

**SoforTe** – Solutions for Teams



**SOFORTE**  
SOLUTIONS FOR TEAMS

# TAURUS zServer

October 17, 2011

Udo Partsch

[partsch@soforte.de](mailto:partsch@soforte.de)

<http://www.soforte.com>

- Erlaubt die Ausführung von Programmen auf dem Mainframe (Server), initiiert durch einen beliebigen Client (Windows, Linux, Mainframe etc.)
- Benutzt als Protokoll TCP/IP, d.h. der Client benötigt keine zusätzlichen Produkte für die Kommunikation.

- Der Server wird auf dem Mainframe als Long-Running-Task gestartet.
- Verwendet Multitasking und kann daher parallele Anforderungen verarbeiten.
- Ist speziell für den Onlinebetrieb mit hoher Last konzipiert.
- Stellt den Anwenderprogrammen ein vollständig initialisiertes LE-Environment zur Verfügung und ermöglicht dadurch eine extrem schnelle Initialisierung der Anwendung.

- Auszuführende Befehle und Parameter werden in einen vorgeschalteten Header definiert.
- Für Windows- und Linux-Clients steht ein spezieller ASCII-Header zur Verfügung.
- Unterstützt u.a. die folgenden Befehle
  - CALL
    - Aufruf von Programmen mittels BALR
  - EXEC
    - Aufruf interpretativer REXX-Prozeduren
  - LINK
    - Aufruf von Programmen mittels SVC (Anwendung läuft in eigener UOW)
    - Aufruf von CICS-Programmen mittels ECI

- Anwendungen können in den folgenden Sprachen geschrieben sein
  - COBOL
  - PL/1
  - REXX (interpretativ oder kompiliert)
  - C/C++
  - Assembler (LE-konform)
  - JAVA
  - Fortran
  
- LE-Debug-Tool kann zum Debugging der Anwendung verwendet werden.

- Erlaubt Task-Level-Security, d.h. Programme können mit der Autorisierung des Clients ausgeführt werden.
- Daten für die Anwendung können als XML-Dokumente übergeben werden.
  - Server verwendet z/OS XML-System-Services um das Dokument zu parsen.
- Kann TSO/E und REXX-Umgebung dynamisch zur Verfügung stellen oder unter TSO/E gestartet werden.

- Daten können in einer beliebigen Codepage vom Client vorgegeben werden (Server verwendet z/OS Unicode-System-Services zur Codepage-Konvertierung.
  - Client schickt ASCII-Header und Daten als UTF-8
  - Server konvertiert Header und Daten nach EBCDIC
  - Programm stellt Ergebnis in EBCDIC zur Verfügung
  - Server konvertiert Ergebnis nach UTF-8 und schickt Ergebnis zum Client

- Unterstützt mehrere Send / Receive-Varianten für Stream-Sockets
  - Single Send/Receive
  - Send/Receive mit vorgegebener Länge
  - Receive mit vorgegebenem Delimiter
  - Send/Receive via API im Anwenderprogramm
    - Server übergibt Descriptor an Anwendung
    - Geeignet für satzweisen Datenaustausch

- Programme können resident geladen und wieder verwendet werden (Refresh via Konsole).
- Kann Programme nach einer definierten Anzahl von Abbrüchen inaktiv setzen.
- Kann Subtask nach Abbruch neu starten.
- Kann für größtmögliche Transparenz der internen Abläufe sehr detaillierte Trace-Informationen erzeugen.

- Besitzt zur Steuerung eine Command-Task für das Operating, z.B.
  - Stopp
  - Refresh von Programmen
  - Änderung des Trace-Levels
  - Kill Subtask bei Loop im Anwenderprogramm
  - Individuelle Commands über Command-Exit möglich

- Besitzt zur Überwachung eine Service-Task für vordefinierte oder individuelle Aktionen.
  - Überwachung von Loops
  - Individuelle Aktionen über Service-Exit

- Alle Funktionen des Servers können mit einer speziellen Ressource-Klasse mit RACF geschützt werden:
  - darf der Server die User-Id wechseln?
  - darf der Server SSI-Calls verwenden?
  - darf der Server Konsol-Befehle absetzen?
  - darf der Server ein bestimmtes Programm mit CALL oder LINK aufrufen?

- Kann der Anwendung mehrere Datenbereiche definierter Größe als Memory zur Verfügung stellen
  - Datenbereiche bleiben bis zum Server-Shutdown erhalten.
  - Die Adresse eines Datenbereiches kann in der Anwendung mit einem API ermittelt werden.
  
- Besitzt einen HTTP-Monitor für die Überwachung.

- **Besitzt einen eigenen Variablenpool**
  - Analog ISPF-Profile-Pool
  - Variablenpool kann mittels API auch von Programmen genutzt werden.

- Spezielle Server für die Verwaltung von ISPF-Clients möglich.
  - Jeder Benutzer erhält einen eigenen TSO/E-ISPF-Adressraum zur Ausführung von Anwendungen
  - Adressraum kann sein
    - Batch-Job
    - ISPF-Client-Gateway

## ➤ Eigenes Host-Command-Environment für REXX.

- Address zServer “GetMsg”
  - Stellt Nachricht vom Client zur Verfügung
- Adress zServer “PutMsg”
  - Stellt Nachricht für Client zur Verfügung
- Address zServer “Dialog”
  - Eröffnet Dialog mit dem Client

## ➤ MVS Catalog Facilities

- LISTCAT
  - Catalog search Interface
  
- LISTDSI
  - Catalog search
  - + Dataset attributes
  
- LISTMEM
  - Catalog search
  - + dataset attributes
  - + member list

## ➤ Recall

- Recall a migrated dataset

## ➤ Verify User (Logon)

- Verify Userid
- Verify Password
- Change Password

## ➤ Getdata

- Read a dataset or member for Browse or Edit
- Optional with Enqueue

## ➤ Putdata

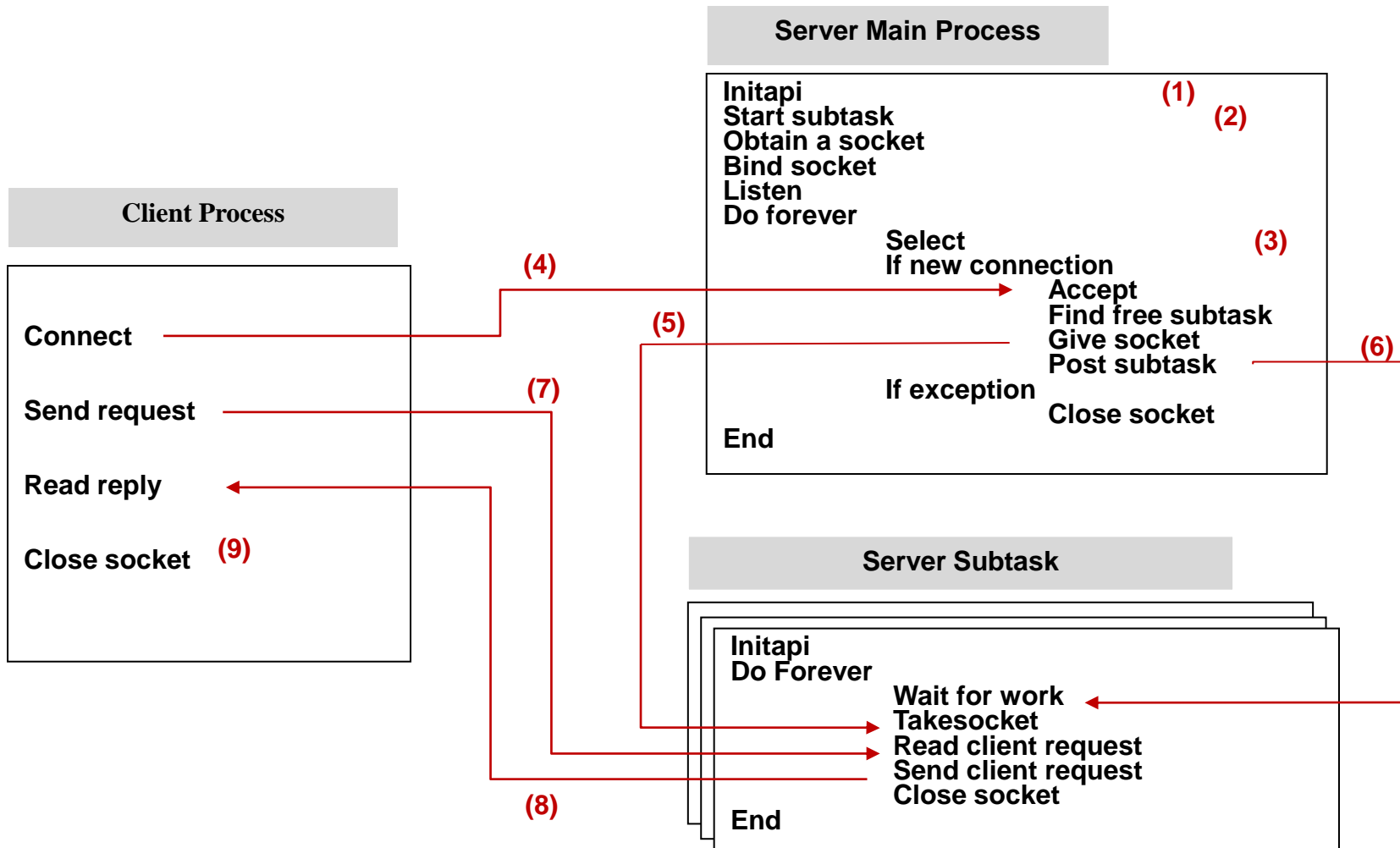
- Update a dataset or member after Edit

## ➤ Dequeue

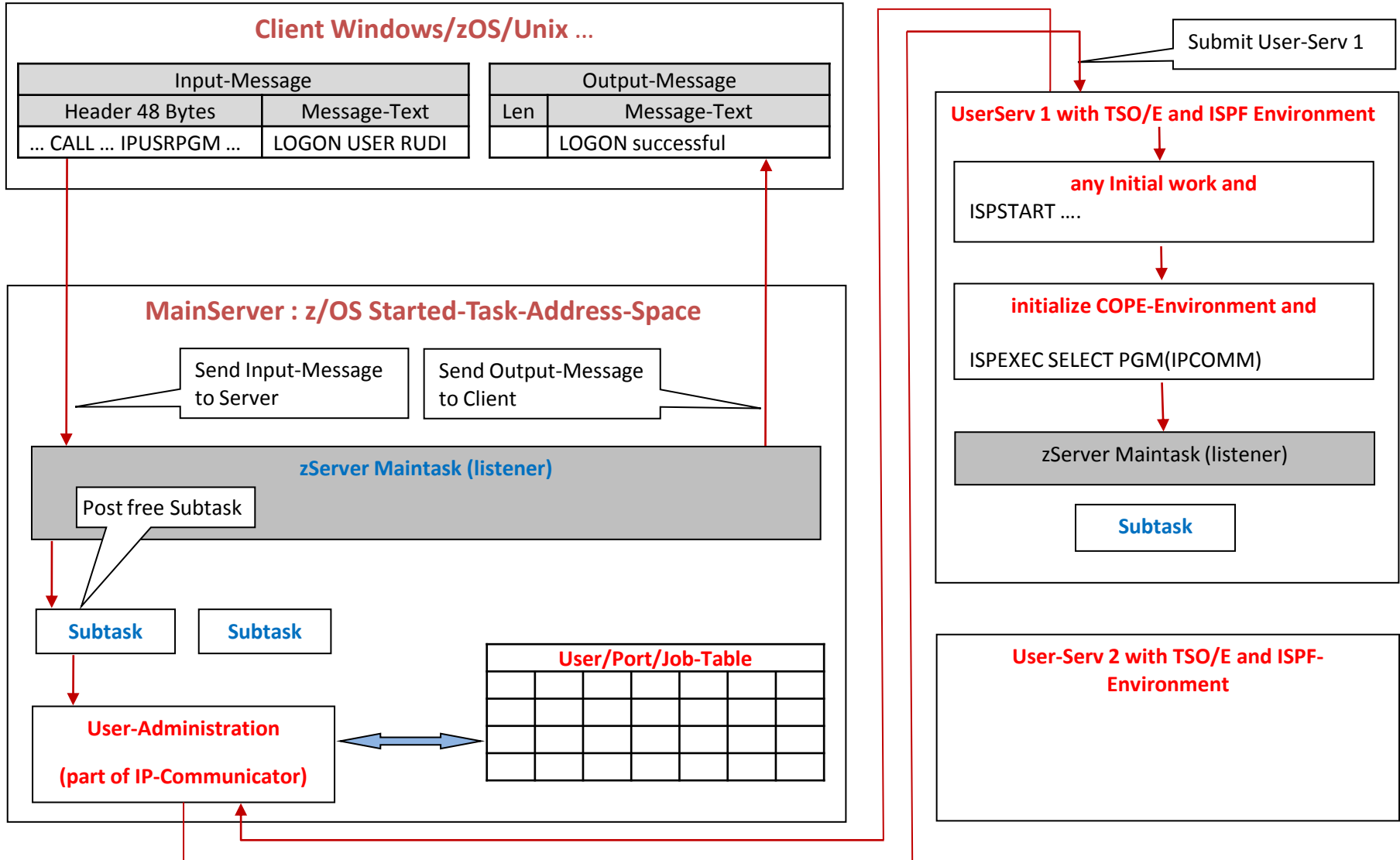
## ➤ JES Job Monitor Facility

- JStatus
  - List Jobs with properties
- JDDlist
  - List Job parts
- JBrowse
  - Browse a Job
- JDDBrowse
  - Browse a job part
- JSubmit
  - Submit a job

# Basic logic in a multitasking concurrent server



# Erzeugen eines ISPF-Adressraumes



# Kommunikation mit ISPF-Dialogen

