

# Design-Konzept für Web Services mit Ivory™ (Top-Down und Bottom-Up Design-Optionen)

---



Composite Business Services sind *neue Services*, die durch die nahtlose Kombination bestehender Anwendungen, ohne vorgenommene Programmänderungen erstellt werden. Diese Composite Business Services können Geschäftsprozesse ausführen, für die einzelne Anwendungen nie erstellt wurden. *Composite Mainframe Services* können viele verschiedene Mainframe-Transaktionen und Datenquellen in einer einzigen Definition bündeln. Diese Composite Mainframe Services können wiederum in höherliegenden Composite Business Services integriert werden. Mit der Verfügbarkeit von Web Services auf der Mainframe ist es nun möglich, diese Composite Business Services als WSDL für die Nutzung mit jedem Web Service Client bereitzustellen.

Es gibt zwei Ansätze zum Design dieser Composite Business Services: Top-Down und Bottom-Up.

Der *Bottom-Up*-Ansatz sieht zuerst auf die verfügbaren Transaktionen und Daten. Dann werden die gesamten Transaktionen oder Datenquellen mit einer Schnittstelle versehen, durch die sie den Clients zur Verfügung gestellt werden. Dies ist eine eins-zu-eins Abbildung der Transaktion zu einer Web Service-Schnittstelle ohne inhärente Kombinationen von Funktionen. Dieser Ansatz kann auf dem Level von COBOL und/oder COPYBOOKs starten und diese als Grundlage des Designprozesses verwenden. Hierdurch würden alle Daten den Clients zur Verfügung stehen (möglicherweise als WSDL) aber die Fähigkeit die Komponenten in einem Composite Service zu kombinieren würde dem Web Service-Verbraucher überlassen (z.B. einem Applikationsserver der auf einer Mid-Tier Plattform läuft). Bei diesem Ansatz ist der Applikationsserver verantwortlich für die Koordination aller Daten und der Ablaufsteuerung zwischen den einzelnen Web Services und kann somit zum Engpass werden. Alle Daten zwischen den verschiedenen Komponenten werden durch den Applikationsserver übertragen und nicht unmittelbar vom Punkt der Generierung bis zum Punkt des Abnehmers.

Dieser Ansatz kann als ein erster Ausgangspunkt für das Design verwendet werden, da es sich um eine eins-zu-eins Abbildung der zugrunde liegenden Mainframe Ressourcen handelt.

Der *Top-Down*-Ansatz sieht zunächst auf die Business-Services, die spezifiziert oder angefragt wurden oder auf ein zuvor erstelltes WSDL. Dann wird geprüft, welche In- und Outputs benötigt werden um den Anforderungen dieses Services gerecht zu werden. In der Regel liefert der Business Analyst die Voraussetzungen, welche in diesem Bereich benötigt werden oder es besteht ein WSDL, das von einem Geschäftsprozess-Modelingtool erstellt wurde. Sobald diese Anforderungen festgelegt wurden, prüft der Designer ob bestehende Businesskomponenten vorhanden sind, die zur Erfüllung der Anforderungen notwendig sind. Möglicherweise sind mehrere Komponenten (Transaktionen oder Datenquellen) notwendig, um die Anfrage zu beantworten. Nachdem diese Geschäftskomponenten identifiziert und spezifiziert wurden, werden sie in einer Definition einschliesslich der dazugehörigen Geschäftsbewegungen zusammengefasst. Dann werden die zugrunde liegenden Komponenten in der erforderlichen Logik des Geschäftsflusses abgebildet um die Anfrage zu vollenden. Diese zusammengesetzte Business Service Definition wird als eine einzige WSDL publiziert, welche von WSDL-

Clients aufgerufen werden kann. Der anfordernde Client stellt eine Anfrage an den Business Service, dieser stellt die Antworten mehrerer Businesskomponenten zusammen und liefert ein Ergebnis. Die Koordinierung dieser Businesskomponenten werden auf dem Host-System durchgeführt – es ist weder Endnutzerwissen nötig noch wird Kenntnis über die Mid-Tier-Plattform benötigt. Es gibt keinen zentralen Koordinator, an dem es zu potentiellen Engpässen kommen könnte. Das Verteilen der Daten reduziert die Netzwerklast und verkürzt die Transferzeiten. Wenn der Mainframe der Provider des Composite Business Services ist, gibt es nur einen einzigen Aufruf – das bedeutet eine Verringerung der Netzwerklatenz, der TCP/IP Stack Zeit, kürzere Zeit für die Erstellung der Session und weitere Zeitersparnis, da die Daten nur einmal pro Anfrage zusammengestellt werden müssen und nicht für jede Teilanfrage einzeln.

### Bottom-Up-Design im Ivory Studio

Mit dem Ivory Studio und dem *“New Project Wizard”* wird der Benutzer durch die notwendigen Schritte zur Erstellung eines Web Services geführt. Der Benutzer gibt den Namen des Projekt, den Namen des Web Services und die Adresse des Ivory Servers an. Nach der Eingabe des Web Service-Operations-Namens, liefert der Benutzer die COBOL-Copybooks und die Namen der CICS-Programme, welche im Service genutzt werden sollen. Die Copybooks können entweder auf dem PC liegen oder aber der Designer greift direkt über den Mainframe auf diese zu. Sobald die COMMAREA eingegeben wurde, hat der Nutzer die Möglichkeit auszuwählen, welche Felder im SOAP-Input und Output als Anforderungen genutzt werden sollen – die Angabe aller verfügbaren Felder ist dabei nicht nötig – es werden nur die Felder ausgewählt, die zur Beantwortung der Anfrage nötig sind. Die ausgewählten COBOL Namen werden normalisiert und zu lesbaren .NET/J2EE Namen, die je nach Wahl des Nutzers umbenannt werden. Nach diesem Input generiert das Tool das Projekt, welches damit zum Einsatz auf dem Ivory Server bereit ist.

### Top-Down-Design im Ivory Studio

Im Ivory Studio gibt es zwei Varianten des Top-Down-Designs. Das erste ist das WSDL-first Design, bei dem WSDLs (die zuvor von einem Designer entwickelt wurden) als Ausgangspunkte zur Entwicklung eines Web Services verwendet werden. Die zweite Variante benutzt eine *Business Service Definition*, die die notwendigen In- und Outputs liefert, um ein Web Service WSDL zu erstellen.

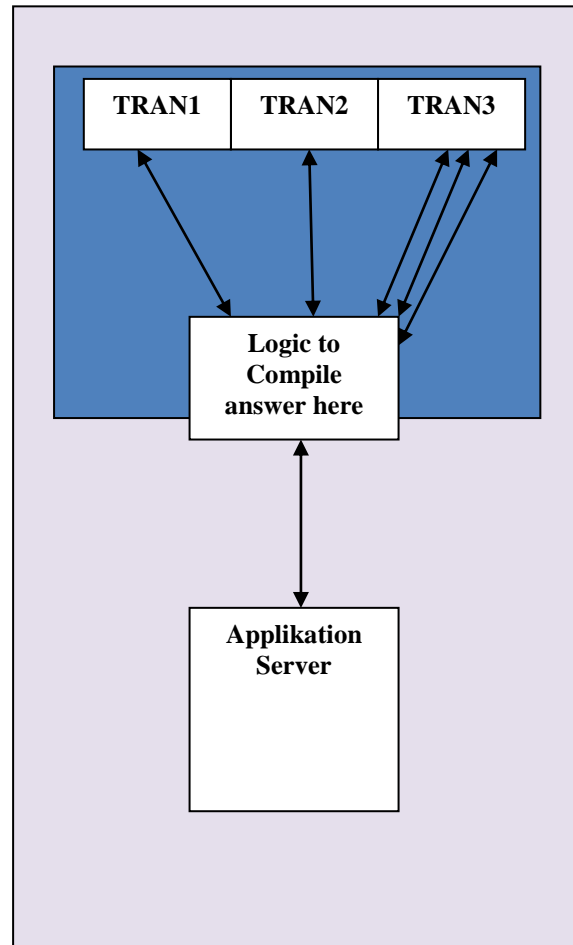
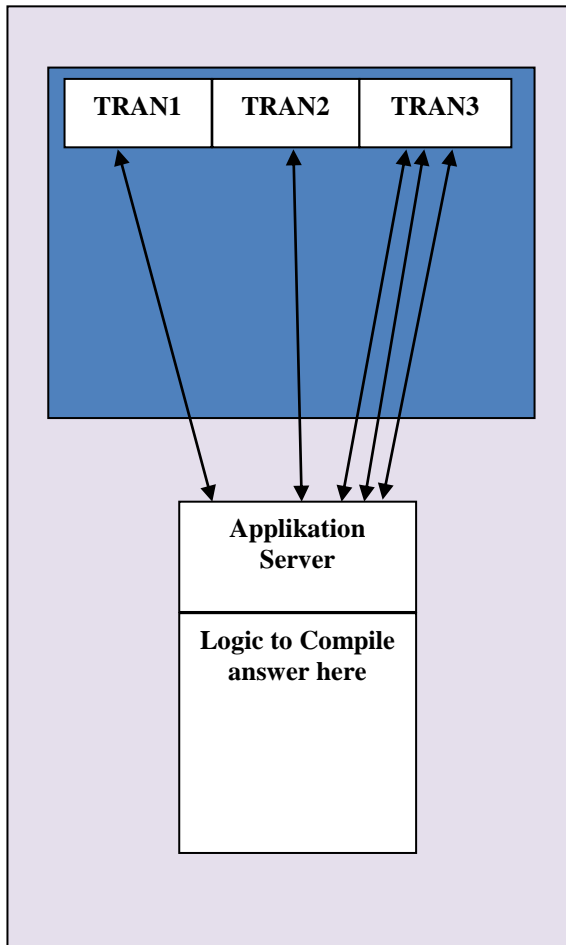
Beim WSDL-first Design öffnet der Benutzer ein neues Projekt im Ivory Studio und gibt diesem einen Namen. Dann wird mit einem WSDL-Discovery-Tool das UDDI oder WSDL-Verzeichnis durchsucht und das WSDL ausgewählt, welches im Web-Service verwendet werden soll. Das Ivory Studio liest alle WSDL-Metadaten und befüllt die In- und Output Beschreibungen mit Daten. Danach definiert der Designer die zugrundeliegenden notwendigen Mainframeprogramme die ausgeführt werden müssen um die Anforderungen zu erfüllen.

Bei der Verwendung des *Business-Service-Designs*, startet der Benutzer zunächst mit einem leeren Projekt. Das Projekt wird benannt sowie der Name des Web Services zugewiesen. Die SOAP In- und Outputs werden vom Nutzer, basierend auf der Definition

des Service-Designs, benannt und beschrieben. Dann werde die In- und Outputs, Copybooks und Programme benannt, die notwendig sind um die Anforderungen zu erfüllen. Die Anzahl der Programme, die zur Erfüllung der Aufgabe nötig sind, kann dabei beliebig groß sein - auch, ob es sich dabei um 3270- oder COMMAREA-Programme, oder eine Mischung aus beiden handelt, ist nicht relevant. Der Anwender definiert den SOAP-Inputs für die COMMAREA oder Screens und hat die Möglichkeit die Cobol-ähnlichen Namen in benutzerfreundlichere Namen für die .NET/J2EE Nutzer umzubenennen. Der Output wird zurückgegeben und der Nutzer kann diese Daten gegebenenfalls weiterleiten um dem SOAP-Output gerecht zu werden. Während dieser Prozesse und Schleifen können weitere Entscheidungspunkte mit einbezogen werden, um die Verarbeitung der Daten zu kontrollieren. Die SOAP Ergebnisse werden dann an den Aufrufer des Services zurück gesendet.

In beiden Fällen steht dem Anwender ein voll funktionsfähiges SOAP-Test-Tool zur Verfügung um die Web Services zu testen.

## Vergleich der Design-Methoden



### Mid-Tier Konsolidierung der Anfrage (Bottom-Up)

Beginnt mit COBOL COPYBOOK oder Code

Mehrere Anfragen an den Mainframe

Logik notwendig um Antworten off-Mainframe zu erstellen

Application Server wird Engpass für Antworten

TRAN3 ruft Screens mehrmals ab

Mehrere Schnittstellen für den Application Server um den Mainframe zu kontaktieren

### Composite Response der Anfrage (Top-Down)

Einstieg mit bereits erstellten WSDL oder Business Service-Anforderungen möglich

Einzelne Anfrage an den Mainframe

Logik notwendig um Antworten on-Mainframe zu erstellen

Application Server wird einziger Anforderer für Services

Eine Schnittstelle für Anwendungen um Composite Applications aufzurufen

## Ivory Web Services benutzen

Ivory Web Services geben dem Benutzer die Möglichkeit, Top-Down oder Bottom-Up Ansätze bei der Entwicklung der Web Services, die Composite Business Services generieren, zu nutzen. Der Ivory Server läuft direkt auf dem Großrechner und verringert damit die Netzwerkbelastung, die notwendig ist um die erforderlichen Antworten zu erhalten. Der Ivory Server kann mehrere Transaktionen und Datenquellen verarbeiten um die Geschäftsanforderungen, die vom Business Analysten in Ivory definiert wurden, zu erfüllen. Ivory Studio ermöglicht dem Entwickler, neue Businesslogik aus vorhandenen Applikationen zusammensetzen, ohne dass neuer Code geschrieben werden muss. Alle Interaktionen der Anwendung können mit Verknüpfungssteuerung (Entscheidungspunkte, Schleifen, etc.) definiert werden ohne dass dabei die Anwendungen verändert werden müssen oder Kenntnisse über die genaue Funktion der einzelnen Anwendungen vorliegen müssen.

Ivory Studio ermöglicht die Erstellung von Prozessabläufen, welche auf neu-zusammengesetzter Funktionalität beruhen, die von bestehenden Ressourcen angetrieben werden. Mit Ivory Studio kann ohne zusätzlichen Programmieraufwand eine Lösung erstellt werden, für die es im Normalfall nötig gewesen wäre, zusätzliche Logiken zu programmieren. Jegliche Logik im Ivory Studio wird per Drag-and-Drop Technologie definiert und dann auf dem Ivory Server veröffentlicht. Der gesamte Composite Business-Service wird durch das generierte WSDL vertreten, welches im Ivory Server bereitgestellt wird. Er kann von jedem Web Service, der diesen Composite Business Service benötigt, aufgerufen werden.

Ivory Studio kann auch verwendet werden, um komplexe Composite Web Services, Outbound Web Services (Web Service Konsumenten) und Web Services, die Mainframe-Datenquellen miteinbeziehen zu erstellen und SOAP-Anfragen zu bedienen.

## Zusammenfassung

Die Verwendung von Composite Mainframe Services, die durch WSDL spezifiziert werden erlaubt es, Clients verschiedenste Mainframeressourcen zugänglich zu machen, ohne sich Gedanken darüber machen zu müssen, wie die einzelnen Teilanwendungen funktionieren. Durch eine einzige, zusammengefasste Anfrage wird die Netzwerklatenz verringert und die Nutzung von Mainframeressourcen optimiert. Ein Top-Down Ansatz gibt Ihnen die Möglichkeit, In- und Outputs von verschiedensten Anwendungen, die auf der Mainframe laufen, zu definieren, sowie neue Businesslogik oder neue Prozessabläufe hinzuzufügen ohne sich von den Begrenzungen der bestehenden Anwendung eingeschränkt zu lassen. Ein Bottom-Up Ansatz ermöglicht die Nutzung der vorhandenen Programme (COBOL COPYBOOKS oder Code) als Startpunkt für die Gestaltung von Web Services. Die Verwendung des leistungsstarken Design Wizards ermöglicht auch die schnelle Erstellung eines Web Services mit einer Schritt-für-Schritt Herangehensweise, durch die der Benutzer geführt wird.

Zusammenfassend liefern Ihnen Ivory Web Services eine leistungsstarke, programmatische Lösung, ohne den sonst anfallenden Programmieraufwand.

**GT Software**

1314 Spring Street, NW  
Atlanta, Georgia 30309  
Telephone: +1 (404) 253-1300  
Fax: (404) 253-1314  
Web: [www.gtsoftware.com](http://www.gtsoftware.com)

**EDV-Beratung Machold GmbH  
Systemhaus 21**

Nordbahnhofstr. 17  
70191 Stuttgart  
Telefon: +49 (711) 25772-0  
Fax: +49 (711) 25772-22  
Web: [www.machold.de](http://www.machold.de)