservices>  
Πληροφορίες για web services από το Microsoft Development Network
- [26] <http://www.gotdotnet.com>  
Κοινότητα ανάπτυξης με την πλατφόρμα του .NET. Περιβάλλον συνεργασίας για τη διαχείριση της ομαδικής ανάπτυξης εφαρμογών μέσω του Internet (Workspaces)
- [27] <http://www.google.com/microsoft.html>  
Αναζήτηση στους δικτυακούς τόπους της Microsoft μέσω της μηχανής αναζήτησης του Google.

# Παραρτήματα

## Παράρτημα 1 – Διαθέσιμα Elements στην γλώσσα XSD

Παρακάτω φαίνεται η λίστα των διαθέσιμων elements που απαρτίζουν την γλώσσα XSD.

Πηγή: .NET Framework Documentation

Tag	Περιγραφή
all	Allows the elements in the group to appear (or not appear) in any order in the containing element.
annotation	Defines an annotation.
any	Enables any element from the specified namespace(s) to appear in the containing sequence or choice element.
anyAttribute	Enables any attribute from the specified namespace(s) to appear in the containing complexType element or in the containing attributeGroup element.
appinfo	Specifies information to be used by applications within an annotation element.
attribute	Declares an attribute.
attributeGroup	Groups a set of attribute declarations so that they can be incorporated as a group for complex type definitions.
choice	Allows one and only one of the elements contained in the selected group to be present within the containing element.
complexContent	Contains extensions or restrictions on a complex type that contains mixed content or elements only.
complexType	Defines a complex type, which determines the set of attributes and the content of an element.
documentation	Specifies information to be read or used by users within an annotation element.
element	Declares an element.
extension (simpleContent)	Contains extensions on simpleContent. This extends a simple type or a complex type that has simple content.
extension (complexContent)	Contains extensions on complexContent.
field	Specifies an XML Path Language (XPath) expression that specifies the value (or one of the values) used to define an identity constraint (unique, key, and keyref elements).
group	Groups a set of element declarations so that they can be incorporated as a group into complex type definitions.
import	Identifies a namespace whose schema components are referenced by the containing schema.
include	Includes the specified schema document in the target namespace of the containing schema.
key	Specifies that an attribute or element value (or set of values) must be a key within the specified scope. The scope of a key is the containing element in an instance document. A key must be unique, non-nullable, and always present.
keyref	Specifies that an attribute or element value (or set of values) correspond to those of the specified key or unique element.
list	Defines a simpleType element as a list of values of a specified data type.
notation	Contains the definition of a notation to describe the format of non-XML

	data within an XML document. An XML Schema notation declaration is a reconstruction of XML 1.0 NOTATION declarations.
redefine	Allows simple and complex types, groups, and attribute groups that are obtained from external schema files to be redefined in the current schema.
restriction (simpleType)	Defines constraints on a simpleType definition.
restriction (simpleContent)	Defines constraints on a simpleContent definition.
restriction (complexContent)	Defines constraints on a complexContent definition.
schema	Contains the definition of a schema.
selector	Specifies an XPath expression that selects a set of elements for an identity constraint (unique, key, and keyref elements).
sequence	Requires the elements in the group to appear in the specified sequence within the containing element.
simpleContent	Contains extensions or restrictions on a complexType element with character data or a simpleType element as content and contains no elements.
simpleType	Defines a simple type, which determines the constraints on and information about the values of attributes or elements with text-only content.
union	Defines a simpleType element as a collection of values from specified simple data types.
unique	Specifies that an attribute or element value (or a combination of attribute or element values) must be unique within the specified scope. The value must be unique or nil.

---

## Παράρτημα 2 - XML Schema Data Types

Πηγή: Scott Short, Building XML Web Services for the Microsoft .NET Platform

Τύπος	Παράδειγμα	Περιγραφή
string	This is a string.	A sequence of legal XML 1 characters.
normalizedString	This is a normalized string.	A sequence of legal XML 1 characters that does not contain carriage returns, line feeds, or tabs.
token	Token1 Token2 Token3	A tokenized string of legal XML 1 characters that does not contain carriage returns, line feeds, or tabs.
byte	-128	A numeric value from -128 through 127.
unsignedByte	255	A numeric value from 0 through 255.
base64Binary	6e7P	Base64-encoded binary data. Base64 encoding is described in RFC 2045.
hexBinary	B2AF	Hex-encoded binary data. Each binary octet is encoded into its two-character hexadecimal equivalent. The example has a binary representation of 1011001010101111.
integer	123456789	A numeric value meeting the mathematical definition of an integer. Basically, an infinitely bounded number that can be positive or negative.
positiveInteger	123456789	A numeric value that meets the mathematical definition of a positive integer. Basically, an infinitely bounded positive number, not including zero.
negativeInteger	-123456789	A numeric value that meets the mathematical definition of a negative integer. Basically, an infinitely bounded negative number, not including zero.
nonNegativeInteger	123456789	A numeric value that meets the mathematical definition of a non-negative integer. Basically, an infinitely bounded positive number, including zero.
nonPositiveInteger	-123456789	A numeric value that meets the mathematical definition of a nonpositive integer. Basically, an infinitely bounded negative number, including zero.
int	-2147483648	A numeric value from -2147483648 through 2147483647.
unsignedInt	4294967295	A numeric value from 0 through 4294967295.
long	-9223372036854775808	A numeric value from -9223372036854775808 through 9223372036854775807.
unsignedLong	18446744073709551615	A numeric value from 0 through 18446744073709551615.
short	-32768	A numeric value from -32768 through 32767.
unsignedShort	65535	A numeric value from 0 through 65535.
decimal	1234.56789	A finite sequence of decimal digits that must contain a single period as a decimal indicator.
float	-123.456789E2, 123.456789e2, 12345.6789 or	A numeric value that meets the requirements of the IEEE single-precision 32-bit floating-point type. The legal special values include positive and negative zero (0, -0), positive and negative infinity (INF, -INF), and not a number (NaN).
double	-123.456789E2,	A numeric value that meets the requirements

	123.456789e2, 12345.6789	or	-of the IEEE single- precision 64-bit floating-point type. The legal special values include positive and negative zero (0, -0), positive and negative infinity (INF, -INF), and not a number (NaN).
boolean	1 or true		A value containing true or false.
time	01:22:15-07:00		A value containing a specific time of day in the format HH:MM:SS. Midnight is represented as 00:00:00. Time is represented using 24-hour notation. The time zone is indicated by the number of hours after Coordinated Universal Time. (Mountain Standard Time in the U.S. is represented as -07:00.)
date	2001-05-21		A value containing a calendar day in the format YYYY-MM-DD.
dateTime	2001-05-21T01:22:15-07:00		A value containing a specific instance of time in the format YYYY-MM-DDTHH:MM:SS.
duration	P3Y1M24DT11H22M10.4S		A value containing a duration of time in the format P#Y#M#DT#H#M#S or any subset, such as P#M#S.
gMonth	--05--		A value containing a Gregorian month in the format --MM--.
gYear	2001		A value containing a Gregorian year in the format YYYY.
gYearMonth	2001-05		A value containing a Gregorian month in a particular year in the format YYYY-MM.
gDay	---21		A value containing a recurring Gregorian day of the month in the format ---DD.
gMonthDay	--05-21		A value containing a recurring Gregorian day of the month in the format --MM-DD.
Name	Address		A value containing an XML 1 name.
QName	po:Address		A value containing a qualified XML 1 name.
NCName	Address		A value containing a "noncolonized" XML 1 name (a QName without the prefix).
anyURI	http://www.microsoft.com		A value containing a URI. The URI can be relative or absolute.
language	en-US		A value containing a natural language identifier as defined by RFC 1766.
ID			A value containing an XML 1 ID attribute type.
IDREF			A value containing an XML 1 IDREF attribute type.
IDREFS			A value containing an XML 1 IDREFS attribute type.
ENTITY			A value containing an XML 1 ENTITY attribute type.
ENTITIES			A value containing an XML 1 ENTITIES attribute type.
NOTATION			A value containing an XML 1 NOTATION attribute type.
NMTOKEN			A value containing an XML 1 NMTOKEN attribute type.
NMTOKENS			A value containing an XML 1 NMTOKENS attribute type.

## Παράρτημα 3

### A) WSDL Έγγραφο των Google APIs

Πηγή: GoogleSearch Web Service

```
<?xml version="1.0"?>

<!-- WSDL description of the Google Web APIs.
      The Google Web APIs are in beta release. All interfaces are subject to
      change as we refine and extend our APIs. Please see the terms of use
      for more information. -->

<!-- Revision 2002-08-16 -->

<definitions name="GoogleSearch"
      targetNamespace="urn:GoogleSearch"
      xmlns:typens="urn:GoogleSearch"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
      xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/"
      xmlns="http://schemas.xmlsoap.org/wsdl/">

  <!-- Types for search - result elements, directory categories -->

  <types>
    <xsd:schema xmlns="http://www.w3.org/2001/XMLSchema"
      targetNamespace="urn:GoogleSearch">
      <xsd:complexType name="GoogleSearchResult">
        <xsd:all>
          <xsd:element name="documentFiltering" type="xsd:boolean"/>
          <xsd:element name="searchComments" type="xsd:string"/>
          <xsd:element name="estimatedTotalResultsCount" type="xsd:int"/>
          <xsd:element name="estimateIsExact" type="xsd:boolean"/>
          <xsd:element name="resultElements"
            type="typens:ResultElementArray"/>
          <xsd:element name="searchQuery" type="xsd:string"/>
          <xsd:element name="startIndex" type="xsd:int"/>
          <xsd:element name="endIndex" type="xsd:int"/>
          <xsd:element name="searchTips" type="xsd:string"/>
          <xsd:element name="directoryCategories"
            type="typens:DirectoryCategoryArray"/>
          <xsd:element name="searchTime" type="xsd:double"/>
        </xsd:all>
      </xsd:complexType>
      <xsd:complexType name="ResultElement">
        <xsd:all>
          <xsd:element name="summary" type="xsd:string"/>
          <xsd:element name="URL" type="xsd:string"/>
          <xsd:element name="snippet" type="xsd:string"/>
          <xsd:element name="title" type="xsd:string"/>
          <xsd:element name="cachedSize" type="xsd:string"/>
          <xsd:element name="relatedInformationPresent" type="xsd:boolean"/>
          <xsd:element name="hostName" type="xsd:string"/>
          <xsd:element name="directoryCategory"
            type="typens:DirectoryCategory"/>
          <xsd:element name="directoryTitle" type="xsd:string"/>
        </xsd:all>
      </xsd:complexType>
      <xsd:complexType name="ResultElementArray">
        <xsd:complexContent>
          <xsd:restriction base="soapenc:Array">
            <xsd:attribute ref="soapenc:arrayType"
              wsdl:arrayType="typens:ResultElement[]" />
          </xsd:restriction>
        </xsd:complexContent>
      </xsd:complexType>
      <xsd:complexType name="DirectoryCategoryArray">
        <xsd:complexContent>
          <xsd:restriction base="soapenc:Array">

```

```

        <xsd:attribute ref="soapenc:arrayType"
wsdl:arrayType="typens:DirectoryCategory[]"/>
    </xsd:restriction>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="DirectoryCategory">
    <xsd:all>
        <xsd:element name="fullViewableName" type="xsd:string"/>
        <xsd:element name="specialEncoding" type="xsd:string"/>
    </xsd:all>
</xsd:complexType>
</xsd:schema>
</types>

<!-- Messages for Google Web APIs - cached page, search, spelling. -->
<message name="doGetCachedPage">
    <part name="key" type="xsd:string"/>
    <part name="url" type="xsd:string"/>
</message>
<message name="doGetCachedPageResponse">
    <part name="return" type="xsd:base64Binary"/>
</message>
<message name="doSpellingSuggestion">
    <part name="key" type="xsd:string"/>
    <part name="phrase" type="xsd:string"/>
</message>
<message name="doSpellingSuggestionResponse">
    <part name="return" type="xsd:string"/>
</message>

<!-- note, ie and oe are ignored by server; all traffic is UTF-8. -->
<message name="doGoogleSearch">
    <part name="key" type="xsd:string"/>
    <part name="q" type="xsd:string"/>
    <part name="start" type="xsd:int"/>
    <part name="maxResults" type="xsd:int"/>
    <part name="filter" type="xsd:boolean"/>
    <part name="restrict" type="xsd:string"/>
    <part name="safeSearch" type="xsd:boolean"/>
    <part name="lr" type="xsd:string"/>
    <part name="ie" type="xsd:string"/>
    <part name="oe" type="xsd:string"/>
</message>

<message name="doGoogleSearchResponse">
    <part name="return" type="typens:GoogleSearchResult"/>
</message>

<!-- Port for Google Web APIs, "GoogleSearch" -->
<portType name="GoogleSearchPort">
    <operation name="doGetCachedPage">
        <input message="typens:doGetCachedPage"/>
        <output message="typens:doGetCachedPageResponse"/>
    </operation>
    <operation name="doSpellingSuggestion">
        <input message="typens:doSpellingSuggestion"/>
        <output message="typens:doSpellingSuggestionResponse"/>
    </operation>
    <operation name="doGoogleSearch">
        <input message="typens:doGoogleSearch"/>
        <output message="typens:doGoogleSearchResponse"/>
    </operation>
</portType>

<!-- Binding for Google Web APIs - RPC, SOAP_over HTTP -->
<binding name="GoogleSearchBinding" type="typens:GoogleSearchPort">
    <soap:binding style="rpc"
        transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="doGetCachedPage">
        <soap:operation soapAction="urn:GoogleSearchAction"/>
        <input>
            <soap:body use="encoded"
                namespace="urn:GoogleSearch"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>

```

```

    </input>
  </output>
  <soap:body use="encoded"
    namespace="urn:GoogleSearch"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</output>
</operation>
<operation name="doSpellingSuggestion">
  <soap:operation soapAction="urn:GoogleSearchAction" />
  <input>
    <soap:body use="encoded"
      namespace="urn:GoogleSearch"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </input>
  <output>
    <soap:body use="encoded"
      namespace="urn:GoogleSearch"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </output>
</operation>
<operation name="doGoogleSearch">
  <soap:operation soapAction="urn:GoogleSearchAction" />
  <input>
    <soap:body use="encoded"
      namespace="urn:GoogleSearch"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </input>
  <output>
    <soap:body use="encoded"
      namespace="urn:GoogleSearch"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </output>
</operation>
</binding>

<!-- Endpoint for Google Web APIs -->
<service name="GoogleSearchService">
  <port name="GoogleSearchPort" binding="typens:GoogleSearchBinding">
    <soap:address location="http://api.google.com/search/beta2" />
  </port>
</service>
</definitions>

```

## B) Αντικειμενοστραφές Μοντέλο του Google Web Service



## Παράρτημα 4

### WSDL έγγραφο των Web Services του CertServer

```
<?xml version="1.0" encoding="utf-8" ?>
<definitions xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:s0="http://dipapadi.info/uomthesis/webservice"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
targetNamespace="http://dipapadi.info/uomthesis/webservice"
xmlns="http://schemas.xmlsoap.org/wsdl/"
  <types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://dipapadi.info/uomthesis/webservice">
      <s:import namespace="http://www.w3.org/2001/XMLSchema"/>
      <s:element name="GetCertificates">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1"
maxOccurs="1" name="authorityID"
type="s:int"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetCertificatesResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0"
maxOccurs="1"
name="GetCertificatesResult">
              <s:complexType>
                <s:sequence>
                  <s:element ref="s:schema"/>
                </s:sequence>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetAuthorities">
        <s:complexType/>
      </s:element>
      <s:element name="GetAuthoritiesResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0"
maxOccurs="1"
name="GetAuthoritiesResult">
              <s:complexType>
                <s:sequence>
                  <s:element ref="s:schema"/>
                </s:sequence>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetCitizenPublicKey">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="cIDNum"
type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetCitizenPublicKeyResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="GetCitizenPublicKeyResult" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="InsertCertificate">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1"
name="authorityID" type="s:int"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:schema>
  </types>

```

```

        name="certificateID" type="s:int"/>
        <s:element minOccurs="0" maxOccurs="1"
        name="cIDNum" type="s:string"/>
        <s:element minOccurs="0" maxOccurs="1"
        name="signedCertificate" type="s:string"/>
        <s:element minOccurs="1" maxOccurs="1"
        name="ccIsPrivate" type="s:boolean"/>
    </s:sequence>
</s:complexType>
</s:element>
<s:element name="InsertCertificateResponse">
    <s:complexType/>
</s:element>
<s:element name="DataSet" nillable="true">
    <s:complexType>
        <s:sequence>
            <s:element ref="s:schema"/>
            <s:any/>
        </s:sequence>
    </s:complexType>
</s:element>
<s:element name="string" nillable="true" type="s:string"/>
</s:schema>
</types>
<message name="GetCertificatesSoapIn">
    <part name="parameters" element="s0:GetCertificates"/>
</message>
<message name="GetCertificatesSoapOut">
    <part name="parameters" element="s0:GetCertificatesResponse"/>
</message>
<message name="GetAuthoritiesSoapIn">
    <part name="parameters" element="s0:GetAuthorities"/>
</message>
<message name="GetAuthoritiesSoapOut">
    <part name="parameters" element="s0:GetAuthoritiesResponse"/>
</message>
<message name="GetCitizenPublicKeySoapIn">
    <part name="parameters" element="s0:GetCitizenPublicKey"/>
</message>
<message name="GetCitizenPublicKeySoapOut">
    <part name="parameters" element="s0:GetCitizenPublicKeyResponse"/>
</message>
<message name="InsertCertificateSoapIn">
    <part name="parameters" element="s0:InsertCertificate"/>
</message>
<message name="InsertCertificateSoapOut">
    <part name="parameters" element="s0:InsertCertificateResponse"/>
</message>
<message name="GetCertificatesHttpGetIn">
    <part name="authorityID" type="s:string"/>
</message>
<message name="GetCertificatesHttpGetOut">
    <part name="Body" element="s0:DataSet"/>
</message>
<message name="GetAuthoritiesHttpGetIn"/>
<message name="GetAuthoritiesHttpGetOut">
    <part name="Body" element="s0:DataSet"/>
</message>
<message name="GetCitizenPublicKeyHttpGetIn">
    <part name="cIDNum" type="s:string"/>
</message>
<message name="GetCitizenPublicKeyHttpGetOut">
    <part name="Body" element="s0:string"/>
</message>
<message name="InsertCertificateHttpGetIn">
    <part name="certificateID" type="s:string"/>
    <part name="cIDNum" type="s:string"/>
    <part name="signedCertificate" type="s:string"/>
    <part name="ccIsPrivate" type="s:string"/>
</message>
<message name="InsertCertificateHttpGetOut"/>
<message name="GetCertificatesHttpPostIn">
    <part name="authorityID" type="s:string"/>
</message>
<message name="GetCertificatesHttpPostOut">
    <part name="Body" element="s0:DataSet"/>
</message>
<message name="GetAuthoritiesHttpPostIn"/>
<message name="GetAuthoritiesHttpPostOut">
    <part name="Body" element="s0:DataSet"/>
</message>
<message name="GetCitizenPublicKeyHttpPostIn">
    <part name="cIDNum" type="s:string"/>
</message>
<message name="GetCitizenPublicKeyHttpPostOut">

```

```

        <part name="Body" element="s0:string"/>
    </message>
    <message name="InsertCertificateHttpPostIn">
        <part name="certificateID" type="s:string"/>
        <part name="cIDNum" type="s:string"/>
        <part name="signedCertificate" type="s:string"/>
        <part name="ccIsPrivate" type="s:string"/>
    </message>
    <message name="InsertCertificateHttpPostOut"/>
    <portType name="serviceSoap">
        <operation name="GetCertificates">
            <input message="s0:GetCertificatesSoapIn"/>
            <output message="s0:GetCertificatesSoapOut"/>
        </operation>
        <operation name="GetAuthorities">
            <input message="s0:GetAuthoritiesSoapIn"/>
            <output message="s0:GetAuthoritiesSoapOut"/>
        </operation>
        <operation name="GetCitizenPublicKey">
            <input message="s0:GetCitizenPublicKeySoapIn"/>
            <output message="s0:GetCitizenPublicKeySoapOut"/>
        </operation>
        <operation name="InsertCertificate">
            <input message="s0:InsertCertificateSoapIn"/>
            <output message="s0:InsertCertificateSoapOut"/>
        </operation>
    </portType>
    <portType name="serviceHttpGet">
        <operation name="GetCertificates">
            <input message="s0:GetCertificatesHttpGetIn"/>
            <output message="s0:GetCertificatesHttpGetOut"/>
        </operation>
        <operation name="GetAuthorities">
            <input message="s0:GetAuthoritiesHttpGetIn"/>
            <output message="s0:GetAuthoritiesHttpGetOut"/>
        </operation>
        <operation name="GetCitizenPublicKey">
            <input message="s0:GetCitizenPublicKeyHttpGetIn"/>
            <output message="s0:GetCitizenPublicKeyHttpGetOut"/>
        </operation>
        <operation name="InsertCertificate">
            <input message="s0:InsertCertificateHttpGetIn"/>
            <output message="s0:InsertCertificateHttpGetOut"/>
        </operation>
    </portType>
    <portType name="serviceHttpPost">
        <operation name="GetCertificates">
            <input message="s0:GetCertificatesHttpPostIn"/>
            <output message="s0:GetCertificatesHttpPostOut"/>
        </operation>
        <operation name="GetAuthorities">
            <input message="s0:GetAuthoritiesHttpPostIn"/>
            <output message="s0:GetAuthoritiesHttpPostOut"/>
        </operation>
        <operation name="GetCitizenPublicKey">
            <input message="s0:GetCitizenPublicKeyHttpPostIn"/>
            <output message="s0:GetCitizenPublicKeyHttpPostOut"/>
        </operation>
        <operation name="InsertCertificate">
            <input message="s0:InsertCertificateHttpPostIn"/>
            <output message="s0:InsertCertificateHttpPostOut"/>
        </operation>
    </portType>
    <binding name="serviceSoap" type="s0:serviceSoap">
        <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
        <operation name="GetCertificates">
            <soap:operation
soapAction="http://dipapadi.info/uomthesis/webservice/GetCertificates"
style="document"/>
            <input><soap:body use="literal"/></input>
            <output><soap:body use="literal"/></output>
        </operation>
        <operation name="GetAuthorities">
            <soap:operation
soapAction="http://dipapadi.info/uomthesis/webservice/GetAuthorities"
style="document"/>
            <input><soap:body use="literal"/></input>
            <output><soap:body use="literal"/></output>
        </operation>
        <operation name="GetCitizenPublicKey">
            <soap:operation

```

```

soapAction="http://dipapadi.info/uomthesis/webservice/GetCitizenPublicKey"
        style="document"/>
        <input><soap:body use="literal"/></input>
        <output><soap:body use="literal"/></output>
    </operation>
    <operation name="InsertCertificate">
        <soap:operation
soapAction="http://dipapadi.info/uomthesis/webservice/InsertCertificate"
        style="document"/>
        <input><soap:body use="literal"/></input>
        <output><soap:body use="literal"/></output>
    </operation>
</binding>
<binding name="serviceHttpGet" type="s0:serviceHttpGet">
    <http:binding verb="GET"/>
    <operation name="GetCertificates">
        <http:operation location="/GetCertificates"/>
        <input><http:urlEncoded/></input>
        <output><mime:mimeXml part="Body"/></output>
    </operation>
    <operation name="GetAuthorities">
        <http:operation location="/GetAuthorities"/>
        <input><http:urlEncoded/></input>
        <output><mime:mimeXml part="Body"/></output>
    </operation>
    <operation name="GetCitizenPublicKey">
        <http:operation location="/GetCitizenPublicKey"/>
        <input><http:urlEncoded/></input>
        <output><mime:mimeXml part="Body"/></output>
    </operation>
    <operation name="InsertCertificate">
        <http:operation location="/InsertCertificate"/>
        <input><http:urlEncoded/></input>
        <output/>
    </operation>
</binding>
<binding name="serviceHttpPost" type="s0:serviceHttpPost">
    <http:binding verb="POST"/>
    <operation name="GetCertificates">
        <http:operation location="/GetCertificates"/>
        <input><mime:content type="application/x-www-form-
urlencoded"/></input>
        <output><mime:mimeXml part="Body"/></output>
    </operation>
    <operation name="GetAuthorities">
        <http:operation location="/GetAuthorities"/>
        <input><mime:content type="application/x-www-form-
urlencoded"/></input>
        <output><mime:mimeXml part="Body"/></output>
    </operation>
    <operation name="GetCitizenPublicKey">
        <http:operation location="/GetCitizenPublicKey"/>
        <input><mime:content type="application/x-www-form-
urlencoded"/></input>
        <output><mime:mimeXml part="Body"/></output>
    </operation>
    <operation name="InsertCertificate">
        <http:operation location="/InsertCertificate"/>
        <input><mime:content type="application/x-www-form-
urlencoded"/></input>
        <output/>
    </operation>
</binding>
<service name="service">
    <port name="serviceSoap" binding="s0:serviceSoap">
        <soap:address
location="http://localhost/certserver/service.asmx"/>
    </port>
    <port name="serviceHttpGet" binding="s0:serviceHttpGet">
        <http:address
location="http://localhost/certserver/service.asmx"/>
    </port>
    <port name="serviceHttpPost" binding="s0:serviceHttpPost">
        <http:address
location="http://localhost/certserver/service.asmx"/>
    </port>
</service>
</definitions>

```

## Παράρτημα 5

### Μήνυμα προς τις κοινότητες της J2EE και του .NET από την Middleware

Πηγή: <http://www.middleware-company.com>

*A message to the J2EE and .NET Communities*

November 8th, 2002

The feedback we have received about the J2EE verses .NET benchmark we posted was overwhelming to say the least. We would like to thank those who have spent the time to review our benchmark, and have provided extensive feedback on the benchmark.

In response to the feedback we would like to make a few public statements about the benchmark:

Our original intention for this benchmark was to put forth a more realistic comparison of J2EE and .NET to replace the original Pet Store comparison that Microsoft published. The TMC team worked to make extensive optimisations to Sun's implementation, which resulted in a 17x performance increase. Further information about the technical decisions we made in this project are available in our report as well as our FAQ.

However, since producing the report, we received valid feedback about additional J2EE optimisations that could be performed. We also received suggestions about using alternative J2EE architectures than the architecture we chose to test (for example, using CMP instead of BMP, or using no EJB at all).

We have reviewed this feedback and have concluded that testing these suggestions are important and necessary to be fair to J2EE. The alternative architectures in particular may show materially different performance results. We have learned a very costly lesson here -- that even though a great deal of time, effort, and care went into this benchmark, it is important to give a complete view of how J2EE and .NET compare for the J2EE community to accept the results.

We also admit to having made an error in process judgment by inviting Microsoft to participate in this benchmark, but not inviting the J2EE vendors to participate in this benchmark. We now realize that following this procedure is critical for the J2EE community to accept benchmark results and perceive them as credible

Because of the above issues, definitive conclusions about how J2EE verses .NET compares cannot be drawn unless a 2nd test is conducted which takes the above into account.

We do not currently have a public position yet about whether a round 2 will be conducted, especially given the public beating we just took. We have heard everything from accusing us of having sold our opinion to Microsoft, to that our parent company, Precise Software, is a strategic partner of Microsoft and that somehow tainted the results.

The above accusations are far from the truth, and we are disappointed that this has turned so ugly. We have done a lot of hard work for the J2EE community over the past few years, building TheServerSide.com community, writing the leading books on the technology, and much more. In fact, the reason we started this company was to help customers succeed. That is what gets us excited about our work. We tried to make J2EE perform well to show as realistic a comparison as possible, because we knew that one day we would need to post the results. The irony is that the perception right now is completely the opposite -- that this was a rush job, and that we sold our opinion to Microsoft as a marketing gimmick. This really hurts.

Going forward, we have an important decision to make. Despite our wounds, as engineers, a fair and complete performance comparison of J2EE verses .NET remains extremely interesting to us. A round 2 held in a fair and equitable fashion would be useful to get the facts out there about how J2EE and .NET compare, so people can make informed decisions. It would also be a chance to see how different J2EE architectures compare, such as BMP verses CMP, and EJB verses no EJB. We are willing to completely recode Pet Store with modern design patterns if necessary to achieve this, so that it is more benchmark-worthy -- something we did not have time to do in round 1.

A 2nd round test would fully involve vendors from both the Microsoft side and the J2EE side. It would also involve you -- the J2EE community at large. The process would be public, open, and auditable. The community would have the chance to review code as it is developed, and provide feedback to ensure J2EE is well represented. We would use many of the common performance and architectural suggestions received based on the first study.

We have not made a final decision about whether to hold a round 2 benchmark, since this is a very controversial subject, with many faces. We are interested in hearing your opinion on the matter, to help us make the best decision for the community. The two questions we have for you are:

A) Do you think we should conduct a 2nd benchmark?

B) How do you think that benchmark should differ from the current benchmark?