

Web services σε ASP.NET

Εισαγωγικά

Η δουλειά που είχαμε ήταν να κάνουμε develop και deploy τα web services που μας δόθηκαν, να υλοποιήσουμε τους αντίστοιχους clients και τέλος να κάνουμε consume τα services μέσα από την πλατφόρμα του gridlabd. Δεδομένου του ότι εργαζόμασταν σε Visual Studio και ASP.NET, αντιμετωπίσαμε κάποιες επιπλέον δυσκολίες από όσους εργάστηκαν με το εργαλείο staff για το οποίο υπήρχαν αναλυτικές οδηγίες και σειρά βοηθητικών βίντεο από την βοηθό του μαθήματος Αντωνία Νασιάκου.

Αρχικά, μετά από αρκετή αναζήτηση στο διαδίκτυο φτάσαμε στο συμπέρασμα πως δεν μπορούμε να αναπτύξουμε ASP.NET web services στο Visual Studio 2010/2013 τα οποία υποστηρίζουν πλέον μόνο WCF και C#. Η τελευταία έκδοση του Visual Studio που μπορούσε να χρησιμοποιηθεί καλύπτοντας της ανάγκες μας ήταν το Visual Studio 2005. Χρειάστηκε αρκετός χρόνος και χώρος στον δίσκο μέχρι να καταφέρουμε να κάνουμε το συγκεκριμένο, 10 χρονών IDE να τρέξει σωστά.

Development

Το development των web services προχωρούσε κανονικά μέχρι που διαπιστώσαμε πως δεν μπορούμε να χρησιμοποιήσουμε το complex.h αρχείο που μας έστειλε η Αντωνία και το οποίο είναι αυτό που χρησιμοποιείται σε όλα τα projects του gridlabd. Σαν μια πρώτη προσέγγιση προσπαθήσαμε να αλλάξουμε τον κώδικα των web services με τρόπο τέτοιο ώστε να έχει την ίδια λειτουργικότητα με τον αρχικό αλλά να χρησιμοποιεί το complex.h αρχείο που βρίσκεται στις βιβλιοθήκες του visual studio. Αυτό απαιτούσε να αλλάξουμε τόσο τον τρόπο με τον οποίον δηλώνονται οι μιγαδικοί αριθμοί όσο και να υλοποιήσουμε δικιές μας συναρτήσεις που να πάρουν την θέση κάποιων συναρτήσεων που υπήρχαν στο complex.h αρχείο της Αντωνίας μα έλειπαν από το δικό μας καταλήγοντας συχνά το μέγεθος ενός service να είναι τελικά διπλάσιο από τον αρχικό κώδικα.

Εν παραλλήλῳ, προσπαθούσαμε να βρούμε τρόπο ὥστε να χρησιμοποιήσουμε το complex.h ἀρχείο που μας δόθηκε ἀρχικά, διότι γνωρίζαμε πως αὐτό θα πρέπει να χρησιμοποιηθεῖ στην συνέχεια ὅταν θα πρέπει να γίνει το consume ενός service μέσω gridlabd και πιθανόν να βρεθούμε μπροστά σε νέα conflicts. Ἐστω και την τελευταία στιγμή ἀνακαλύψαμε πως ἔχουμε να κάνουμε με ἓνα confirmed bug του visual studio το οποίο μπορείτε να δείτε [εἰς ἄκρον](#). Λύνοντας το bug αὐτό μπορούσαμε πλέον να κάνουμε include και να χρησιμοποιήσουμε το complex.h ἀρχείο που μας δόθηκε ἀρχικά. Παρότι εἶχαμε ἤδη τελειώσει με το develop και των πέντε services αποφασίσαμε να τα κάνουμε ἀπὸ την ἀρχή για να είμαστε σίγουροι πως δεν θα βρεθούμε προ νέων προβλημάτων στην συνέχεια.

Deployment

Ἡ διαδικασία του deployment ενός ASP.NET εἶναι ἐξαιρετικά ἀπλή. Ἐχοντας τελειώσει το development ενός service, ἀρκεί κανεῖς να κάνει ἓνα καινούριο project του τύπου “web setup” και να συνδέσει τα δύο projects μεταξύ τους. Ἐτσι παράγεται ἓνα setup.exe ἀρχείο το οποίο μετακινούμε στον server μας, το εκτελούμε και εγκαθίσταται το service. Να σημειωθεῖ πως τα ASP.NET web services μπορούν να «κάθονται» μόνο σε windows server για αὐτό ζητήσαμε και πήραμε ἀπὸ την τεχνική υποστήριξη ἓνα Virtual Machine που τρέχει windows server 2008. Μια ἄλλη λύση θα ἦταν να χρησιμοποιήσουμε το project [MONO](#) και τότε θα μπορούσε να «κάθεται» σε οποιονδήποτε server.

Clients

Ἐχοντας ἓνα service να τρέχει στον server εἶναι ἀρκετά ἀπλό να υλοποιηθεῖ ο client μέσω Visual Studio 2005. Χρησιμοποιώντας το wsdl ἀρχείο του service δημιουργεῖται αὐτόματα ο client proxy και ἀπὸ κει και πέρα πρέπει ἀπλά να γράψουμε τον κώδικα ὥστε να δημιουργήσουμε ἓνα proxy ἀντικείμενο, να κάνουμε invoke το service, να ἀρχικοποιήσουμε τις μεταβλητές, να καλέσουμε τις μεθόδους, να χρησιμοποιήσουμε κατάλληλα τις τιμές που μας επιστρέφουν κτλ.

Οι clients υλοποιήθηκαν κι αὐτοί σε visual studio 2005 καθὼς στις νεότερες ἐκδόσεις δεν υποστηρίζεται πια το “add web reference” για C++ , δηλαδή ἡ διαδικασία με την οποία βρίσκει το wsdl ἀρχείο και κατασκευάζει αὐτόματα τον client proxy.

Consume

Σε αὐτό το σημείο μπορούσαμε πλέον να εργαστούμε σε ἓνα σύγχρονο IDE και χρησιμοποιήσαμε το Visual Studio 2010. Δουλέψαμε τα projects residential, generators και market στα οποία προσθέσαμε τα ἀπαραίτητα .h και .cpp ἀρχεία ἀπὸ τον κάθε client. Χρησιμοποιήσαμε τον κώδικα που μας ἔστειλε ἡ Ἀντωνία για την ἀρχικοποίηση των μεταβλητῶν και την κλήση των μεθόδων, με κάποιες διαφοροποιήσεις και πάλι μιας και εμεῖς δεν χρησιμοποιήσαμε staff και ὀρισμένα πράγματα ἔπρεπε να ἀλλάξουν. Παρατηρήθηκε μια ἰδιάζουσα συμπεριφορά του compiler σε σχέση με την χρησιμοποίηση ἢ μὴ του common language runtime ἡ οποία ἀντιμετωπίστηκε και συμβουλές δόθηκαν στην Ἀντωνία ὥστε να μπορεῖ με εὔκολο τρόπο να ξεπερνᾷ αὐτό το πρόβλημα ἀν της ἐμφανιστεῖ.

Το κάθε project παράγαγε κάποιο .dll αρχείο το οποίο κάθε φορά μεταφερόταν στον φάκελο με τα .glm αρχεία ώστε να γίνει consume το κάθε service.

Αντί επιλόγου

Η Microsoft έχει πλέον σταματήσει την υποστήριξη στα ASP.NET web services και την θέση τους έχει πάρει το **Windows Communication Foundation** το οποίο ενσωματώνει όλες τις λειτουργίες που παλιότερα πρόσφερε το ASP.NET και προσφέρει και πολλά περισσότερα. Ανάμεσα σε άλλους λόγους το WFC συνιστάται πλέον γιατί προσφέρει δύο πολύ σημαντικά πλεονεκτήματα έναντι του κλασσικού ASP.NET. Αυτά είναι η επιπλέον ασφάλεια που προσφέρει (πράγμα πολύ επιθυμητό στην περίπτωσή μας) και δεύτερον και σημαντικότερο την σειριοποίηση που προσφέρει η οποία μας βελτιώνει κατά πολύ την επίδοση.

Παρ' όλα αυτά τα πολύ θετικά σημεία, δεν ήταν δυνατόν να χρησιμοποιήσουμε αυτήν την τεχνολογία στην περίπτωσή μας γιατί το WFC "δουλεύει" μόνο με VB και C#. Αυτό μας δημιουργεί μεγάλο θέμα αφού το project μας στηρίζεται στην C++. Μια λύση θα ήταν να χρησιμοποιήσουμε έναν C++ to C# converter, κάτι το οποίο δοκιμάστηκε και απέτυχε λόγω της πληθώρας βιβλιοθηκών και αναφορών που είχαμε.

Χρήσιμοι σύνδεσμοι

- <https://msdn.microsoft.com/en-us/library/a86z84tw%28v=vs.80%29.aspx>
- <https://msdn.microsoft.com/en-US/library/14hykb68%28v=vs.80%29.aspx>
- <https://msdn.microsoft.com/en-us/library/ms123401.aspx>
- <http://stackoverflow.com/>