Sage CRM

Sage CRM 2016 R1 Webservices-Handbuch

Inhalt

Kapitel 1: Einführung in Webservices

Mithilfe der API (Application Programming Interface) der Sage CRM Webservices können Entwickler CRM Datensätze per Fernzugriff mit SOAP (Simple Object Access Protocol) über HTTP anhand von XML (Extensible Markup Language) verändern. Es ist möglich, auf einen CRM Server oder ein Host-System von einem bestimmten Client-Computer (in der Regel einem anderen Server) aus zuzugreifen, um Datensätze für jede verfügbare Entität (beispielsweise Firmen, Personen, Verkaufschancen, Tickets, Angebote und Aufträge) zu lesen, zu erstellen, zu aktualisieren oder zu löschen.

Weitere Informationen zum Einfügen und Aktualisieren von Feldern mit Angebots- und Auftragspositionen finden Sie unter Liste der Webservices-Objekte (Seite 5-1).

Hinweis: Die SOAP-Webservices-API ist in der Sage CRM Essentials Version nicht verfügbar. Weitere Informationen finden Sie in der Versionsmatrix in der Hilfe zum Produkt.

In erster Linie spielen die folgenden Schritte bei der Kommunikation mit der Sage CRM Webservices eine Rolle:

1. Die WSDL (Web Service Description Language) wird auf dem CRM Server generiert.
2. Anschließend greift der Benutzer vom Client-Computer aus auf die WSDL-Datei zu und bereitet die Anfrage vor.
3. Der Client-Computer leitet die Anfrage mit ihren Parametern an den Webservice weiter.
4. Der Webservice verarbeitet die Anfrage und sendet eine Antwort an den Client-Computer.
5. Der Client-Computer empfängt die Antwort gleichzeitig und verarbeitet die zurückgegebenen Daten oder bearbeitet Fehler.

Allgemeiner Überblick über die Webservice-Technologie

Webservices stellen eine standardisierte Methode zum Integrieren webbasierter Anwendungen unter Verwendung von XML, SOAP und WSDL über ein Internet-Protocol-Backbone dar. Die Webservice-Komponenten arbeiten wie folgt:

- XML markiert die Daten.
- SOAP überträgt die Daten. Weitere Einzelheiten zu SOAP finden Sie unter http://www.w3.org/TR/SOAP.
- WSDL beschreibt die verfügbaren Services.

Mithilfe dieser Technologie können Unternehmen Daten austauschen, ohne dass sie die IT-Systeme der jeweils anderen hinter der Firewall genauer kennen. Sie bietet den Benutzern keine GUI, wie es bei herkömmlichen Client-Server-Modellen der Fall ist. Stattdessen gibt Webservices Geschäftslogik, -daten und -prozesse über eine Programmierschnittstelle innerhalb eines Netzwerks frei. Entwickler können die Webservices einer GUI hinzufügen, wie etwa eine Webseite oder ein ausführbares Programm, um Benutzern auf diese Weise die benötigten Funktionen zur Verfügung zu stellen.

Diese Technologie macht es möglich, dass verschiedene Anwendungen verschiedener Quellen ohne zeitraubendes Custom-Coding miteinander kommunizieren können. Da sämtliche Kommunikation in XML erfolgt, legen die Webservices den Benutzer nicht auf eine bestimmte Programmiersprache fest.

CRM Webservices-Funktionen

Die Fähigkeit zur Manipulation von Datensätzen in Sage CRM bringt mittelbar folgende Funktionen mit sich:

- Ändern von Daten. Es können Datensätze zur CRM Datenbank hinzugefügt, in ihr aktualisiert und aus ihr gelöscht werden.
- Integration mit Anwendungen anderer Anbieter. Durch den Zugang zur API der Sage CRM Webservices können Sie Anwendungen von anderer Anbieter, die innerhalb Ihres Unternehmens verwendet werden, wie beispielsweise Buchführungspakete oder ERP-Systeme (Enterprise Resource Planning), mit dem Sage CRM Server- oder Cloud-Versionen integrieren.
- Cloud-Umgebungen. Sage CRM Webservices kann nicht nur Datensätze auf einem Standard CRM Server bearbeiten, sondern ist auch mit Cloud-Versionen kompatibel. Folglich können auch Kunden in einer Cloud-Umgebung die Technologie und ihre Funktionen einsetzen.

Hinweis: Sage CRM Cloud wurde so konzipiert, dass die Leistung für Kunden optimiert wird, die ihr Sage CRM Konto aktiv einsetzen. Dazu zählen beispielsweise Kunden, die sich an- und abmelden oder das Outlook-Plug-In , Sage CRM SDATA oder auch die Webservices-API verwenden. Bei CRM Kundeninstanzen, die nicht verwendet werden, wird die Bereitstellung beendet, damit sie nicht unnötig Ressourcen verbrauchen.

Kapitel 2: Einrichten der CRM Webservices

Voraussetzungen

Alle aktuellen, mit Soap 1.1 kompatiblen Entwicklungsumgebungen sind mit Sage CRM Webservices kompatibel. Unterstützte Umgebung sind u. a. Microsoft Visual Studio 2003 und höher (C#, J#, VB.NET) und Microsoft Visual C# 2005 Express Edition.

Für das Arbeiten mit Webservices erforderliche Schritte

Die folgenden Schritte spielen bei der Arbeit mit Webservices eine Rolle:

1. Einrichten eines Webservices-Benutzers
2. Angeben der Webservices-Konfigurationseinstellungen
3. Zugreifen auf die WSDL-Datei
4. Vorbereiten der Anfrage und Weiterleiten der Anfrage an Webservices
5. Bearbeiten der Antwort – zurückgegebene Werte oder Fehlermeldungen.

Die Schritte 1 bis 2 werden nachfolgend beschrieben. Informationen zum Vorbereiten der Anfrage und Verarbeiten der Antwort finden Sie unter Übersicht über die Objekte und Funktionen (Seite 3-1) und in den Webservices-Beispielen.

Einrichten eines Webservices-Benutzerkontos

Bevor Sie auf Webservices zugreifen können, muss ein Benutzerkonto für Webservices eingerichtet werden.

So richten Sie ein Benutzerkonto für Webservices ein:

1. Wählen Sie **Administration | Benutzer | Benutzer** aus, und suchen Sie nach dem Benutzer, der Zugriff auf Webservices haben soll.
2. Wählen Sie den Hyperlink des Benutzers aus, und klicken Sie auf die Schaltfläche **Ändern**.
3. Führen Sie einen Bildlauf hinunter zur Registerkarte **Sicherheitsprofil** durch, und setzen Sie das Feld **Webservices-Zugriff erlauben** auf **Wahr**.
4. Klicken Sie auf die Schaltfläche **Speichern**.

Es kann sich jeweils nur ein Webservices-Benutzer mit derselben ID anmelden. Wenn ein Benutzer sich bei einer anderen Anwendung anzumelden versucht, wird eine Fehlermeldung angezeigt, in der der Benutzer darüber informiert wird, dass er sich zuerst abmelden muss. Es ist jedoch möglich, sich beim Desktop oder von einem anderen Rechner mit derselben ID anzumelden, während eine Webservices-Anwendung ausgeführt wird.

Die Funktion **Sicherheit auf Feldebene** wirkt sich darauf aus, auf welche Entitätenfelder zugegriffen werden kann bzw. welche Entitätenfelder unter Verwendung der Webservices-Methoden aktualisiert werden können. Wenn einem Benutzer beispielsweise bei Sicherheit auf

Feldebene der Lesezugriff auf ein Feld verweigert wird, können mit den Methoden, die durch eine Webservices-Sitzung unter Verwendung der Anmeldedetails desselben Benutzers aufgerufen wurden, die Werte des betreffenden Felds nicht zurückgegeben, aktualisiert oder gelöscht werden. Weitere Informationen zur Sicherheit auf Feldebene finden Sie in der/im Handbuch für Systemadministratoren.

Angeben der Webservices-Konfigurationseinstellungen

Wählen Sie für den Zugriff auf die Webservices-Konfigurationseinstellungen **Administration | System | Webservices**.

In der folgenden Tabelle werden die Felder der Seite **Webservices** erläutert.

Feld	Beschreibung
Maximale Anzahl der zurückgegebenen Datensätze	Die maximale Anzahl an Datensätzen, die die Webservices jeweils zu einem Zeitpunkt zurückgeben sollen. Diese Funktion wird in Verbindung mit den Methoden "query" und "queryrecord" verwendet. Die Zahl, die Sie hier eingeben, entspricht der Anzahl der Datensätze, die in jedem Batch in Beantwortung einer Anfrage zurückgegeben werden. Wenn die einzelnen Batches zurückgegeben werden, werden Sie aufgefordert, den nächsten Batch abzurufen, bis alle Datensätze, die der Anfrage entsprechen, zurückgegeben worden sind. Wenn dieses Feld auf 0 gesetzt ist, werden alle Datensätze, die der Anfrage entsprechen, in einem einzigen Batch zurückgegeben.
Maximale Größe der Anfrage	Die maximale Anzahl an Zeichen, die Benutzer an die Webservices schicken können.
WSDL für alle verfügbar machen	Steht diese Einstellung auf **Ja**, Sie finden den URL für die Anzeige der WSDL-Datei unter **Administration** | **Eigenes Konto** | **Externer Zugriff** | **Externer Zugriff – Informationen** finden.
Webservices aktivieren	Wird diese Option auf **Ja** eingestellt, ist die Funktion **Webservices** aktiviert. Stellen Sie die Option auf **Nein** ein, um die Webservices zu deaktivieren. Wechseln zum Aktivieren bzw. Deaktivieren von Webservices für eine einzelne Tabelle oder Entität zu **Administration | Anpassung | <Name der Entität/Tabelle> | Externer Zugriff**, und legen Sie das Feld **Webservices** zum Aktivieren auf **Ja** und zum Deaktivieren auf **Nein** fest.
Dropdown-Felder als Zeichenfolgen in WSDL-Datei	Die Standardeinstellung lautet **Ja**. Dropdown-Felder werden in der WSDL-Datei als aufgezählte Typen angezeigt, also beispielsweise **comp_status** als Aufzählung, die die Werte der Dropdown-Liste enthält. Weitere Informationen finden Sie unter "Objekte und Funktionen". Ist diese Option auf **Ja** festgelegt, werden die aufgezählten Typen zu "Zeichenfolgen". Dies ist die empfohlene Einstellung. Das bedeutet, dass das Feld **comp_status** innerhalb der Firma jetzt vom Typ **String** ist.

Feld	Beschreibung
Alle Datums- und Zeitangaben in UTC-Zeit senden und empfangen	Wenn diese Option ausgewählt ist, werden alle Daten, die vom Server kommen, auf UTC-Zeit gesetzt. Ferner werden alle Daten, die beim Webserver ankommen, abweichend von der UTC-Zeit eingestellt. Dies ist in erster Linie von Bedeutung für Migrationen zur Cloud-Version aus anderen Zeitzonen.
Web-Anfrage von IP-Adresse akzeptieren	Geben Sie die eindeutige IP-Adresse an, von der die WSDL-Datei aufgerufen werden soll. Dabei sollte das Feld **Webservices für alle verfügbar machen** auf **Nein** gesetzt werden.
Webservice-Anmeldung erzwingen	Wird die Verbindung zwischen dem Webservice-Client und dem Dienst überraschend unterbrochen, bleibt dieser Client bei dem Server angemeldet, der den Dienst hostet. Dies bedeutet, dass eine neue Instanz des Clients sich nicht mehr beim Server anmelden kann. Stellen Sie die Einstellung **Webservice-Anmeldung erzwingen** jedoch auf **Ja** ein, wird die alte Instanz des Client automatisch abgemeldet, wenn eine neue Instanz einen Anmeldeversuch unternimmt. Indem neue Anmeldungen erzwungen werden, verhindert dieses Feld, dass Benutzer nach einer fehlgeschlagenen Verbindung oder einer missglückten Abmeldung von einem Webservice abgemeldet werden.

Empfohlene Konfigurationseinstellungen

Es folgen die empfohlenen Einstellungen, um Ihrem Client den Zugriff auf die Webservices während der Entwicklung zu ermöglichen:

1. Legen Sie das Feld **Webservices aktivieren** auf **Ja** fest.
2. Wählen Sie im Feld **WSDL für alle verfügbar machen** die Option **Ja** aus.
3. Setzen Sie das Feld **Webservice-Anmeldung erzwingen** auf **Ja**.

Nachdem Sie mit dem Testen des Webservices-Clients fertig sind, wird empfohlen, dass Sie die Einstellung **WSDL für alle verfügbar machen** im Sinne einer größeren Sicherheit wieder auf **Nein** setzen.

Zugreifen auf die WSDL-Datei

Wie bei typischen SOAP-Webservices bietet CRM eine Webservices-Datei für die Beschreibungssprache, die WSDL(Web Services Description Language)-Datei.

Öffnen Sie für den Zugriff auf diese Datei in der Client-Anwendung die Datei **CRMWebService.WSDL** an Ihrer Installationsadresse.

Unter SageCRM.com können Sie den URL für die Anzeige der WSDL-Datei unter **Administration** | **Eigenes Konto** | **Externer Zugriff** | **Externer Zugriff – Informationen** ermitteln. Der URL sieht etwa so aus:

https://cloud.sagecrm.com/[*Kontoname123*]/eware.dll/webservices/CRMwebservice.wsdl

Die CRM WSDL-Datei beschreibt alle APIs, über die CRM verfügt, sowie alle XML-Typen, die die

APIs erwarten. Ferner beschreibt die Datei den Server und das Verzeichnis, in dem diese speziellen Dienste zu finden sind. Sobald der Client die WSDL-Datei gelesen und analysiert hat, kann er die APIs auf die gleiche Weise wie jede andere typische Funktion aufrufen. Da diese Daten als XML weitergeleitet und zurückgegeben werden, lassen sich die Daten problemlos durch den Client interpretieren und manipulieren.

Wenn Sie z. B. in Microsoft Visual Studio eine Client-Anwendung erstellen, muss Ihr Visual Studio-Projekt einen Webverweis zu beispielsweise enthalten.

https://cloud.sagecrm.com/[*Kontoname123*]/eware.dll/webservices/CRMwebservice.wsdl

Wenn Sie den Verweis in Visual Studio hinzufügen, sind im Hauptfenster die in den Webservices verfügbaren Methoden aufgelistet.

Wenn Sie den Dienst **CRMWebServices** nennen, wird der neue Ordner **CRMWebServices** Ihrem Projekt hinzugefügt, der die Dateien **webservice.discomap** und **webservice.wsdl** enthält. Der "Webservice-Proxy", eine C#-Version der WSDL-Datei zur Verteilung der Daten im SOAP-Format an den Webservice, wird automatisch erstellt.

Hinweis: Um in Visual Studio 2008 einen Webverweis hinzuzufügen, müssen Sie **Dienstverweis hinzufügen | Erweitert | Webverweis hinzufügen** auswählen.

Kapitel 3: Übersicht über die Objekte und Funktionen

Bearbeiten von Datensätzen

Bevor Sie die Arbeit mit den CRM Webservices aufnehmen, müssen Sie sich zum einen mit allen Funktionen vertraut machen, die Sie aufrufen können, um Datensätze zu bearbeiten, und zum anderen mit den Objekten (zu denen die Funktionen aufgerufen werden), die in der API verfügbar sind.

Funktionen

Funktionen sind Aktionen, die vom Client-Computer aufgerufen werden, um bestimmte Aufgaben auf dem Server auszuführen, wie etwa das Hinzufügen, Aktualisieren oder Löschen von Informationen. Sage CRM Funktionen sind synchrone Anfragen, die automatisch übernommen werden. Sobald sie übernommen worden sind, bearbeitet Sage CRM Webservices die Anfrage und gibt eine Antwort zurück. Anschließend bearbeitet die Client-Anwendung die Antwort entsprechend.

Hinweis: Alle Einfügungen sollten normalerweise auf Entitätsbasis ausgeführt werden. Sie können eine Firma (oder eine Person) jedoch gemeinsam mit ihrer Adresse, Telefonnummer und E-Mail-Information aktualisieren. Dies dient der Vereinfachung der Integration. In vielen Systemen beinhaltet ein einzelner Kontaktdatensatz Firmen-, Personen-, Telefon-, E-Mail- und Adressinformationen.

Eine vollständige Liste finden Sie unter Liste der Webservices-Funktionen (Seite 4-1).

Objekte

Objekte sind programmgesteuerte Abbildungen von Daten im System. In Sage CRM stellen Objekte Hauptentitäten wie Firmen und Personen sowie sekundäre Entitäten wie Adressen und Produkte dar. Die Daten werden bearbeitet, wenn die API der Webservices mit den Objekteigenschaften interagiert, die bestimmte Felder in den Entitäten darstellen.

Eine vollständige Liste finden Sie unter Liste der Webservices-Objekte (Seite 5-1). Siehe auch Das CRM RecordType-Objekt (Seite 6-1)und Auswahlfelder in Webservices (Seite 7-1).

Kapitel 4: Liste der Webservices-Funktionen

Alle nachstehend genannten Objekte werden in der WSDL-Datei definiert.

Funktion	Beschreibung
logon	Führt die Anmeldung beim Server aus und beginnt eine Sitzung.
logoff	Führt die Abmeldung vom Server aus und beendet die Sitzung.
query	Führt basierend auf einer WHERE-Klausel eine Abfrage zu einem bestimmten Objekt aus und gibt einen Datensatz oder eine Datensatzgruppe zurück, durch die die Abfrage zufriedenstellend beantwortet wird. Gibt Ergebnisse in Batches (deren Größe anhand des Felds **Maximale Anzahl** der zurückgegebenen Datensätze unter **Administration | System | Webservices** festgelegt wird) zurück. Jeder Batch verfügt über eine Kennzeichnung, die **More** heißt. Wenn **More** auf **Wahr** festgelegt ist, warten auf dem Server noch weitere Datensätze auf diese Abfrage. Rufen Sie **Next** auf, um zum nächsten Daten-Batch zu gelangen. Wenn Sie etwas anderes als **Next** aufrufen, wird die Anfrage geschlossen.
next	Gibt den nächsten Datensatz-Batch zurück, der zu einer Abfrage passt. Jeder Batch verfügt über eine Kennzeichnung, die **More** heißt. Wenn **More** auf **True** gesetzt ist, können Sie so lange **Next** aufrufen, bis alle Batches zurückgegeben wurden (d. h. bis **More** auf **False** gesetzt wird).
queryentity	Gibt einen Datensatz zurück, wenn Sie ein Objekt (beispielsweise eine Firma) und eine ID (beispielsweise **queryentity (company, 42)**) angeben.

Funktion	Beschreibung
queryid	Gibt ein Objekt vom Typ *aisid* zurück (siehe Liste der Webservices-Objekte (Seite 5-1)). Richten Sie mit einer WHERE-Klausel eine Abfrage an die Datenbank, werden ein Datum und eine Reihe von IDs jeweils mit verschiedenen Kennzeichnungen zurückgegeben, die angeben, ob der betreffende Datensatz seit diesem Datum erstellt, aktualisiert oder gelöscht wurde. Dies ist sehr hilfreich bei der Datensynchronisierung.
queryidnodate	Gibt ein Objekt vom Typ *aisid* zurück (siehe Liste der Webservices-Objekte (Seite 5-1)). Richten Sie mit einer WHERE-Klausel eine Abfrage an die Datenbank. Dies ist hilfreich, wenn Sie beispielsweise verschiedene Firmen-IDs benötigen und nicht direkt alle Firmendaten abrufen möchten.
getmetadata	Wenn Sie einen Tabellennamen übergeben, wird hiermit eine Liste der CRM Feldtypen zurückgegeben, die Metadaten (beispielsweise Feldname, Typ) zu der angefragten Tabelle enthalten.
getdropdownvalues	Wenn Sie eine Tabelle übergeben, werden hiermit eine Liste der Dropdown-Felder in der betreffenden Tabelle und die Liste der Werte zurückgegeben, die CRM zu den jeweiligen Feldern erwartet. Dies ist wichtig, da CRM für Dropdown-Felder ein bestimmtes Set von Werten erwartet, sodass Sie in der Lage sein müssen, diese Werte programmatisch abzurufen.
add	Fügt einem bestimmten Objekt (beispielsweise Firma) Datensätze oder Listen von Datensätzen hinzu. Beispielsweise **add(„company“, NewCompany1, New Company2, New Company3)**.
addresource	Fügt einen Benutzer als Ressource hinzu. Dieser Benutzer ist kein vollständig aktivierter Benutzer. Die Funktion besteht nur, um die Datenmigration zu erleichtern.
update	Aktualisiert für ein bestimmtes Objekt (beispielsweise **company**) Datensätze oder Listen von Datensätzen.
altercolumnwidth	Wird zur Anpassung der Spaltenbreite verwendet, um die Kompatibilität mit Datenbanken von Drittanbietern, beispielsweise ACT!, sicherzustellen.

Funktion	Beschreibung
delete	Löscht für ein bestimmtes Objekt (beispielsweise **company**) Datensätze oder Listen von Datensätzen. Bitte beachten Sie, dass Sie aus den folgenden Tabellen keine Datensätze löschen können, da sie historische Daten enthalten: newproduct, uomfamily, productfamily, pricing, pricinglist.
addrecord	Dieselbe Funktion wie die Funktion **add**, nur dass sie eine andere Signatur hat und die Listen der Felder im Typ crmrecord verwendet. Siehe Das CRM RecordType-Objekt (Seite 6-1).
queryrecord	Dieselbe Funktion wie die Funktion **query**, nur dass sie eine andere Signatur hat und die Listen der Felder im Typ **crmrecord** verwendet. Siehe Das CRM RecordType-Objekt (Seite 6-1).
nextqueryrecord	Gibt den nächsten Datensatz-Batch zurück, der zu einem Abfragedatensatz passt. Jeder Batch verfügt über eine Kennzeichnung, die **More** heißt. Wenn **More** auf **True** gesetzt ist, können Sie so lange **Next** aufrufen, bis alle Batches zurückgegeben wurden (d. h. bis **More** auf **False** gesetzt wird).
updaterecord	Dieselbe Funktion wie die Funktion **update**, nur dass sie eine andere Signatur hat und die Listen der Felder im Typ crmrecord verwendet. Siehe Das CRM RecordType-Objekt (Seite 6-1).
getallmetadata	Gibt eine Liste von mit allen Tabellen verbundenen Feldern sowie verschiedene Informationen zum Typ zurück.
getversionstring	Gibt die jeweilige Version von CRM zurück.

Kapitel 5: Liste der Webservices-Objekte

Die folgenden Objekte stellen (primäre und sekundäre) CRM-Entitäten dar. Wenn dem CRM-System benutzerdefinierte Entitäten hinzugefügt werden, stehen diese Entitäten ebenfalls zur Verfügung. Auf Grund des Umstands, dass die WSDL dynamisch generiert wird, werden Anpassungen, die am System vorgenommen werden – wie etwa das Hinzufügen einer neuen Entität – jedes Mal erfasst, wenn die WSDL auf Client-Seite aktualisiert wird.

Abstrakte Objekte

Objektname	Beschreibung
ewarebase abstract	Hierbei handelt es sich um eine abstrakte Deklaration, von der alle anderen CRM Objekte abgeleitet werden.
idbase abstract	Hierbei handelt es sich um eine abstrakte Deklaration, von der alle ID-Typen abgeleitet werden.
ewarebaselist	Dies stellt eine Liste der vorstehenden abstrakten Objekte dar.
crmrecordtype	Eine Aufzählung, die die Typen eines CRM Felds darstellen, d. h. „string“, „datetime“, „integer“, „decimal“. Der Wert „multiselectfield“ kennzeichnet einen geschachtelten Array von Strings, die die Werte eines Mehrfachauswahlfelds darstellen. Die letzte Option ist „crmrecord“. Sie kennzeichnet einen Feldtyp, der andere Felder enthält. Weitere Informationen finden Sie unter Das CRM RecordType-Objekt (Seite 6-1).
crmrecord	Enthält einen Entitätsnamen und eine Liste von Objekten vom Typ „recordfield“, die einen Datensatz in der CRM-Datenbank darstellen.
aisid	Enthält die ID des Datensatzes, das Erstellungs- und das Aktualisierungsdatum sowie eine Kennzeichnung, aus der hervorgeht, ob der Datensatz seit Übergabe des Tokens an **queryid** hinzugefügt, aktualisiert oder gelöscht wurde.
multiselectfield	Dieser Typ stellt ein Feld zur Mehrfachauswahl in CRM dar. Er enthält einen Feldnamen und einen Array aus Strings, die die Werte des Felds in CRM darstellen. Bitte beachten Sie, dass es sich bei diesen Werten wie auch bei den anderen Feldern um Übersetzungen handelt.
recordfield	Dies stellt ein Feld in einem Datenbankdatensatz dar. Es hat einen Namenwert und ist vom Typ „crmrecordtype“. Es kann ferner eine verschachtelte Struktur darstellen. Beispielsweise könnte der Name von „recordfield“ in einer Firma „crmrecord“ „person“ lauten.

Objektname	Beschreibung
	Der Typ wäre "crmrecord" und die Datensatzeigenschaft würde eine Liste aus "crmrecords" enthalten, einen für jede Person in der Firma.

Standardobjekte

Objektname	Beschreibung
company	Dieses Objekt steht für die Entität „Firma“ in CRM.
person	Dieses Objekt steht für die Entität „Person“ in CRM.
lead	Dieses Objekt steht für die Entität „Interessent“ in CRM.
communication	Dieses Objekt steht für die Entität „Kommunikation“ in CRM.
opportunity	Dieses Objekt steht für die Entität „Verkaufschance“ in CRM.
cases	Dieses Objekt steht für die Entität „Tickets“ in CRM.
users	Dieses Objekt steht für die Entität „Benutzer“ in CRM.
quotes	Dieses Objekt steht für die Entität „Angebote“ in CRM.
orders	Dieses Objekt steht für die Entität „Aufträge“ in CRM.
quoteitem	Dieses Objekt steht für die Entität „Angebotspositionen“ in CRM.
orderitem	Dieses Objekt steht für die Entität „Auftragsposition“ in CRM.
opportunityitem	Dieses Objekt steht für die Entität „Verkaufschancenposition“ in CRM.
currency	Dieses Objekt steht für die Entität „Währung“ in CRM.
address	Dieses Objekt steht für die Entität „Adresse“ in CRM.
phone	Dieses Objekt steht für die Entität „Telefon“ in CRM.
email	Dieses Objekt steht für die Entität „E-Mail“ in CRM.
newproduct	Dieses Objekt steht für die Entität „Neues Produkt“ in CRM.
uom	Dieses Objekt steht für die Entität „Mengeneinheit“ in CRM.
uomfamily	Dieses Objekt steht für die Entität „Mengeneinheitsgruppe“ in CRM.
pricing	Dieses Objekt steht für die Entität „Preisfestsetzung“ in CRM.
pricinglist	Dieses Objekt steht für die Entität „Preisliste“ in CRM.
productfamily	Dieses Objekt steht für die Entität „Produktfamilie“ in CRM.

Einfügen und Aktualisieren von Angebots- und Auftragspositionen

Beachten Sie beim Einfügen und Aktualisieren von Feldern für Angebots- und Auftragspositionen, dass verschiedene Positionstypen bestimmte Felder erfordern. Wenn diese nicht gefunden werden, löst der Webservice eine Ausnahme aus.

Beim Einfügen einer neuen Standardposition sind die folgenden Felder erforderlich:

- orderquoteid
- opportunityid
- lineitemtype (entweder 'i' für "simple item" (einfache Position), 'f' für Freitextposition oder 'c' für Kommentarzeile)
- productid
- uomid
- quantity
- quotedprice

Beim Einfügen einer neuen frei wählbaren Textposition sind die folgenden Felder erforderlich:

- orderquoteid
- opportunityid
- lineitemtype ('i', 'f' oder 'c')
- description
- quantity
- quotedprice

Beim Einfügen einer neuen Kommentarposition sind die folgenden Felder erforderlich:

- orderquoteid
- opportunityid
- lineitemtype ('i', 'f' oder 'c')
- description

Beim Aktualisieren einer Standardposition sind die folgenden Felder erforderlich:

- quantity
- quotedprice
- uomid

Beim Aktualisieren einer frei wählbaren Textposition sind die folgenden Felder erforderlich:

- description
- quantity

Beim Aktualisieren einer Kommentarposition sind die folgenden Felder erforderlich:

- description

Die folgenden beiden Felder können nicht aktualisiert werden und lösen eine Ausnahme aus:

- linetype
- orderquoteid

Darüber hinaus werden bestimmte Felder von CRM im Webservice-Code berechnet oder überschrieben, und die Werte, die der Benutzer an sie übergibt, werden ignoriert. Es handelt sich bei diesen Feldern um:

- quotedpricetotal
- listprice
- discount
- discountsum

Kapitel 6: Das CRM RecordType-Objekt

Das Objekt „crmrecordtype“ (mit den mit ihm verbundenen Funktionen **add**, **update** und **delete**) sorgt für eine dynamische und flexible Programmierumgebung. Anstatt eine Entität (beispielsweise eine Firma) abzufragen und ein Objekt eines starken Typs (Firma) zurückzuerhalten, kann durch die Flexibilität, die das Objekt **crmrecordtype** bietet, eine Entität abgefragt werden und es wird eine Liste der Felder zurückgegeben, durch die Sie schrittweise durchlaufen können. Das bedeutet, dass es möglich ist anzugeben, welche Felder Sie in Ihrer Abfrage zurückerhalten möchten.

Die Möglichkeit, durch Datensätze zu iterieren, bietet Programmierern eine wirkungsvolle und flexible Schnittstelle. Durch sie können den Webservices-Entitäten Felder dynamisch hinzugefügt werden, und in Client-Anwendungen werden keine Objekte mit starker Typisierung mehr benötigt. Beim Ausführen dieser Aufgaben sollten die Code-Muster strikt befolgt werden.

Im Folgenden sehen Sie ein Abfragebeispiel, in dem eine Feldliste, ein Entitätsname, eine WHERE-Klausel und eine Order-by-Anweisung angegeben sind. Bitte beachten Sie, dass alle Felder zurückgegeben werden, wenn Sie ein Sternchen (*) eingeben oder die Feldliste leer lassen.

```
Private static void CallQueryRecordOnCompanyEntity()
{
       String companyid = ReadUserInput("Bitte Firmennamen eingeben: ");
       Queryrecordresult aresult = Binding.queryrecord("comp_companyid,address","comp_
name='compo1'","company","comp_companyid");
}
```

Kapitel 7: Auswahlfelder in Webservices

Wenn Ihre Dropdown-Felder aus Strings bestehen, werden diese Felder nicht in der WSDL angezeigt. Da Strings die Standardoption sind, werden diese Felder in einer Standardeinrichtung nicht angezeigt.

In der folgenden Tabelle werden die Auswahlfelder in CRM erläutert. In der WSDL-Datei steht für diese Werte ein enumerierter Typ für jedes Feld, das Werte enthält. Für jede Entität gibt es mehrere dieser Felder.

Hinweis: Enumerierte Werte werden in der Standardsystemsprache zurückgegeben.

```
<s:simpleType name="case_problemtype">
<s:restriction base="s:string">
<s:enumeration value="Additional Software Required" />
<s:enumeration value="Software Bug" />
<s:enumeration value="Setup/Installation" />
<s:enumeration value="Customer knowledge" />
</s:restriction>
</s:simpleType>
```

Liste der Auswahlfelder

Firmenauswahlfelder

- comp_employees
- comp_indcode
- comp_mailrestriction
- comp_revenue
- comp_sector
- comp_source
- comp_status
- comp_territory
- comp_type

Personenauswahlfelder

- pers_gender
- pers_salutation
- pers_source
- pers_status
- pers_territory
- pers_titlecode

Interessentenauswahlfelder

- lead_decisiontimeframe
- lead_priority
- lead_rating
- lead_source
- lead_stage
- lead_status

Kommunikationsauswahlfelder

- comm_action
- comm_hasattachments
- comm_notifydelta
- comm_outcome
- comm_priority
- comm_status
- comm_type

Verkaufschancen-Auswahlfelder

- oppo_priority
- oppo_product
- oppo_scenario
- oppo_source
- oppo_stage
- oppo_status
- oppo_type

Ticketauswahlfelder

- case_foundver
- case_problemtype
- case_productarea
- case_solutiontype
- case_source
- case_stage
- case_status
- case_targetver

Adress- und Produktauswahlfelder

- addr_country
- prod_uomcategory

Die Funktion "GetDropDownValues"

Verwenden Sie die Funktion *getdropdownvalues*. Siehe Liste der Webservices-Funktionen (Seite 4-1)), um eine Liste der Dropdown-Felder in einer Tabelle und die Liste der Werte zurückgegeben, die CRM zu den jeweiligen Feldern erwartet.

Es folgt ein Beispiel in C# einer Funktion zum Auffüllen eines Kombinationsfeldes mit Auswahlwerten aus einem bestimmten Feld.

```
private void LoadDropDowns(string entity, string fieldname, ComboBox controlname,
WebService WS)
{
	dropdownvalues[] DropDowns;
	DropDowns = WS.getdropdownvalues(entity);
	controlname.Items.Clear();
	for (int i = 0; i < DropDowns.Length; i++)
	{
		if (DropDowns[i].fieldname == fieldname)
		{
			for (int x = 0; x < DropDowns[i].records.Length; x++)
			{
				controlname.Items.Add(DropDowns[i].records[x].ToString());
			}
		}
	}
	controlname.SelectedIndex = 0;
}
```

So verwenden Sie die Funktion zum Anzeigen der **comp_sector**-Auswahlfelder im Kombinationsfeld **comboSector** (wobei **oWebService** das aufgerufene Webservice-Objekt ist):

```
LoadDropDowns("company", "sector", comboSector, oWebService);
```

Kapitel 8: SOAP-Beispielanfragen

Die folgenden Abschnitte enthalten verschiedene Beispiel-Soap-Anfragen. Einige der Anfragebeispiele sind in C# verfasst, während andere in XML verfasst sind.

Dynamisches Abrufen von WSDL-Daten und Beispiel-Authentifizierungsanforderung

Sie können den Endpunkt von GetCRMURL zum dynamischen Abrufen der benutzerdefinierten URL für die Webservice-WSDL und -Verbindungszeichenfolge für ein Sage CRM Cloud-Konto verwenden. Dieser Endpunkt gibt auch die benutzerdefinierten URLs für die SDATA-API und den URL für die Web-Interessenten-Erfassung des Kontos zurück.

Zum Abrufen dieser Details führen Sie einfach eine HTTP GET-Anforderung für einen der beiden folgenden Endpunkte durch:

- Endpunkt des NA-Rechenzentrums: https://cloud.na.sagecrm.com/getcrmurls
- Endpunkt des EU-Rechenzentrums: https://cloud.eu.sagecrm.com/getcrmurls

Sie müssen sich beim Endpunkt authentifizieren, indem Sie in den nachfolgend angegebenen Kopfzeilen gültige CRM Benutzeranmeldeinformationen senden. Stellen Sie sicher, dass der Webservices-Zugriff für die verwendeten Benutzeranmeldeinformationen in CRM auf "Wahr" eingestellt ist.

Sage CRM Cloud-Sessions laufen nach 30 Minuten Inaktivität ab.

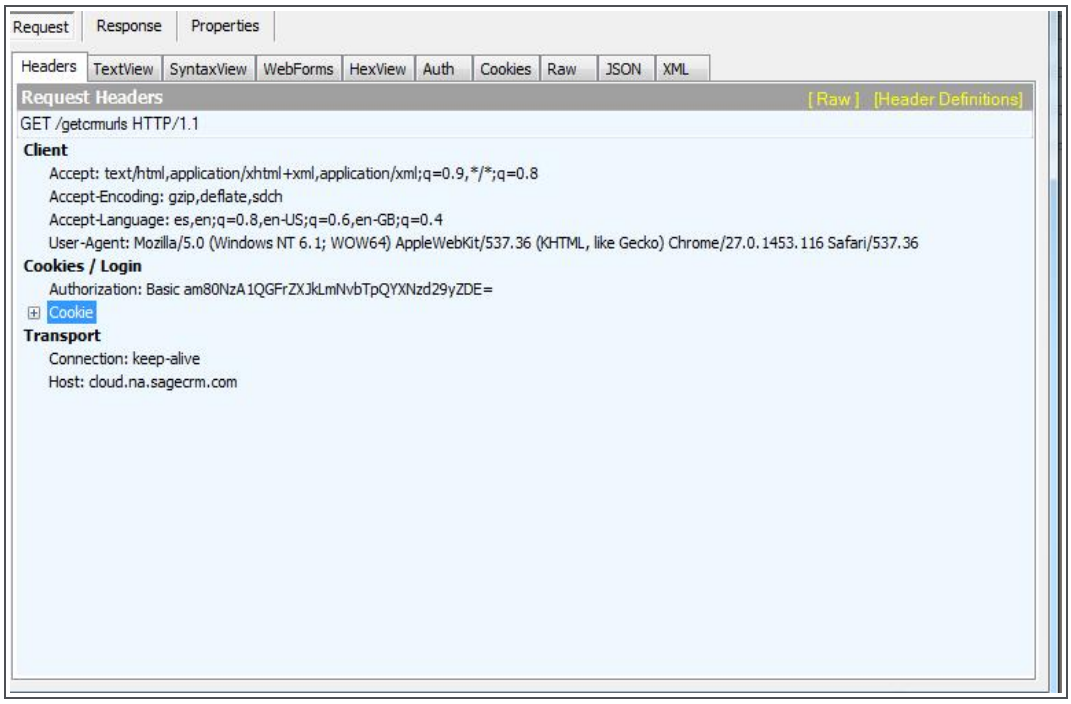
Nach der erfolgreichen Authentifizierung beim Endpunkt von GetCRMURL wird Ihr Sage CRM Cloud-System gestartet, sofern es nicht bereits auf der Cloud-Plattform ausgeführt wird. Falls Ihre Sage CRM Instanz derzeit nicht bereitgestellt ist, löst der Endpunkt eine Bereitstellung aus und ruft Ihre Webservice- oder SData-URL für eine Sage CRM Cloud-Installation ab. Die Authentifizierung erfolgt dabei mit Ihrem Sage CRM Cloud-Benutzernamen und -Passwort. Sie können diese Endpunkte nicht im Browser verwenden, da Sie Ihren Benutzernamen und Ihr Passwort in einem Authentifizierungs-Header übergeben müssen.

Entwickler sollten diese API verwenden, um die WSDL-Adresse für Ihre Webdienste während der Anmeldung abzurufen, damit sichergestellt wird, dass Ihr Sage CRM Cloud-Konto dazu bereit ist, Webdienstanforderungen anzunehmen.

Das folgende Beispiel zeigt, wie Sie eine Authentifizierungsanforderung durchführen:

Hinweis:

- Sie müssen sich beim Endpunkt mit einem standardmäßigen, einfachen Authentifizierungs-Header authentifizieren.
- Die Authentifizierung für diesen Endpunkt wird in den Kopfzeilen der Anforderung übermittelt.
- Benutzername und Passwort müssen das Format #Benutzername#:#Passwort# haben.
- Der Authentifizierungstyp ist "Basic", und als Codierung muss Base64 verwendet werden. Das Format ist: `Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==`



Anforderungskopfzeile für eine HTTP-Authentifizierungsanforderung

Es folgt eine Beispiel-Antwort auf die erfolgreiche Authentifizierung beim GetCRMURLs-Endpunkt. Die Antwort liegt in JSON-Format vor:

```
{ "
installName":"<myCustomerID>",

"wsdlUrl":"https://cloud.na.sagecrm.com/<myCustomerID>/eware.dll/webservices/CRMwe
bservice.wsdl",

"wsUrlConnection":"https://cloud.na.sagecrm.com/<myCustomerID>/eware.dll/webservic
es/",
"web2LeadUrl":"https://cloud.na. sagecrm.com/<myCustomerID>/eware.dll/SubmitLead",
"sDataUrl":"https://cloud.na. sagecrm.com/sdata/<myCustomerID>j/sagecrm/",
"edition":"professional",
"domain":"cloud.na.sagecrm.com",
"userId":1,
"teamId":1,
"found":true,
"userAuthenticated":true,
"userWebServicesEnabled":true,
"userDisabled":false
}
```

Hinweis: Wenn sich Ihr Cloud-Konto im europäischen Rechenzentrum befindet, verwenden Sie den EU-Endpunkt. Ist Ihr Konto dem Rechenzentrum für Nordamerika zugeordnet, verwenden Sie bitte den NA-Endpunkt. Sollten Sie Ihre Anforderung an den falschen Endpunkt senden, wird sie zum richtigen Rechenzentrum umgeleitet, also dem Rechenzentrum, in dem sich Ihr Sage CRM Cloud-Konto befindet. Bei Verwendung des richtigen Endpunkts erhalten Sie jedoch eine schnelle Antwort von der Sage CRM Plattform und gewährleisten damit den ordnungsgemäßen Betrieb Ihrer Anwendung.

Beispiel-Soap-Anfrage zur Anmeldung

Beim Erstellen einer Anwendung, die die Webservices-API von Sage CRM Cloud verwendet, ist es wichtig, den Parameter "AllowAutoRedirect" auf "Wahr" einzustellen, damit die Anwendung interne HTTP-Umleitungen verarbeiten kann, die eventuell innerhalb der Sage CRM Cloud Plattform auftreten.

In diesem C#-Beispiel ist dargestellt, wie Sie sich beim Server anmelden:

```
//Eine Instanz des Webservice.
private static WebService binding = null;
//Persistent für die Dauer des Programms; Anmeldeergebnisse beibehalten
private static logonresult SID = null;
private static void LogonToCRMSystem()
{
try
       {
               HttpWebRequest  request  = (HttpWebRequest)
               WebRequest.Create
("http://cloud.sagecrm.com/myCustomerID/eware.dll/webservices/CRMwebservice.wsd
l");
               HttpWebResponse response = (HttpWebResponse)request.GetResponse();

               //Automatische Umleitung von Webservices-Anmeldeanforderungen zur CRM Instanz
der Kunden
               binding.AllowAutoRedirect = true;
               SID = binding.logon("admin", "");
               binding.SessionHeaderValue = new SessionHeader();
               binding.SessionHeaderValue.sessionId = SID.sessionid; //Persistente SID
               return true;
       }
       catch (SoapException e)
       {
               Write(e.Message);
       }
       catch (Exception e)
{
               Write(e.Message + "\n" + e.StackTrace);
       }
}
```

Dies ist die XML-Anfrage, die Webservices verarbeitet:

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <soap:Body>
                <logon xmlns="http://tempuri.org/type">
                        <username>admin</username>
                        <password />
                </logon>
        </soap:Body>
 </soap:Envelope>
```

Beispiel-Soap-Anfrage zur Abmeldung

In diesem XML-Beispiel ist dargestellt, wie Sie sich abmelden:

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema">
       <soap:Header>
               <SessionHeader xmlns="http://tempuri.org/type">
                       <sessionId>57240080053832</sessionId>
               </SessionHeader>
       </soap:Header>
       <soap:Body>
               <logoff xmlns="http://tempuri.org/type">
                       <sessionId>57240080053832</sessionId>
               </logoff>
       </soap:Body>
</soap:Envelope>
```

Beispiel-Soap-Anfrage zum Löschen

Anhand dieses C#-Beispiels wird gezeigt, wie Sie eine Firma mit der ID 66 löschen:

```
ewarebase[] idList = new ewarebase[1];
companyid aCompanyId = new companyid();
aCompanyId.companyid1 = 66;  //66 ist ID der zu löschenden Firma
idList[0] = aCompanyId;
deleteresult aResult = binding.delete("company",idList);

if(aResult.deletesuccess == true)
       Console.WriteLine("Nummer wurde erfolgreich gelöscht: " + aResult.numberdeleted);
```

Dies ist die XML-Anfrage, die Webservices verarbeitet:

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:xsd="http://www.w3.org/2001/XMLSchema">
       <soap:Header>
               <SessionHeader xmlns="http://tempuri.org/type">
                       <sessionId>127169567253830</sessionId>
               </SessionHeader>
       </soap:Header>
       <soap:Body>
               <delete xmlns="http://tempuri.org/type">
                       <entityname>company</entityname>
                       <records xsi:type="companyid">
                               <companyid>66</companyid>
                       </records>
               </delete>
       </soap:Body>
</soap:Envelope>
```

Beispiel-Soap-Anfrage zum Aktualisieren

Anhand dieses C#-Beispiels wird gezeigt, wie Sie einem Firmennamen für eine Firma mit der ID 66 ändern:

```
private static void UpdateACompany()
{
```

```
	String idString = "66";
	String newName = "neuerName";
	ewarebase[] companyList = new ewarebase[1];//kann eine Reihe von Firmen
aktualisieren
	company aCompany = new company();
	aCompany.companyid = Convert.ToInt16(idString);
	aCompany.companyidSpecified = true;

	aCompany.name = neuerName;
	companyList[0] = aCompany;
	updateresult aresult = binding.update("company", companyList);

	if(aresult.updatesuccess == true)
	{}
else
	{}
}
```

Dies ist die XML-Anfrage, die Webservices verarbeitet:

```
<?xml version="1.0" encoding="utf-8" ?>
	<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Header>
	<SessionHeader xmlns="http://tempuri.org/type">
		<sessionId>12663708753831</sessionId>
	</SessionHeader>
</soap:Header>
<soap:Body>
	<update xmlns="http://tempuri.org/type">
		<entityname>company</entityname>
		<records xsi:type="company">
			<people xsi:nil="true" />
			<address xsi:nil="true" />
			<email xsi:nil="true" />
			<phone xsi:nil="true" />
			<companyid>933</companyid>
			<name>Design Wrong Inc</name>
		</records>
	</update>
</soap:Body>
</soap:Envelope>
```

Beispiel-Soap-Anfrage zum Abfragen einer Entität

In diesem Beispiel wird ein Datensatz einer Firma mit der ID 66 abgefragt:

```
company aCompany = (company) binding.queryentity( 66, "Firma").records;
```

Beispiel-Soap-XML zur Darstellung einer Firma

Die folgende XML stellt eine Firma mit der ID 65 dar:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
	xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
```

```
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<SOAP-ENV:Body>
<queryentityresponse xmlns="http://tempuri.org/type">
<result>
<records xsi:type="typens:company" mlns:typens="http://tempuri.org/type">
	<typens:companyid>65</typens:companyid>
	<typens:primarypersonid>79</typens:primarypersonid>
	<typens:primaryaddressid>77</typens:primaryaddressid>
	<typens:primaryuserid>9</typens:primaryuserid>
	<typens:name>AFN Interactive</typens:name>
	<typens:website>http://www.AFNInteractive.co.uk</typens:website>
	<typens:createdby>1</typens:createdby>
	<typens:createddate>2004-08-30T18:10:00</typens:createddate>
	<typens:updatedby>1</typens:updatedby>
	<typens:updateddate>2004-08-30T18:10:00</typens:updateddate>
	<typens:timestamp>2004-08-30T18:10:00</typens:timestamp>
	<typens:librarydir>A\AFN Interactive(65)</typens:librarydir>
	<typens:secterr>-1845493753</typens:secterr>
	<email>
		<entityname>email</entityname>
		<records xsi:type="typens:email" xmlns:typens="http://tempuri.org/type">
			<typens:emailid>120</typens:emailid>
			<typens:companyid>65</typens:companyid>
			<typens:type>Sales</typens:type>
			<typens:emailaddress>sales@AFNInteractive.co.uk</typens:emailaddress>
			<typens:createdby>1</typens:createdby>
			<typens:createddate>2004-08-30T18:10:00</typens:createddate>
			<typens:updatedby>1</typens:updatedby>
			<typens:updateddate>2004-08-30T18:10:00</typens:updateddate>
			<typens:timestamp>2004-08-30T18:10:00</typens:timestamp>
		</records>
	</email>
	<phone>
		<entityname>phone</entityname>
		<records xsi:type="typens:phone" xmlns:typens="http://tempuri.org/type">
			<typens:phoneid>211</typens:phoneid>
			<typens:companyid>65</typens:companyid>
			<typens:type>Business</typens:type>
			<typens:countrycode>44</typens:countrycode>
			<typens:areacode>208</typens:areacode>
			<typens:number>848 1051</typens:number>
			<typens:createdby>1</typens:createdby>
			<typens:createddate>2004-08-30T18:10:00</typens:createddate>
			<typens:updatedby>1</typens:updatedby>
			<typens:updateddate>2004-08-30T18:10:00</typens:updateddate>
			<typens:timestamp>2004-08-30T18:10:00</typens:timestamp>
		</records>
	</phone>
	<address>
		<entityname>address</entityname>
		<records xsi:type="typens:address" xmlns:typens="http://tempuri.org/type">
			<typens:addressid>77</typens:addressid>
			<typens:address1>Greenside House</typens:address1>
			<typens:address2>50 Station Road</typens:address2>
			<typens:address3>Wood Grn</typens:address3>
			<typens:city>LONDON</typens:city>
			<typens:postcode>N22 7TP</typens:postcode>
			<typens:createdby>1</typens:createdby>
			<typens:createddate>2004-08-30T18:10:00</typens:createddate>
			<typens:updatedby>1</typens:updatedby>
```

```
                        <typens:updateddate>2004-08-30T18:10:00</typens:updateddate>
                        <typens:timestamp>2004-08-30T18:10:00</typens:timestamp>
                </records>
        </address>
</records>
</result>
</queryentityresponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

CRM Dokumentversionscode: CLO-WES-DEU-161-1.0