Tutorial 8

Erstellen eines WSDL-Dokuments
mit anschließender Generierung eines Web Services unter z/OS

Dieses Tutorial beschreibt die Erstellung eines WSDL Dokuments mit Hilfe des WSDL Editors aus der Entwicklungsumgebung *Rational Developer for System z 7.0* (RDz 7.0). Im Anschluss wird diese Beschreibung zur Generierung eines Web Services unter z/OS dienen.
Der Web Service Tutorial7-Fibonacci erwartet eine ganze Zahl i zwischen einschließlich 0 und einschließlich 100 und liefert die Summe der Fibonacci-Zahlenfolge mit der oberen Grenze i.

Voraussetzungen

Dieses Tutorial setzt grundlegende Kenntnisse unter z/OS sowie RDz 7.0 voraus. Unter letzterem wurde eine ferne Verbindung zu z/OS konfiguriert (siehe Tutorial 6).

Erstellen eines WSDL-Dokuments

Im folgenden wird Schritt für Schritt die Erstellung eines WSDL-Dokuments unter Verwendung der Entwicklungsumgebung *Rational Developer for System z 7.0* (RDz 7.0) gezeigt.

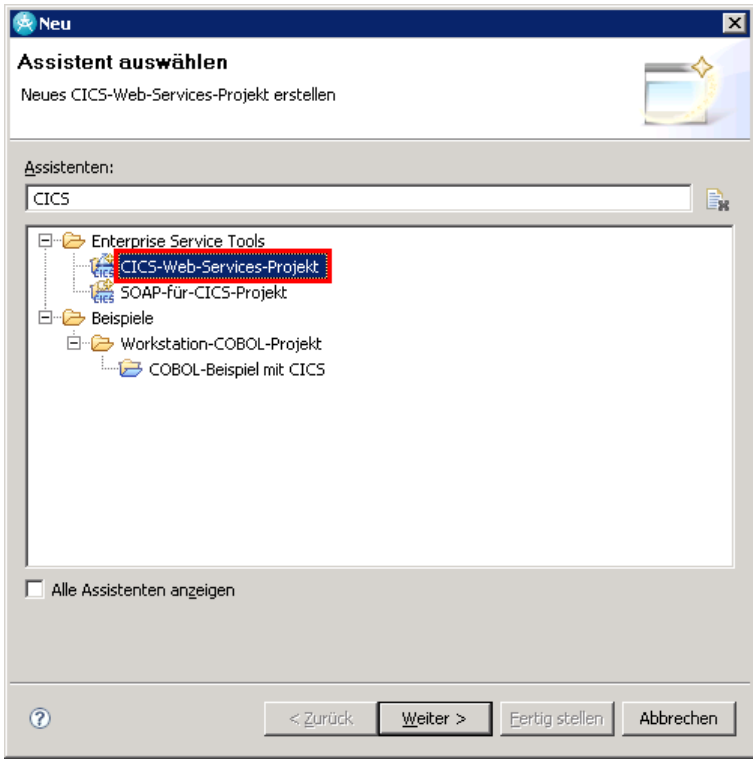

Abbildung 1: Neues CICS-Web-Services-Projekt erstellen

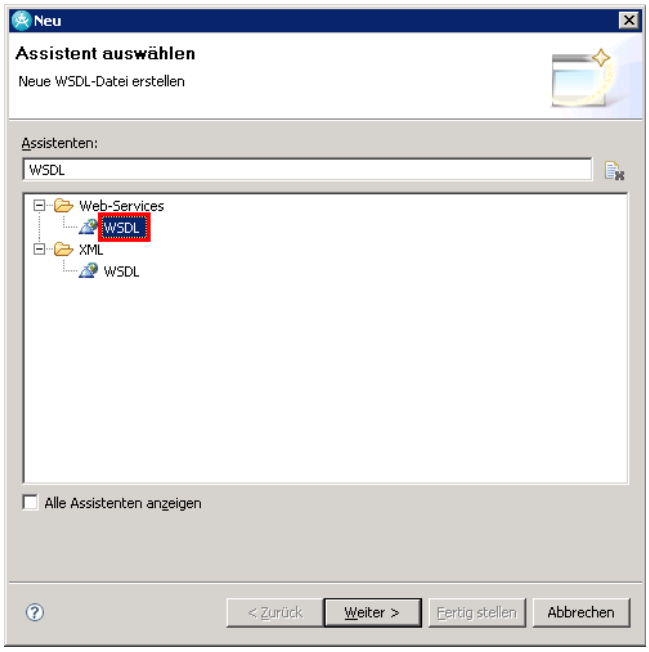

Abbildung 2: Neue WSDL-Datei

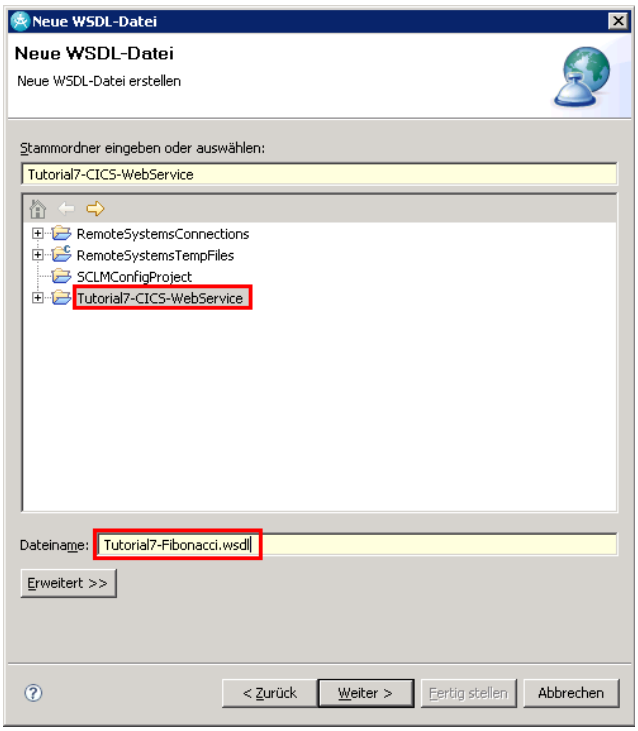

Abbildung 3: Neuer Dateiname für WSDL-Datei

Zuerst erstellen wir ein neues *CICS-Web-Services-Projekt*. Dazu wird der Menüpunkt

Datei → Neu → Andere...

aufgerufen. Anschließend schränkt man die Auswahl an Assistenten ein, durch die Eingabe von *CICS* (siehe Abb. 1). Nach der Auswahl von *CICS-Web-Services-Projekt* und einem Klick auf *Weiter*, gibt man den Projektnamen *Tutorial7-CICS-WebService* an und klickt auf *Fertig stellen*.

Wir sollten uns jetzt in der Perspektive *Enterprise Service Tools (EST)* befinden. Auf der linken Seite befindet sich der Projekt-Explorer von EST. Wir klicken unser eben erstelltes Projekt *Tutorial7-CICS-WebService* an, sodass es markiert ist und wählen darauf folgend den Menüpunkt

Datei → Neu → Andere...

Auch hier schränken wir wieder die Auswahl an Assistenten ein, indem wir *WSDL* eingeben (siehe Abb. 2). Anschließend wählt man *WSDL* unterhalb des Eintrags *Web-Services* aus und klickt auf *Weiter*. Im darauf folgenden Dialog wählt man den Stammordner *Tutorial7-CICS-WebService* aus und vergibt den WSDL-Dateinamen *Tutorial7-Fibonacci.wsdl* (siehe Abb. 3). Ein Klick auf *Weiter* bringt uns zum nächsten Dialog. Darin ändern wir den Zielnamensbereich (Namespace) zu http://www.informatik.uni-leipzig.edu/Tutorial7-Fibonacci/ und klicken auf *Fertig stellen*.

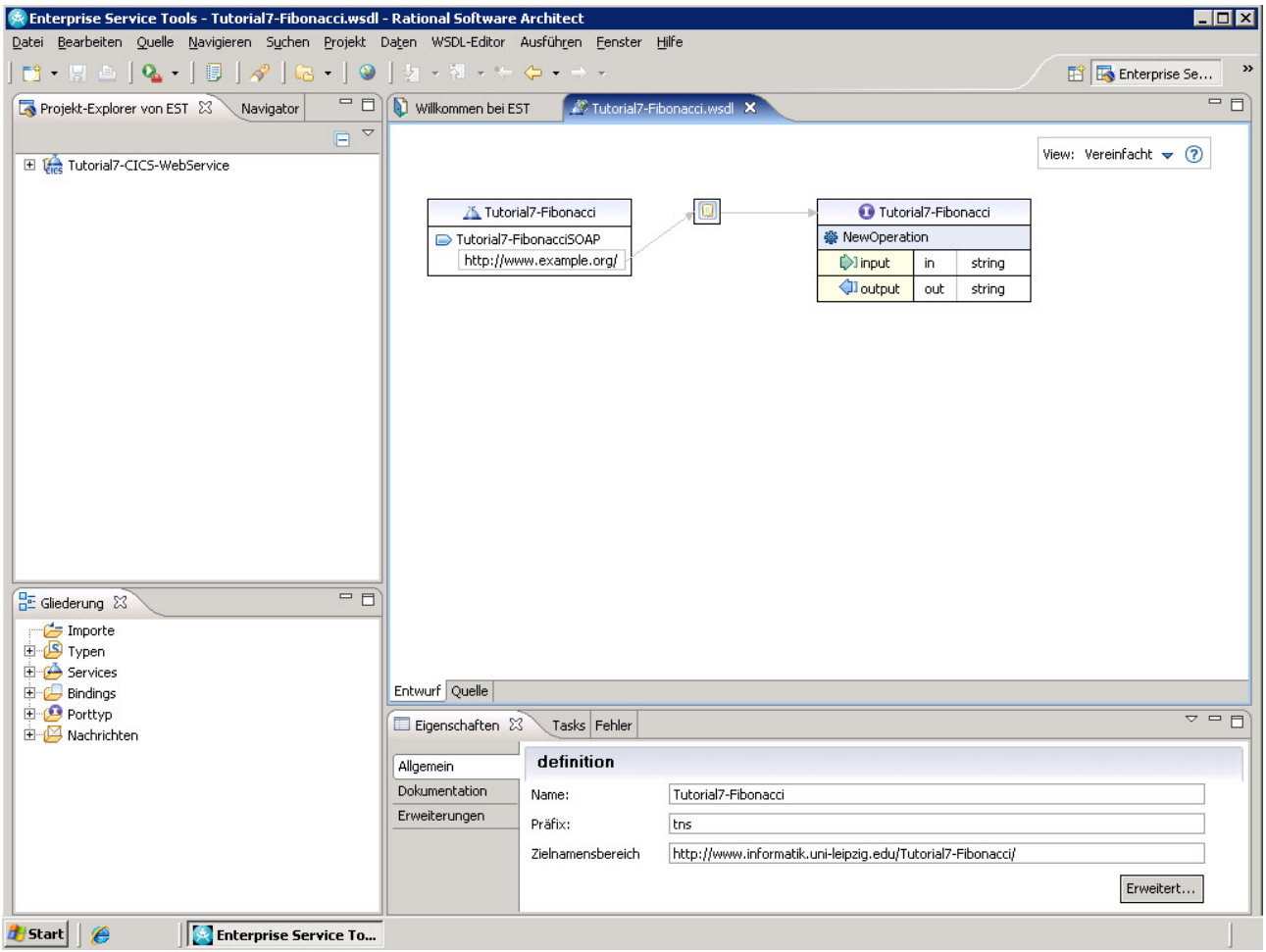

Abbildung 4: Der WSDL-Editor

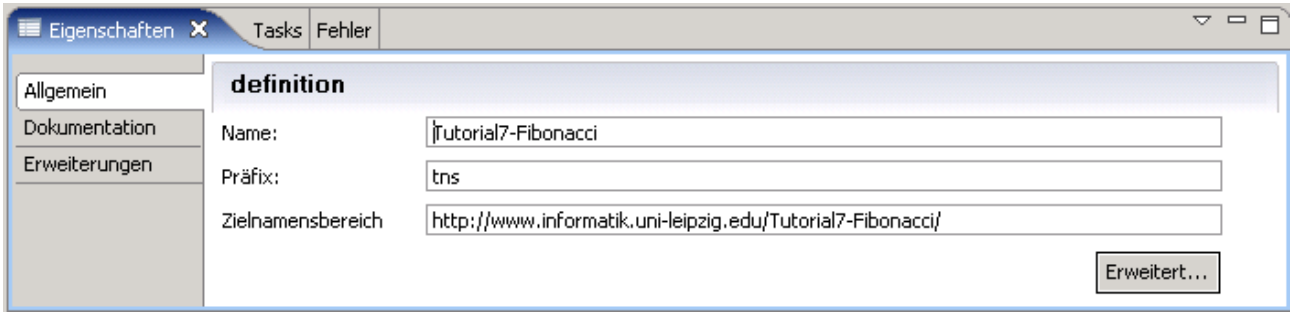

Abbildung 5: Eigenschaften des WSDL-Dokuments

Jetzt öffnet sich der WSDL-Editor und wir sehen eine grafische Repräsentation der WSDL-Datei (siehe Abb. 4). Nach einem Klick auf den weißen Bereich innerhalb dieser, ist unterhalb der Reiterleiste *Entwurf, Quelle* der Reiter *Eigenschaften.* Der Reiter *Entwurf* enthält die grafische Sicht auf die WSDL-Datei. Der Reiter *Quelle* beinhaltet die textuelle Sicht (Quellcode). Der Reiter *Eigenschaften* enthält Eigenschaften eines ausgewählten Elements. Da wir zuvor die weiße Fläche angeklickt hatten, beinhaltet dieses die Eigenschaften des WSDL-Dokuments (siehe Abb. 5). Zuerst geben wir eine Beschreibung an. Dazu klicken wir innerhalb der Eigenschaften-Sicht auf *Dokumentation* und füllen das betreffende Textfeld wie folgt aus (siehe Abb. 6):

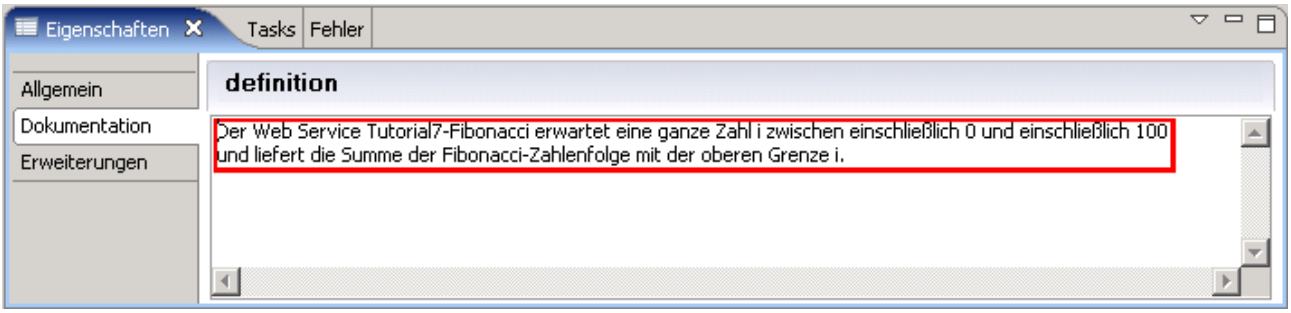

Abbildung 6: Dokumentation des WSDL-Dokuments erstellen

> *Der Web Service Tutorial7-Fibonacci erwartet eine ganze Zahl i zwischen einschließlich 0 und einschließlich 100 und liefert die Summe der Fibonacci-Zahlenfolge mit der oberen Grenze i.*

Im nächsten Schritt ändern wir den Namen der Operation. Dieser ist bis dato mit dem Wert *NewOperation* belegt. Dazu klicken wir einmal innerhalb der grafischen Sicht auf das Wort *NewOperation*. Im Eigenschaften-Reiter unter *Allgemein* kann nun der Name geändert werden zu *getFibonacciProduct* (siehe Abb. 7).

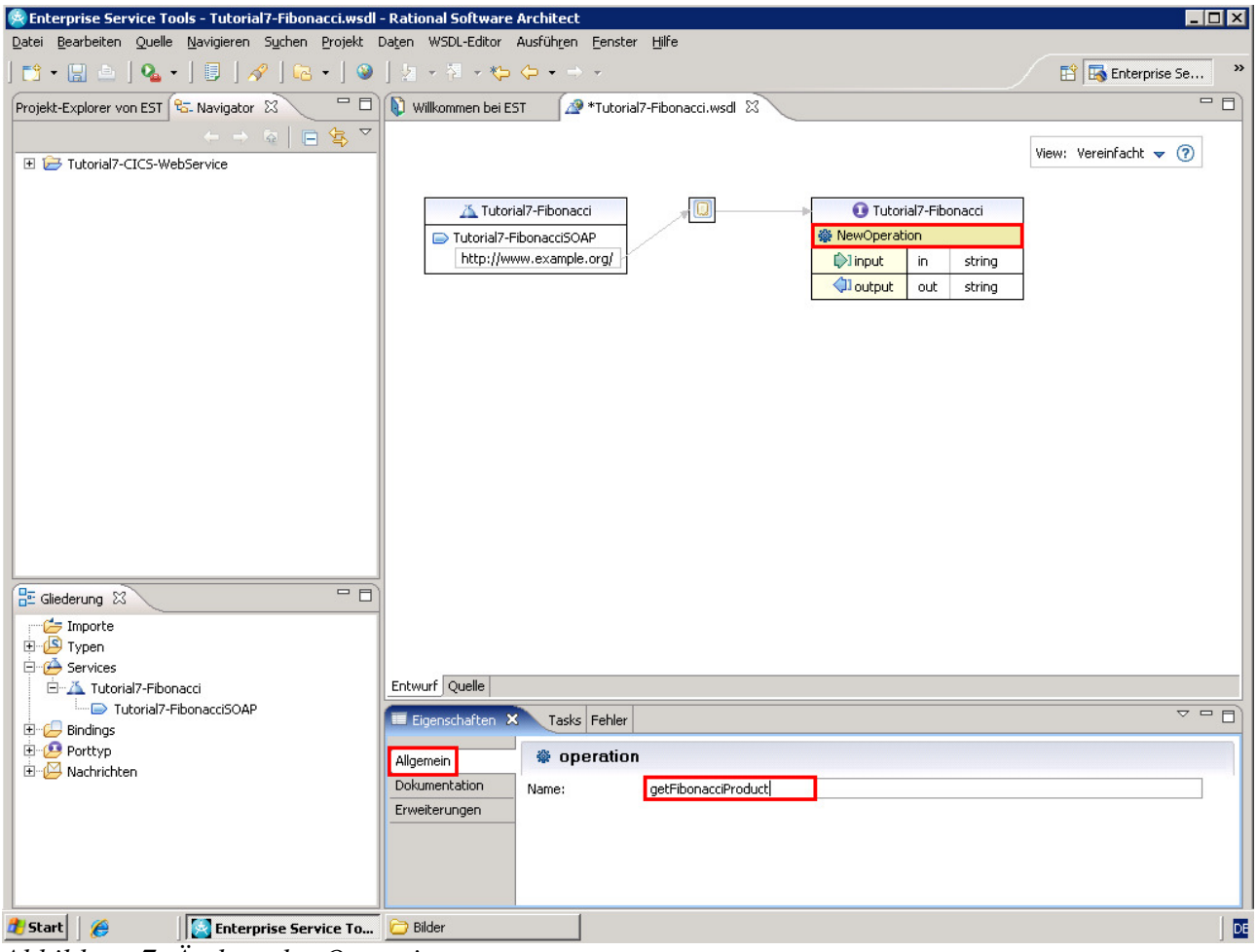

Abbildung 7: Ändern der Operation

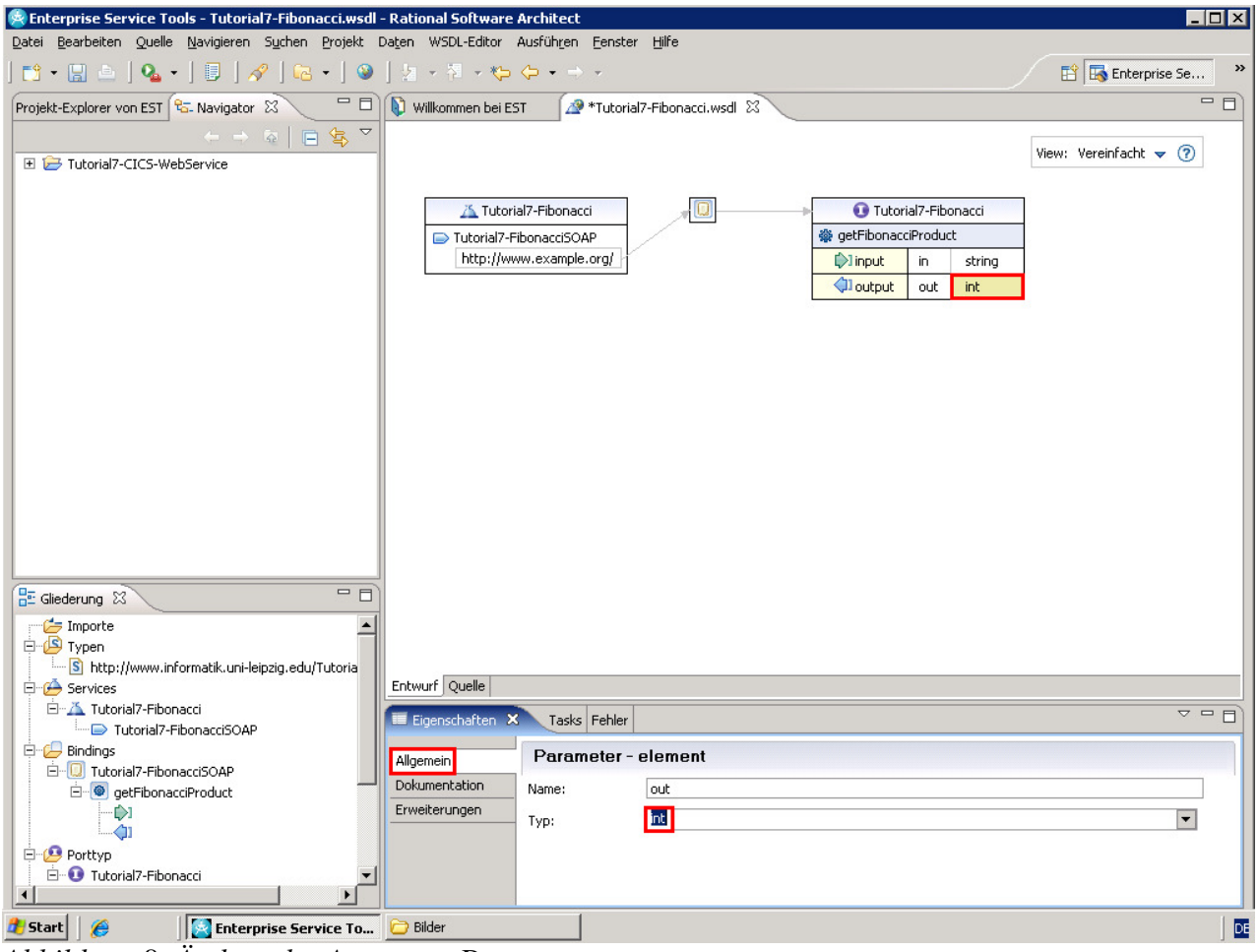

Abbildung 8: Ändern des Ausgangs-Datentyps

Als nächstes ändern wir die Datentypen. Der Web Service soll eine Integer-Zahl zwischen einschließlich 0 und 100 entgegennehmen und anschließend eine Integer-Zahl zurückgeben. Der Rückgabe-Datentyp wird geändert, indem auf das Wort *string* in der Zeile *output* geklickt wird. Im Reiter *Eigenschaften* unter *Allgemein* wechselt man den *Typ* von *string* zu *int* (siehe Abb. 8). Den Input-Datentyp *string* ändern wir genauso ab, nur das anstatt *int* jetzt *new...* gewählt wird. Wie in Abbildung 9 zu sehen ist, definieren wir einen *Einfachen Typ* mit dem Namen *intFib*. Ein Klick auf *Ok* legt diesen Datentyp neu an. Anschließend drücken wir in der grafischen Sicht den in Abbildung 10 rot umrandeten grauen Pfeil. Es öffnet sich ein weiteres Fenster mit einer grafischen Repräsentation eines eingebetteten Schemas unseres neuen Datentyps *intFib*. Da dieser auf dem Datentyp *int* basiert, ändern wir das Textfeld *Basistyp* innerhalb des Eigenschaften-Reiters unter *Allgemein* von *string* zu *int*. Anschließend wechseln wir von *Allgemein* zu *Integritätsbedingungen*

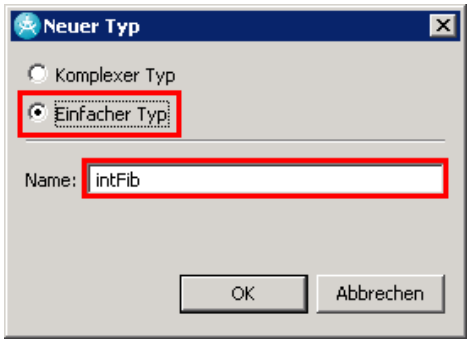

Abbildung 9: Neuer Datentyp

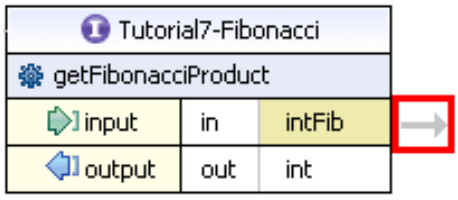

Abbildung 10: Datentypen bearbeiten

und füllen die Textfelder *Mindestwert* und *Maximalwert* inklusive der Optionsfelder wie in Abbildung 11. Das Fenster *Inline-Schema von Tutorial7-Fibonacci.wsdl* kann nach dem drücken von Strg+s (Speichern) geschlossen werden.

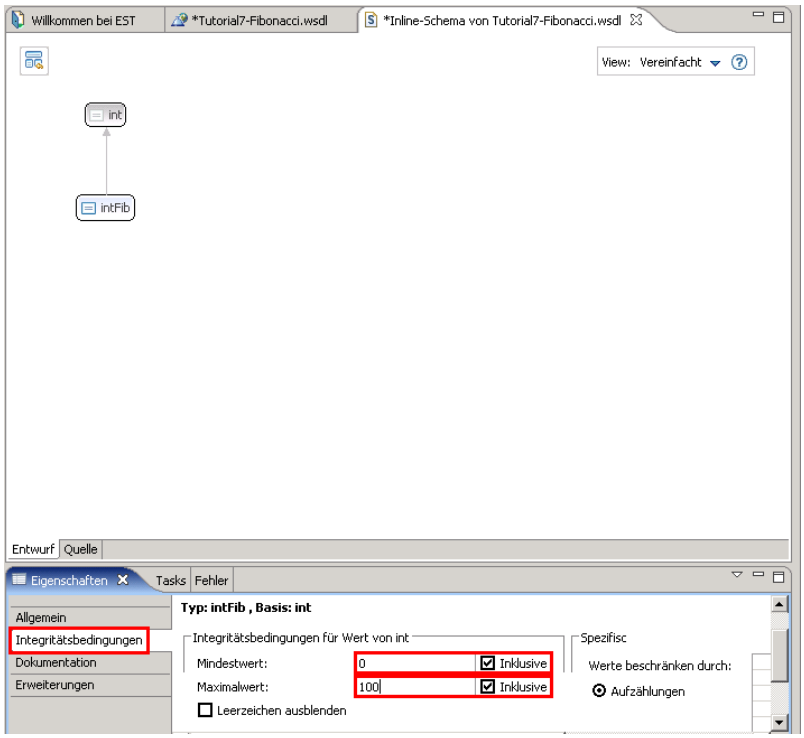

Abbildung 11: Integritätsbedingungen

Als letztes ändern wir noch die URL http://www.example.org zu http://139.18.4.34:3601/Tutorial7-Fibonacci/ indem wir diese anklicken und anschließend unter dem Reiter *Eigenschaften* das Textfeld *Adresse* dem entsprechend bearbeiten. Dabei handelt es sich um den Endpunkt des Web Services. Abbildung 12 zeigt dieses Feld sowie die endgültige grafische Darstellung, welche wir mit Strg+s speichern.

Aufgabe: Erstellen Sie die WSDL-Datei Tutorial7-Fibonacci.wsdl mit dem Endpunkt http://139.18.4.34:3601/cics/services/prak0XX/Fibonacci!

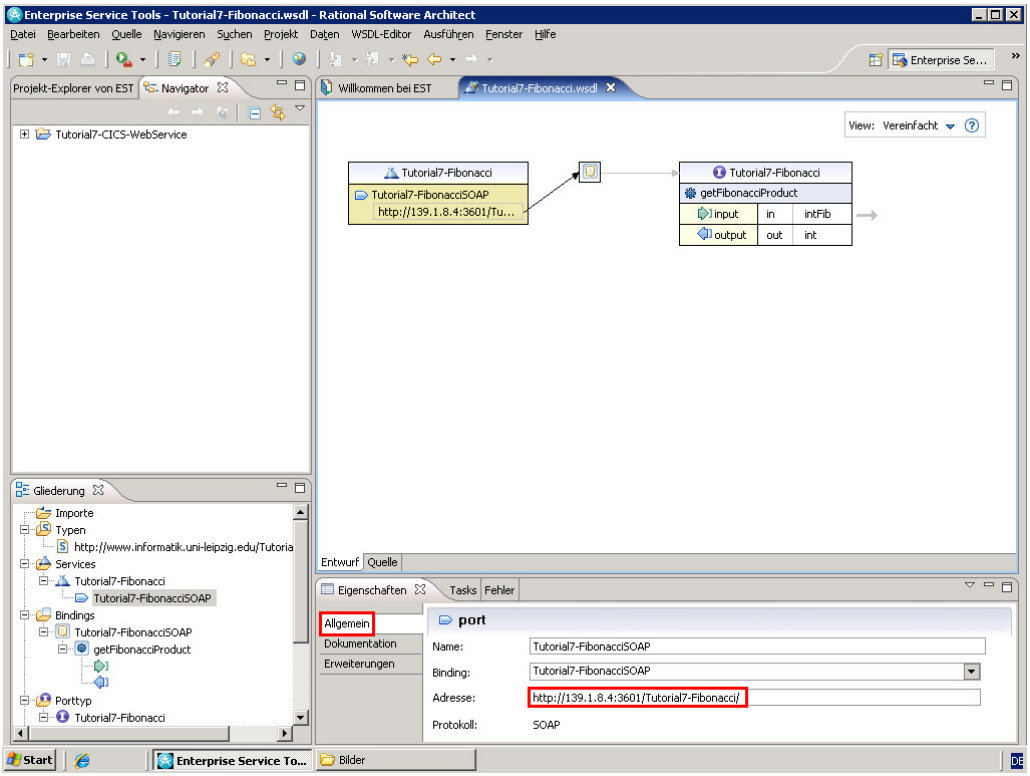

Abbildung 12: Port Adresse ändern

Im nächsten Schritt generieren wir die CICS Ressourcen. Dazu wechseln wir oben links zum *Navigator* und klicken mit rechts auf unsere WSDL-Datei Tutorial7-Fibonacci.wsdl. Im folgenden Menü wählen wir *Versetzen...* und im darauf folgenden Fenster den Ordner *Quelle* innerhalb unseres Projekts. Nach dem Klick auf *Ok* befindet sich unsere WSDL-Datei in dem Verzeichnis Quelle.

Jetzt klicken wir erneut innerhalb des Navigators auf unsere WSDL-Datei und wählen den Menüpunkt *Ressourcen für Web Services für CICS generieren*.

Jetzt startet ein Assistent (siehe Abb. 13) dessen Einstellungen wir nicht abändern. Wir drücken einfach auf den Button *Starten*. Es folgt ein weiterer Dialog in dem unter den beiden Reitern

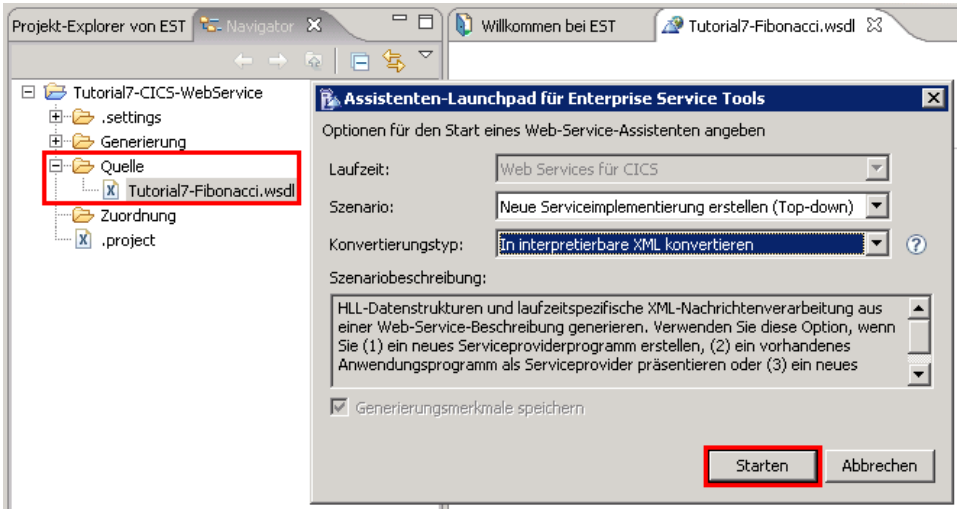

Abbildung 13: Assistent zur Generierung der CICS-Ressourcen

Anwendungsmerkmale und *Servicemerkmale* weitere Einstellungen wie in Abbildung 14 und 15

vorzunehmen sind. Danach klicken wir auf *Weiter*. Nachdem wie in Abbildung 16 unter dem Reiter *Schablone* das Textfeld *Name der Schablonendatei* angepasst wurde, klicken wir auf *Fertig stellen*.

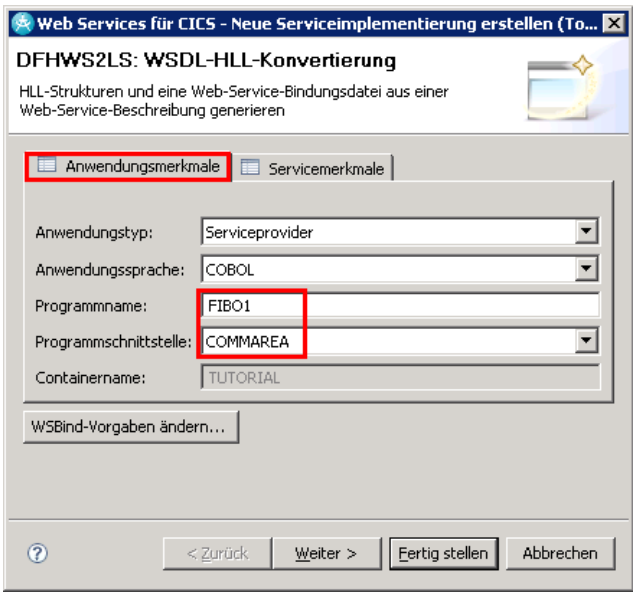

Abbildung 14: Assistent zur Generierung der CICS-Ressourcen

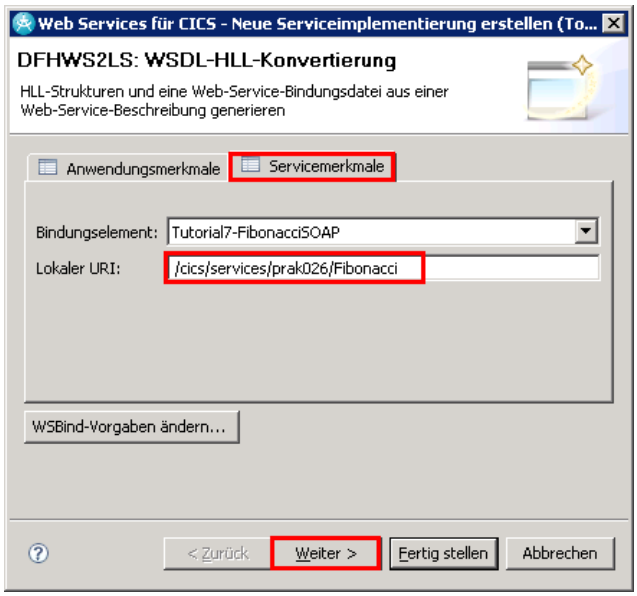

Abbildung 15: Assistent zur Generierung der CICS-Ressourcen

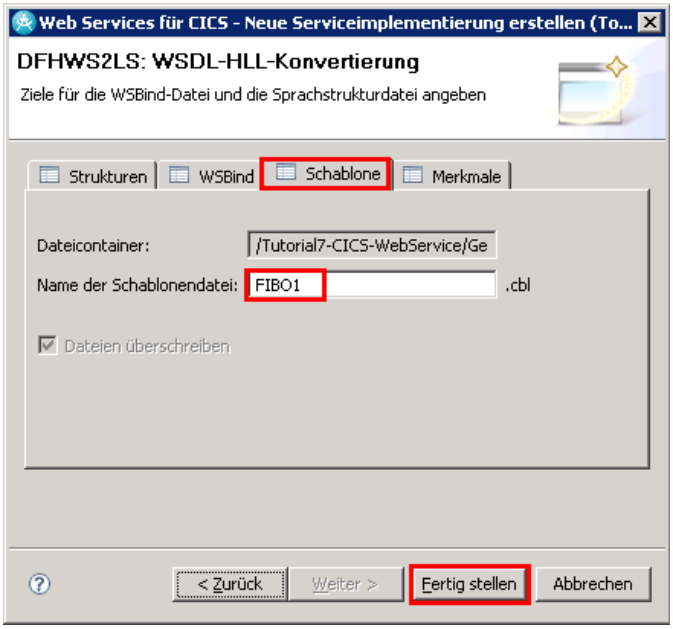

Abbildung 16: Assistent zur Generierung der CICS-Ressourcen

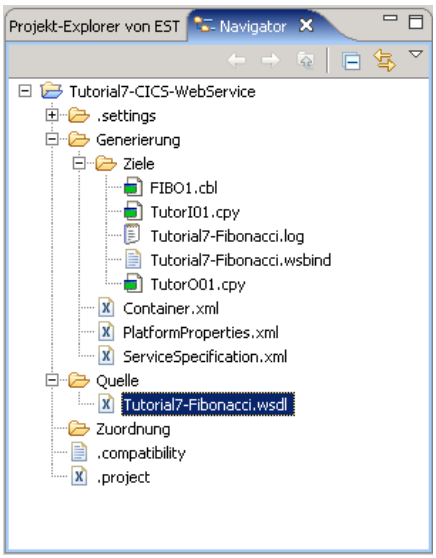

Abbildung 17: Erzeugte Dateien

Die erzeugten Dateien finden wir anschließend im Ordner *Generierung/Ziele*. Die Datei *FIBO1.cbl* ist das Cobol Programm mit den beiden Copybooks *TutorI01.cpy* und *TutorO01.cpy* (siehe Abbildung 17).

Im nächsten Schritt passen wir das Cobol Programm *FIBO1.cbl* an, indem wir es per Doppelklick

öffnen.

Aufgabe: Passen Sie das COBOL Programm FIBO1.cbl an unter Verwendung des folgenden Quellcodes.

```
PROCESS NODYNAM,NSYMBOL(NATIONAL),TRUNC(STD)
IDENTIFICATION DIVISION.
PROGRAM-ID. FIBO1.
AUTHOR. PRAK0XX.
DATA DIVISION.
WORKING-STORAGE SECTION.
01 SPCOUNT  PIC 9(8) VALUE ZERO.
77 Fib PIC 9(9) VALUE 1.
77 Fib-1 PIC 9(9) VALUE 0.
77 Fib-2 PIC 9(9) VALUE 0.
77 End-Flag PIC 9(1) VALUE 0.
77 RESULT PIC 9(9) VALUE 0.
77 RESULT2 PIC Z(8)9.
77 WSINP PIC 9(9) COMP-5 SYNC.
77 WSOUT PIC S9(9) COMP-5 SYNC.
LOCAL-STORAGE SECTION.
LINKAGE SECTION.
01 DFHCOMMAREA.
    05 getFibonacciProduct.
      10 Xin                          PIC S9(9) COMP-5 SYNC.
PROCEDURE DIVISION.
INIT.
    MOVE getFibonacciProduct TO WSINP.
    PERFORM UNTIL (End-Flag = 1)
    IF Fib IS NOT EQUAL TO Fib-1
    COMPUTE RESULT = RESULT + Fib
    END-IF
    MOVE Fib-1 TO Fib-2
    MOVE Fib TO Fib-1
    COMPUTE Fib = Fib-1 + Fib-2
    ON SIZE ERROR
    MOVE 1 TO End-Flag
    END-COMPUTE
    IF Fib GREATER THAN WSINP
    MOVE 1 TO End-Flag
    END-IF
    END-PERFORM.
    IF WSINP IS EQUAL TO ZERO
    MOVE 0 TO RESULT
    END-IF.
    MOVE RESULT TO RESULT2.
    INSPECT RESULT2 TALLYING SPCOUNT FOR LEADING SPACE.
    MOVE RESULT2(SPCOUNT + 1 : ) TO WSOUT.
    MOVE SPACES TO DFHCOMMAREA.
    MOVE WSOUT TO getFibonacciProduct.
    EXEC CICS RETURN
    END-EXEC.
END PROGRAM FIBO1.
```

Jetzt benötigen wir einige Partition Datasets. Diese werden unter RDz angelegt, indem man zuerst

in die Perspektive *Remote System Explorer*. wechselt. In der Sicht ferne Systeme klickt man dann mit rechts auf *MVS-Dateien* und wählt *PDS anlegen* aus. Den Dialog füllt man wie folgt aus:

- Kursname des Hosts: demomvs
- High Level Qualifier: PRAK0XX
- Dateigruppenname: CICS.COB

Aufgabe: Erstellen Sie innerhalb des RDz, falls noch nicht vorhanden, die PDS

- *PRAK0XX.CICS.COB,*
- *PRAK0XX.CICS.CNTL und*
- *PRAK0XX.CICS.LOAD.*

Kopieren Sie anschließend das COBOL Programm FIBO1 als Member in das Dataset PRAK0XX.CICS.COB.

Nach einem Klick auf *Fertigstellen* findet man unter *Meine Dateigruppen* das neu angelegte PDS. In dieses könnten nun einfach über den bekannten *Kopieren/Einfügen* Mechanismus Dateien (Members) hinzugefügt werden.

Auch neue Members lassen sich ganz einfach erstellen indem man mit rechts auf ein PDS seiner Wahl klickt und dann *PDS-Member erstellen...* wählt. Im folgenden Dialog muss nur noch der Name vergeben werden. Dieser wäre zum Beispiel für ein JCL *meinJob1.jcl*. Dieses könnte man dann theoretisch per Doppelklick öffnen und per SUB <ENTER> in der unteren Zeile ausführen (siehe Abbildung 18).

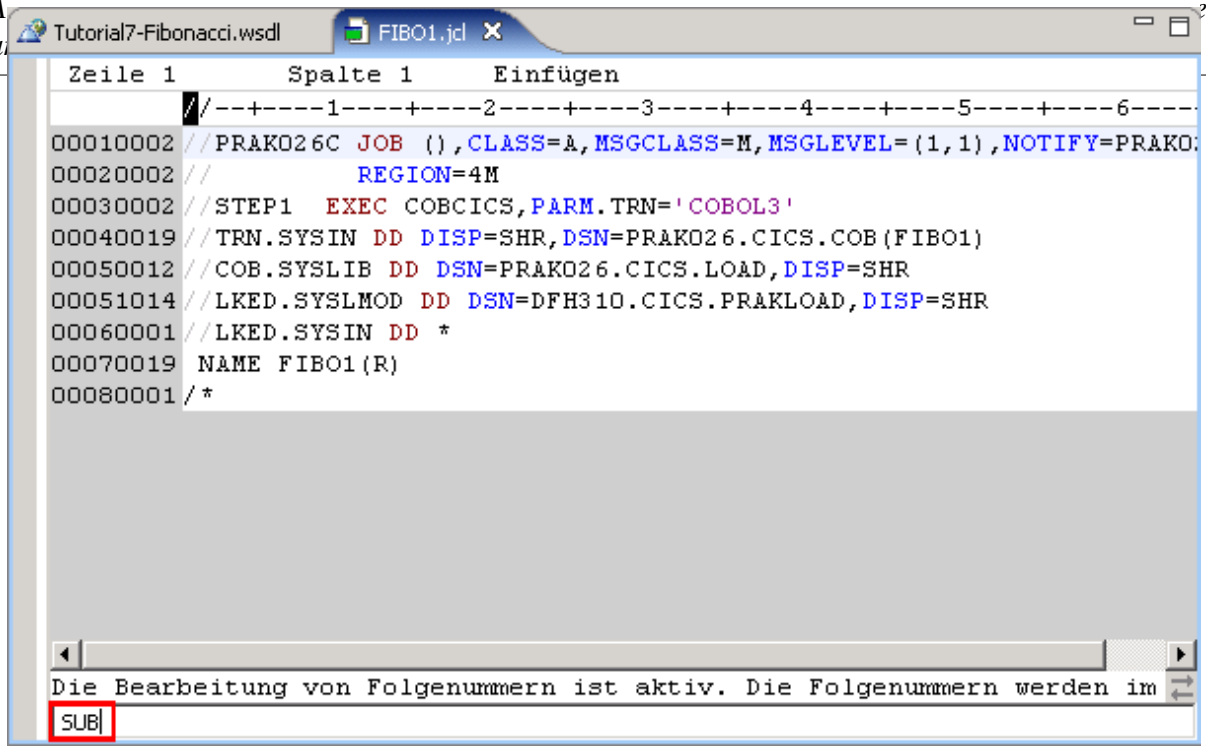

Abbildung 18: Ein JCL

Im nächsten Schritt legen wir eine Dateistruktur innerhalb des USS-Ausgangsverzeichnisses an.

Dazu klicken wir in der Sicht *Ferne Systeme* mit rechts auf *Mein Ausgangsverzeichnis* unterhalb von *demomvs* → *USS-Dateien* und wählen *Ordner*. Im folgenden Dialog vergeben Sie den Ordnernamen *services* und klicken auf *Fertig stellen*. Im nächsten Schritt legen wir unterhalb des letzten Verzeichnisses den Ordner *fibonacci* an. Darunter die beiden Ordner *shelf* und *wsbind*. Im Ordner *wsbind* erstellen wir den Ordner *provider*.

Jetzt können wir in den Ordner *provider* die beiden generierten Dateien *Tutorial7-Fibonacci.log* und *Tutorial7-Fibonacci.wsbind* kopieren.

Anschließend testen wir das Programm unter CICS, um zu überprüfen ob es vorhanden ist und korrekt funktioniert. Dazu klicken wir in der Sicht *Ferne Systeme* mit rechts auf *demomvs* und dann auf *Unterstützung für Hostverbindungsemulator*. Mit *l cics* <*ENTER*> gelangen Sie auf CICS. Mit den folgenden Kommandos definieren wir ein Programm *FIBO1* innerhalb einer Gruppe *FIBO1*.

```
CEDA DEFINE PROGRAM(FIBO1) GROUP(FIBO0XX)<ENTER>
```

Dabei ist zu beachten die Sprache *Le370* anzugeben. Anschließend installieren wir dieses Programm.

```
CEDA INSTALL PROGRAM(FIBO1) GROUP(FIBO0XX) <ENTER>
```

Jetzt können wir das Programm testen mit

```
CECI LINK PROGRAM(FIBO1) COMMAREA('000000005') LENGTH(9)<ENTER>
```

Sie sehen jetzt die Eingabe. Mit <ENTER> wechseln Sie immer zwischen der Eingabe und Ausgabe.

Eingabe:

```
STATUS:  ABOUT TO EXECUTE COMMAND                                NAME=
 EXEC CICS  LInk Program( 'FIBO1    ' )
  < COmmarea( '000000005' ) < Length( +00009 ) > < Datalength() > >
  < SYSid() >
  < SYNconreturn >
  < Transid() >
  < INPUTMSG() < INPUTMSGLen() > >
  < CHannel() >
```

Ausgabe:

```
STATUS:  COMMAND EXECUTION COMPLETE                              NAME=
 EXEC CICS  LInk Program( 'FIBO1    ' )
  < COmmarea( '000000011' ) < Length( +00009 ) > < Datalength() > >
  < SYSid() >
  < SYNconreturn >
  < Transid() >
  < INPUTMSG() < INPUTMSGLen() > >
  < CHannel() >
```

Im folgenden erstellen wir unter CICS die benötigten CICS-Ressourcen

- PIPELINE ,
- TCPIPSERVICE ,
- WEBSERVICE und
- URIMAP .

Die PIPELINE-Ressource stellt eine Verbindung her zu den Webservice Beschreibungen unter z/OS und dem Webservice. Sie beinhaltet Informationen zum verwendeten SOAP Standard und besitzt einen Namen (WSP0XX01) und liegt in der Gruppe (WSFIBO1). Erstellt wird sie mit dem

folgendem Kommando :

```
CEDA <ENTER>(WSP0XX01)
DEFINE PIPELINE
'-WSDIR(/u/prak0XX/services/fibonacci/wsbind/provider)-'<ENTER>
```

Als Ergebnis erhalten wir den folgenden Screen:

```
OVERTYPE TO MODIFY                                          CICS RELEASE = 0640
 CEDA  DEFine PIpeline(          )
  PIpeline     ==>
  Group        ==>
  Description  ==>
  STatus       ==> Enabled           Enabled | Disabled
  Configfile   ==>
  (Mixed Case) ==>
               ==>
               ==>
               ==>
  SHelf        ==>
  (Mixed Case) ==>
               ==>
               ==>
               ==>
  Wsdir          : /u/prak0XX/services/fibonacci/wsbind/provider
  (Mixed Case)   :
                 :
 MESSAGES: 2 SEVERE  4 ERROR
                                                  SYSID=CICS APPLID=CICS
```

Jetzt füllen wir die Felder *Pipeline*, *Group*, *Configfile* und *Shelf* aus. Das Ergebnis sollte wie folgt aussehen:

```
 OVERTYPE TO MODIFY                                          CICS RELEASE = 0640
  CEDA  DEFine PIpeline(          )
   PIpeline     ==> WSP0XX01
   Group        ==> WSFI0XX
   Description  ==>
   STatus       ==> Enabled           Enabled | Disabled
   Configfile   ==> /usr/lpp/cicsts/cicsts31/samples/pipelines/basicsoap11prov
   (Mixed Case) ==> ider.xml
                ==>
                ==>
                ==>
   SHelf        ==> /u/prak0XX/services/fibonacci/shelf
   (Mixed Case) ==>
                ==>
                ==>
                ==>
   Wsdir          : /u/prak0XX/services/fibonacci/wsbind/provider
   (Mixed Case)   :
+                 :
  MESSAGES: 2 SEVERE  4 ERROR
```

Verlief alles fehlerfrei sollten wir die Meldung *I new group WSFI0XX created*.

Mit dem nächsten Kommando definieren wir die TCPIPSERVICE-Ressource.

```
CEDA DEFINE TCPIPSERVICE(EXMPPORT) GROUP(WSFI0XX)
```

Auch hier sind wieder einzelne Werte anzupassen:

```
OVERTYPE TO MODIFY                                          CICS RELEASE = 0640
 CEDA  DEFine TCpipservice( EXMPPORT )
```

```
  TCpipservice ==> EXMPPORT
  GROup        ==> WSFI026
  DEscription  ==>
  Urm          ==> NONE
  POrtnumber   ==> 03601              1-65535
  STatus       ==> Open               Open | Closed
  PROtocol     ==> Http               Iiop | Http | Eci | User
  TRansaction  ==> CWXN
  Backlog      ==> 00005              0-32767
  TSqprefix    ==>
  Ipaddress    ==>
  SOcketclose  ==> No                 No | 0-240000 (HHMMSS)
  Maxdatalen   ==>                    3-524288
 SECURITY
  SSl          ==> No                 Yes | No | Clientauth
  CErtificate  ==>
  (Mixed Case)
 MESSAGES: 1 SEVERE  1 WARNING
```

Falls wiederum alles fehlerfrei verlief wird dies mit der Meldung *Define successful* quittiert.

Als Letztes installieren wir die PIPELINE-Ressource mit dem Kommando

```
CEDA INSTALL PIPELINE(WSP0XX01) GROUP(WSFI0XX)
```

Der Webservice ist nun nach der Meldung *Install successful* fertig konfiguriert und installiert. Um dies zu überprüfen führen wir das folgende Kommando aus:

```
CEMT INQUIRE WEBSERVICE
```

Wenn alles korrekt verlaufen ist sollte das Ergebnis nach mehrmaligem Blättern mit der Taste *F8* wie folgt aussehen

```
Webs(Tutorial7-Fibonacci             ) Pip(WSP0XX01)
   Ins Uri($253080 ) Pro(FIBO1   ) Com                       Dat(20081012)
```

Aufgabe: Erstellen Sie unter CICS den Web Service Tutorial7-Fibonacci*! Nutzen Sie dazu die obige Beschreibung und ersetzen prak0XX durch Ihren eigenen Benutzernamen.*

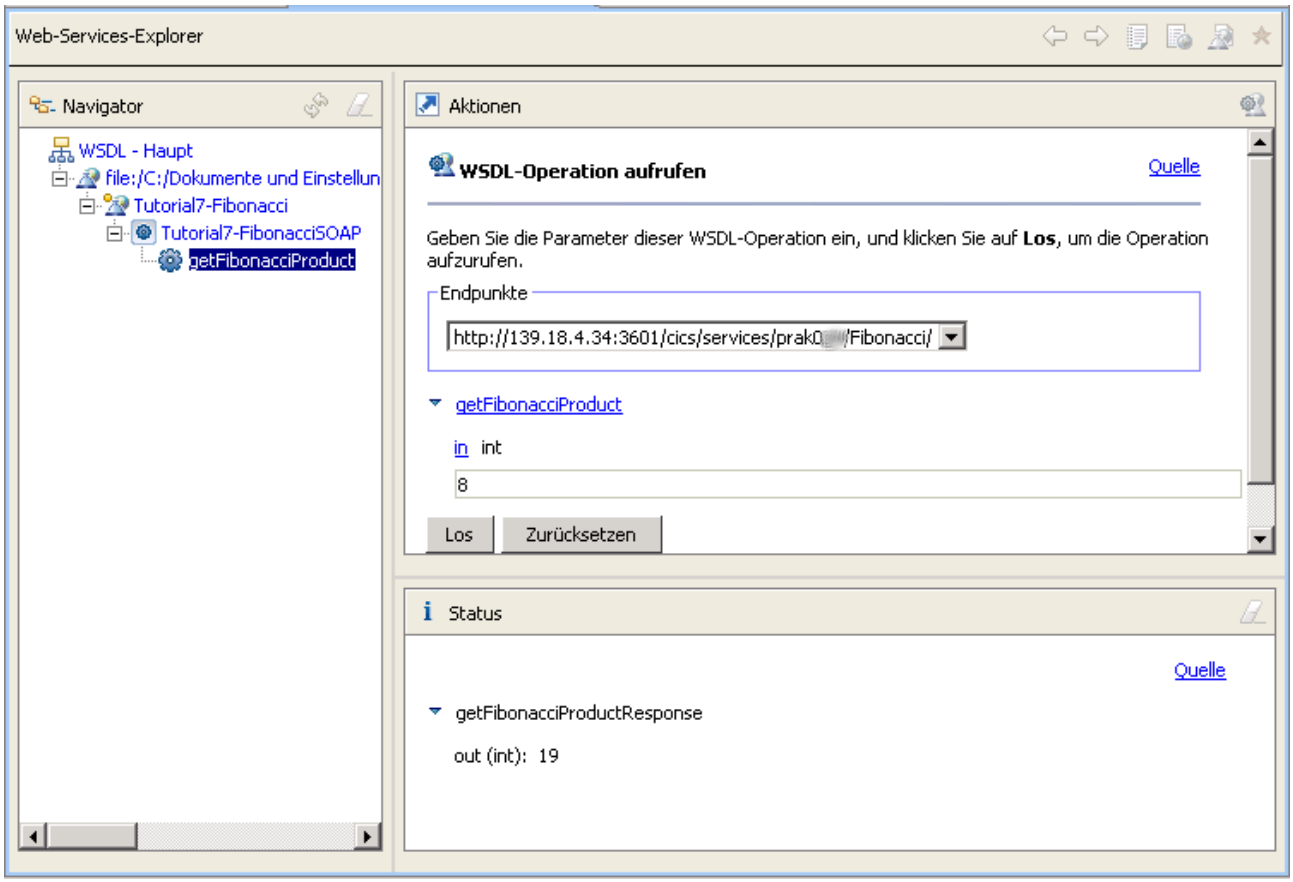

Abbildung 20: Erfolgreicher Test des Fibonacci Web Services