Erfahrungen bei der Portierung von Delphi Legacy Code nach .NET*

Stephan Reiter, Reinhard Wolfinger
Christian Doppler Labor für Automated Software Engineering
Johannes Kepler Universität, 4040 Linz
{reiter,wolfinger}@ase.jku.at

Abstract: In diesem Bericht beschreiben wir Lösungsansätze für typische Problemstellungen bei der Portierung einer komplexen Win32–Delphi–Anwendung nach Delphi.NET. Wir berichten über praktische Erfahrungen aus einem Pilotprojekt zur Portierung des *New Technology Commercial System* (NTCS) des österreichischen Softwareherstellers BMD Systemhaus GmhH. Dabei wurde das BMD *Application Framework NTCS Tools* und eine Anwendung aus dem CRM-Paket erfolgreich nach .NET portiert. Ziele des Pilotprojekts sind Schätzgrundlagen für den Portierungsaufwand und Ansätze für Werkzeuge zur Automatisierung der Portierung.

1 Einleitung

Das österreichische Unternehmen BMD Systemhaus GmbH ist Hersteller von Business Software und mit seinem Produkt *New Technology Commercial System* (im Folgenden kurz: NTCS) seit langem im Sektor des Enterprise Resource Planning etabliert. BMD NTCS wird derzeit mit der Entwicklungsumgebung Borland Delphi 7 als native Win32–Anwendung entwickelt [Kna99].

Im Juli 2006 lief am Christian Doppler Labor für Automated Software Engineering ein Pilotprojekt an, das überprüfen soll, mit welchen technischen Herausforderungen bei einer Portierung auf Microsoft .NET zu rechnen ist. Es wird von Stephan Reiter, einem Studenten der Johannes Kepler Universität Linz, im Zuge einer Halbtags-Anstellung durchgeführt und ist auf ein Jahr ausgelegt. Erkenntnisse aus der prototypischen Portierung von NTCS–Teilen sollen Grundlagen für Entscheidung und Planung liefern und können später im Zuge einer Diplomarbeit weiter vertieft werden.

Ein weiteres Ziel im Rahmen des Pilotprojekts ist die Erstellung eines Prototypen für eine neu gestaltete Komponenten–Architektur. Dabei wird das im Christian Doppler Labor für Automated Software Engineering entwickelte .NET Plugin–Framework verwendet [WDPM06]. Das Plugin–Framework soll Erweiterbarkeit durch Dritthersteller und Endbenutzer sowie individuell auf Benutzer angepasste Benutzeroberflächen ermöglichen.

*Diese Arbeit wurde im Rahmen des Christian Doppler Labors für Automated Software Engineering durchgeführt und von BMD Systemhaus GmbH bzw. von der Christian Doppler Forschungsgesellschaft unterstützt.

2 Ausgangsbasis und Vorgangsweise

NTCS wird mit Borland Delphi 7 als native Win32–Anwendung entwickelt. Der Umfang des Quellcodes ist in den vergangenen Jahren auf rund 4 Mio. Zeilen gewachsen. Davon entfällt die Hälfte auf das *Application Framework NTCS Tools* und die andere Hälfte auf die verschiedenen Pakete des Anwendungsprogramms. In unserem Pilotprojekt sind die NTCS Tools und deren Portierung deshalb von großer Bedeutung, weil dieses *Application Framework* eine Abstraktionsschicht darstellt und die Anwendungsprogramme weitgehend von der technischen Basis entkoppelt. Der signifikante Portierungsaufwand ist demnach im Bereich der NTCS Tools zu erwarten.

Im Hinblick auf die spätere Entwicklung von Plugins trafen wir die Entscheidung, die NTCS Tools vor der Durchführung der Portierung in kleinere Einheiten aufzuteilen. Dies hat den Vorteil, dass Abhängigkeiten eines gegebenen Plugins, das eine bestimmte Funktionalität implementiert, genauer angegeben werden können und nur die tatsächlich benötigten Teile des Frameworks referenziert werden. Der Umstand, dass die NTCS Tools mit Bedacht auf die Gliederung in Komponenten entworfen wurden, kommt uns in diesem Schritt entgegen. Obwohl NTCS im Quellcode modular strukturiert ist, wird das Gesamtsystem nicht in Module, sondern in eine einzige monolithische ausführbare Datei übersetzt. Weder das Application Framework noch die einzelnen Pakete werden als getrennte Module ausgeliefert. Als Folge davon entstanden durch komponentenübergreifende Erweiterungen im Zuge der Weiterentwicklung zwischen den Komponenten unerwünschte Abhängigkeiten, deren Reduktion auch ein Ziel des Pilotprojekts ist.

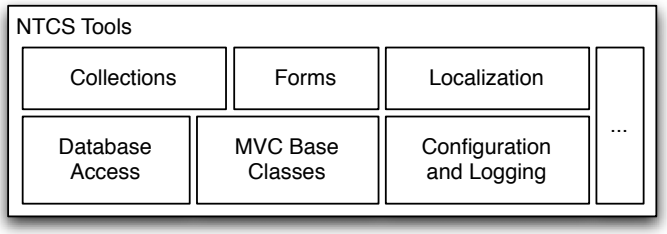

Abbildung 1: Teilbereiche der NTCS Tools werden als separate Bibliotheken portiert.

Die Zerlegung der NTCS Tools in kleinere Einheiten hat eine Reihe weiterer Vorteile, welche unabhängig vom Einsatz eines Plugin–Frameworks sind:

- Die Zerteilung des Application Frameworks in getrennte Module erleichtert das zielgerichtete Testen.
- Zudem erlaubt diese Vorgangsweise eine bessere Dokumentation des Projektfortschritts, da der Umfang einer Teilaufgabe besser abgeschätzt werden kann. Analogieschätzungen verbessern die Planbarkeit der verbleibenden Portierungsaufgaben.
- Die Reorganisation des Codes in mehrere Bibliotheken ermöglicht deren getrennte

Versionierung. Damit werden die bei Softwareaktualisierungen auszutauschenden Komponenten kleiner.

3 Legacy Code und Delphi.NET

Mit dem Developer Studio 2006 bietet Borland eine Version seines Delphi–Compilers an, der die Übersetzung von Projekten für die Zielplattformen Microsoft .NET 1.0 und 1.1 ermöglicht. Besonders von Vorteil für die Portierung bestehender Anwendungen ist die nahezu vollständige Unterstützung von unmanaged Delphi–Code. Dennoch ist es kaum möglich, komplexe Projekte ohne Anpassungen für die neue Plattform zu übersetzen. Im Folgenden soll ein Überblick über die wichtigsten Bereiche gegeben werden, die davon betroffen sind.

3.1 Unsichere Zeigeroperationen

Die Verwendung von Zeigern stellte in unserem Fall den häufigsten Grund für notwendige Codeanpassungen dar. Obwohl Zeiger unter .NET nach Kennzeichnung der entsprechenden Methode als *unsafe* prinzipell erlaubt sind, entschieden wir uns dafür, die Verwendung von Zeigern durch andere Lösungen zu ersetzen. Dies war in den meisten Fällen auch möglich, z.B. durch die Umwandlung von Records in Klassen, sodass diese mittels Referenzen in Containern verwaltet werden konnten.

3.2 Strengere Typenprüfung

Im Zuge der Portierung zeigte sich wiederholt, dass Delphi bei der Überprüfung von Typenumwandlungen größere Freiheiten einräumt als Delphi.NET. Von diesem Verhalten betroffen waren vor allem Prozeduren zur Nachrichtenbehandlung, deren Signaturen von der Borland–Vorgabe abwichen: Ungültige Umwandlungen der Parametertypen waren die Folge, die sich ausschließlich zur Laufzeit durch Ausnahmefehler zeigten, da bei der Übersetzung des Codes keine Überprüfung der Signatur von *Message Procedures* durchgeführt wird.

3.3 Änderungen in Bibliotheken und Verfügbarkeit für .NET

Die öffentlichen Schnittstellen der Klassenbibliothek für Delphi.NET unterscheiden sich nur marginal von der Version für unmanaged Delphi, weshalb bei der Portierung in diesem Bereich kaum Änderungen vorgenommen werden müssen.

Kritischer für den Verlauf des Projekts stellte sich die Abhängigkeit der NTCS Tools von

diversen externen Bibliotheken heraus: Diese werden z.B. für den Zugriff auf Datenbanken benötigt oder ermöglichen die Ausführung von benutzerdefinierten Skripten. Ist es nicht möglich, eine .NET–kompatible Version einer Bibliothek zu verwenden, bieten sich unterschiedliche Möglichkeiten an, der Situation zu begegnen:

- Ist der Quellcode einer Dritthersteller–Komponente verfügbar und der Dritthersteller bietet selbst keine .NET Variante an, kann der Quellcode der Komponente ebenfalls nach .NET portiert werden. Bei BMD ist es eine Voraussetzung für den Einsatz einer Fremdhersteller–Komponente, dass der Quellcode zur Verfügung gestellt wird, um im Fall von Fehlern in der Bibliothek selbst reagieren zu können.
- Die Erstellung eines Wrappers, der die Funktionen der Bibliothek für .NET–Anwendungen zugänglich macht. Das hat den Vorteil, dass Wrapper im Allgemeinen mit geringerem Aufwand entwickelt werden können. Von Nachteil sind allerdings die Performanzeinbußen, die durch die vom Wrapper zusätzlich eingeführte Schicht verursacht werden.

Bei der Portierung der NTCS Tools im Zuge des Pilotprojekts entschieden wir uns für die Entwicklung von Wrappern für jene Bibliotheken, die nicht .NET–kompatibel waren. Sollte sich dies später bezüglich der Performanz als Flaschenhals herausstellen, ist die Portierung der betroffenen Bibliothek zu erwägen.

4 Fazit und Ausblick

Aufgrund der Kompatibilität von Delphi.NET zu unmanaged Delphi ließen sich bis zum jetzigen Zeitpunkt bereits große Teile der NTCS Tools und des CRM-Pakets nach .NET portieren. Als gute Wahl stellte sich die diskutierte Vorgangsweise der Unterteilung der Codebasis in separate Einheiten heraus, da die nun für .NET verfügbaren Bibliotheken bereits in diversen Testanwendungen eingesetzt werden können.

Nach Abschluss der Portierung wollen wir noch klären, welchen Aufwand es darstellen würde, Komponenten der graphischen Benutzeroberfläche, die aktuell mit VCL.NET realisiert werden, durch deren Äquivalent in den Windows Forms zu ersetzen. Dadurch würde der Quellcode näher an die .NET–Klassenbibliothek rücken und die Integration von Anwendungsteilen, die in anderen Programmiersprachen erstellt wurden, erleichtern.

Literatur

[Kna99] Markus Knasmüller. Quo Vadis, BMD? Research Projects at BMD Steyr - An Experience Report. In *Proc. European Software Day*, Milano, Italy, September 1999.

[WDPM06] Reinhard Wolfinger, Deepak Dhungana, Herbert Prähofer und Hanspeter Mössenböck. A Component Plug-In Architecture for the .NET Platform. In *Modular Programming Languages*, Jgg. Volume 4228/2006, Seiten 287–305, 2006.