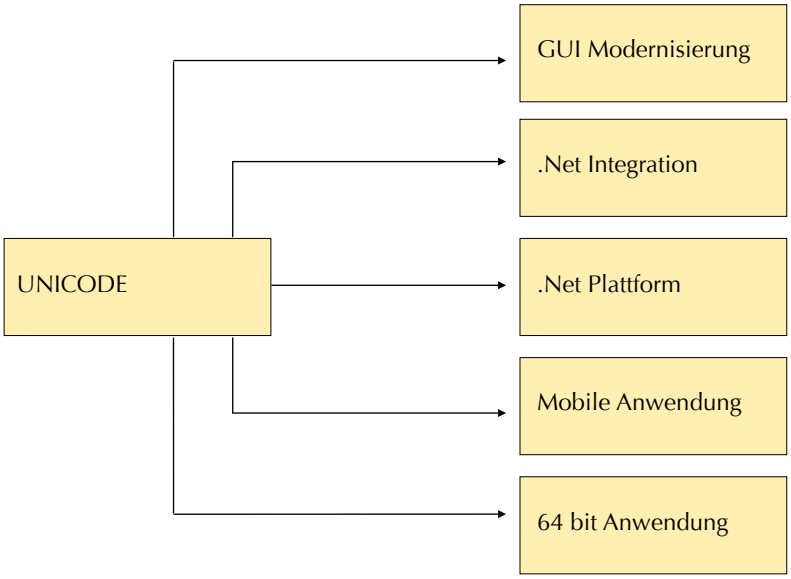
Sie haben Ihren Quellcode erfolgreich in die aktuelle Version 6.3 von Team Developer übernommen. Die Codebasis ist nunmehr UNICODE und Sie haben erfolgreich eine neue Basis für ihr Softwareprojekt geschaffen.

Andere Kunden haben aber weitere Ziele, wenn sie den Quellcode in die neueste Team-Developer-Version übernehmen: Sie möchten ihre bestehende Client-Server-Anwendung „modernisieren“, wobei sich hinter diesem einen Wort die unterschiedlichsten Wünsche und Vorstellungen verbergen können.

Modernisierung der Bedienoberfläche und funktionale Erweiterungen

Abbildung 1: Schematische Darstellung unterschiedlicher Modernisierungsoptionen

Die Abbildung zeigt schematisiert, welche Technologien ab dem Moment genutzt werden können, wenn die Quellcodebasis eines Team-Developer-Projekts erfolgreich auf UNICODE umgestellt wurde.



Modernisierung der GUI

Unter dem Begriff „Modernisierung“ kann zunächst verstanden werden, die bestehende Anwendung so anzupassen, dass das Erscheinungsbild der Anwendung dem Standard, beispielsweise der aktuellen Microsoft-Produktes entspricht. Hierunter fallen Aspekte wie die Auswahl eines modernen Themes, möglicherweise die Implementation einer „Ribbon Bar“ anstelle der traditionellen Menüstruktur, sowie die Anpassung von Reiter-Strukturen und Tabellen durch Verwendung der modernen, zu den Themes passenden Controls wie Tab-Control, Grid, usw. Derartige Wünsche existieren besonders bei Herstellern einer Standardsoftware mithilfe der Entwicklungsumgebung Team Developer. Für die Vereinfachung dieser Arbeiten finden sich im Lieferumfang von Team Developer entsprechende „Migrationsassistenten“.

Um sich ein Bild davon machen zu können, wie unterschiedlich eine „klassische“ Bedienerführung aussah und wie einfach mit den Mitteln von Team Developer 6 eine neue Bedienerführung implementiert werden kann, habe ich in einem anderen Papier die bestehende Datenstruktur eines Beispiels aus dem Seminarbereich genommen, um eine moderne, datengetriebene Oberfläche zu veranstaulichen.

Wenn es Sie interessiert, an einem einfachen Beispiel die Möglichkeiten eines neuen Bedienoberflächenparadigmas kennenzulernen, lesen Sie das Papier „Besuch in der alten Videothek“.

Die Überarbeitung des Erscheinungsbildes einer Anwendung wird in der Regel nicht das einzige sein, was derartige Kunden vornehmen. Vielmehr werden sie ihre Anwendung auch funktional weiterentwickeln. Dabei

gilt in der Regel, dass die Verteilungsbasis, aber auch die zugrundeliegende Umgebung gleich bleibt: Die Anwendung wird nach wie vor mit der eigenen Ablaufumgebung („runtime“) ausgeliefert.

Weiterentwicklung der Anwendung als „Integrationsplattform"

Schon lange bestehende, mit Team Developer entwickelte Geschäftsanwendungen wurden ursprünglich als „Desktop“-Anwendungen geplant. Um den Wert dieser Anwendungen auch weiter zu erhalten und zu vergrößern, ist es bei anderen Kunden notwendig, zum einen „Komponenten“ anderer Entwicklungsabteilungen, zum anderen aber auch teilweise komplexe Datenstrukturen anderer Systeme „on the fly“ in die Anwendung einbetten zu können.

Während die erste Anforderung beispielsweise bedeutet, dass sich Team Developer „öffnen“ musste, um sogenannte .Net assemblies (auch in klassische Win32-Anwendungen) integrieren zu können, ist es zur Erfüllung der zweitgenannten Anforderung notwendig, die Kommunikation über http: möglichweise im Zusammenhang mit der Verarbeitung von XML- oder JSON-Strukturen zu ermöglichen.

Funktionen aus dem .Net-Framework nutzen

Die in Team Developer angebotene Integration von .Net assemblies eröffnet neue Möglichkeiten, Anforderungen an die Anwendungsentwicklung auf moderne Art und Weise zu lösen. Beispielsweise kann der Komplex der Eingabe-Validierung nunmehr durch Integration der zum .Net-Framework gehörenden Bibliothek RegEx („reguläre Ausdrücke“) gelöst werden. Ein weiteres Beispiel ist die Bearbeitung von XML-Dateien mit Hilfe der zum .Net-Framework gehörenden XML-Bibliothek.

Mit anderen Entwicklern zusammenarbeiten

Die Öffnung von Team Developer (Win32) mit Hilfe des „generic assembly interface layers“ (GAIL) bietet aber nicht nur die Möglichkeit, komplexe Bearbeitungsfunktionalitäten aus dem .Net-Framework zu integrieren. Das gleiche „Layer“ wird auch verwendet, wenn es darum gehen soll, .Net-Komponenten („.Net assemblies“) eines anderen Entwicklerteams (inhouse oder extern) in die bestehende Anwendung zu integrieren.

Die mit der GAIL-Technologie bereitgestellte Möglichkeit, .Net assemblies in Team-Developer-Anwendungen einzubinden, eröffnet die Perspektive, auch im nicht sichtbaren Bereich der Anwendung Modernisierungen vornehmen zu können, um schnell und effizient neue Anforderungen an die Funktionalität der Anwendung zur Verfügung zu stellen.

> Auf einer Roadshow habe ich beispielhaft gezeigt, wie eine von einem Kollegen geschriebene.Net assembly in eine **TD-Mobile**-Anwendung integiert werden kann. Inhaltlich ging es darum, die letzten drei Tweets, die MD Consulting bei Twitter abgesetzt hat, in der mobilen Anwendung anzuzeigen.
>
> In einem weiteren Papier habe ich beschrieben, wie die gleiche .Net Assembly in eine **Team Developer 6.3 Win32-Anwendung** integiert werden kann. Es ist ein Beispiel dafür, wie .Net Komponenten in unterschiedlichen Projekten zum Einsatz kommen können.

Dieser Aspekt der Modernisierung betrifft die Funktionalität der Anwendung im Inneren, nicht die Oberfläche der Anwendung. Es besteht (technisch aufgrund der GAIL-Bibliothek) das Angebot, unterschiedliche Entwicklerteams (C#-Entwickler, Team-Developer-Entwicker) zusammenwachsen zu lassen, um an dem gemeinsamen Ziel, schnell und effizient Projekte fertigzustellen, zu arbeiten.

Nutzung bestehender Geschäftslogik in mobilen Anwendungen

Vollkommen neu ist der Aspekt, dass Bestandteile einer Team-Developer-Anwendung in einer mobilen TD-Mobile-Anwendung verwendet werden können. Damit wird das Bemühen unterstützt, wichtige Geschäftslogik in zentralen Komponenten zu bündeln, um sie dann wiederum in unterschiedlichsten Anwendungen gemeinsam zu nutzen.

Einige Kunden von MD Consulting haben diese Möglichkeit bereits genutzt, um schnell und effizient eine mobile Geschäftsanwendung zu erstellen, die in der Verarbeitungslogik auf schon lang erprobte Team-Developer Functional Classes zurückgreift.

Perspektiven der .Net Runtime

Die Forderung, .Net-Funktionalitäten in eine Win32-Anwendung zu integrieren, kann inhaltlich motiviert sein, kann aber auch als Teilschritt eines Vorhabens sein, die gesamte Anwendung am Ende als .Net-Anwendung bereitstellen zu können.

Team Developer bietet die Möglichkeit, eine bestehende Anwendung entweder als Win32-Anwendung oder als .Net-Anwendung zu kompilieren und bereitzustellen. Während im ersten Fall immer die eigene Ablaufumgebung mit ausgeliefert werden muss, kann im zweiten Fall die Anwendung ohne eigene Runtime – sie ist bereits auf allen Windows-Rechnern vorinstalliert – und mit einer eigenen Installationstoutine (Setup) – ausgeliefert werden. Durch diesen Schritt, aus einer klassischen Win32-Anwendung eine .Net-Anwendung zu machen, wird zum einen die Softwareverteilung vereinfacht, zum anderen bieten sich vollkommen neue Oberflächengestaltungsmöglichkeiten, da in einer Team-Developer .Net-Anwendung vollkommen andersartige Controls – WPF-Controls – für die Oberflächengestaltung integriert und verwendet werden können.

Die Softwareverteilung selbst kann aber noch weiter vereinfacht werden, wenn der Quellcode nicht in eine .Net-Desktop-Anwendung, sondern in eine XBAP-Anwendung kompilert wird. Eine solche Anwendung wird in den Internet Exploxer des Client-Rechner „heruntergeladen“. Die Softwareverteilung wird quasi automatisch vorgenommen.

64-Bit-Umgebung

Die Plattform aller Team-Developer-Anwendungen – Microsoft Windows – ändert sich. Absehbar ist, dass sich die 32-Bit-Plattform von Windows zugunsten der 64-Bit-Plattform ändern wird. Mit Team Developer 7 – verfügbar ab Mitte 2016 – kann der bestehende Quellcode in eine 64-bit-Anwendung kompiliert werden. Damit ist die Zukunftsfähigkeit einer bestehenden Anwendung in dieser modernen Betriebsumgebung sichergestellt.

Auch wenn das Produkt derzeit noch nicht zur Verfügung steht, muss klar sein, dass eine 64 Bit-Anwendung

- Auf der Basis von UNICODE kompiliert wird und
- nur aus Bestandteilen kompiliert werden kann, die als 64-Bit-Komponenten vorliegen.

Da fast alle Team-Developer-Anwendungen zum einen Teilen der Windows-API verwenden und zum anderen fast keine Team-Developer-Anwendung ohne „Fremdkomponenten“ implementiert ist, wird zu prüfen sein, welche Komponenten eines gegebenen Projekts durch ihre 64-Bit-Entsprechungen ersetzt werden können, um die bestehende Anwendung als 64-Bit-Anwendung mit Team Developer 7 problemlos kompilieren zu können.

In diesem Zusammenhang wird auch die Thematik „Bereitstellung einer 64-Bit-Anwendung in einem 64-Bit-Terminal-Server oder in einer 64-Bit-Citrix-Umgebung“ wieder an Bedeutung gewinnen.

Beratung

Die in diesem Papier beschriebenen Szenarien stammen aus der Beratungs- und Entwicklungstätigkeit von MD Consulting. Es hat sich gezeigt, dass bei den vielen „Modernisierungsoptionen“, die mittlerweile Team-Developer-Entwicklern zur Verfügung stehen, Beratungsbedarf besteht.

Dieser Bedarf hat wesentlich damit zu tun, dass viele Entwickler nicht wissen, wie sie die vielen Optionen für sich im Sinne der Firmenstrategie optimal nutzen können. MD Consulting als „Premium Value Added Reseller“ versteht sich daher nicht nur als Anbieter der Lizenzen von (OpenText) Gupta Technologies, sondern als ihr Partner, der Ihnen anbietet, eine auf Ihre Bedürfnisse abgestimmte Modernisierungsstrategie zu entwickeln.

MD Consulting & Informationsdienste GmbH

Michaelisstraße 13a
99084 Erfurt

Berghamer Straße 14
85435 Erding

phone (+49) 8122 97400
email info@md-consulting.de