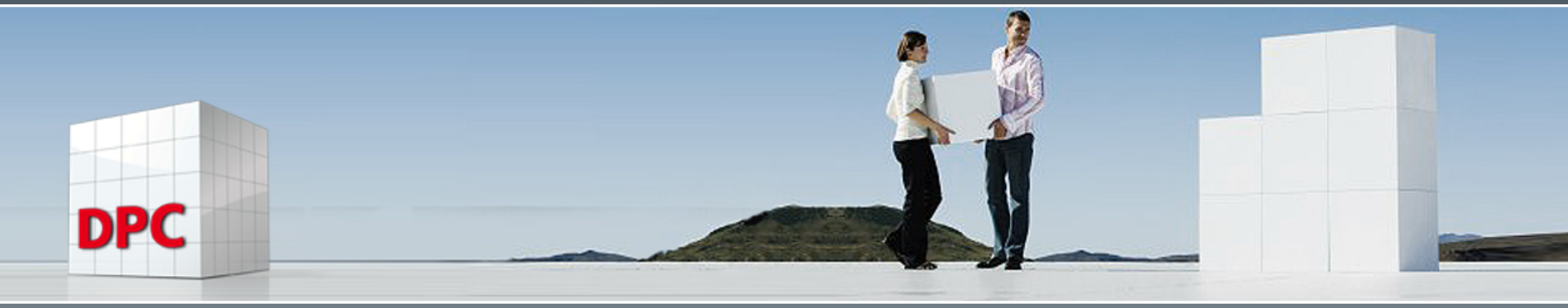


APL-Migrationen

Strategien zur Code-Migration

APL2 / APL+Win → Dyalog APL

Jürgen Wiedemann
DPC

APL-Germany
08. November 2024

APL-Migrationen - Projekthistorie

- APL2 mainframe → APL2/Win
  - Meist nur Session-Anwendungen, teils mit APE oder GDDM
  - meist nur 1:1 Migration zur Hostablösung mit geringem Aufwand
- APL2 mainframe → APL+Win
  - 1:1 Migration in Verbindung mit GUI-Erweiterungen
  - Nutzung von Excel, Komponentendateien, ...
- APL2/Win → APL+Win
  - 1:1 Migration in Verbindung mit GUI-Erweiterungen
  - Komfortablere IDE
- APL2 & APL+Win → Dyalog APL
  - 64-bit, Namespaces, OO, LINK, ...
  - Community, User-Meetings, Webinare, GitHub, ...

APL-Migrationen - Anmerkungen

- Eine Code-Portierung macht Aufwand
- Der Aufwand kann aber gut abgeschätzt werden
- Testen/Abgleichen ist oftmals schwierig
- Oft fehlt es an Dokumentation
- Stufenplan für eine Code-Migration:
  - A) Portierung: Basis-Migration 1:1
    - Test/Abgleich
    - Abnahme
  - B) Adaption: Erweiterungen, Optimierungen und Aufräumarbeiten
  - C) Refactoring: Weiterentwicklung unter Nutzung neuer Features

APL-Migrationen - Typscher Ansatz

- Ausgangspunkt: Transferfiles *.atf
- Code Analyse / Aufwandsschätzung
- Lokalisierung von Sprachunterschieden zur
  - automatischen Ersetzung
  - manuellen Anpassung
  - Speziellen Bearbeitung von Schnittstellen
- Behandlung globaler Variablen
  - Kein Konvertierungsbedarf
  - Speicherung in Komponenten- oder Textdateien (LINK)
- Problemfall Codeänderungen im Alt-System
  - Nutzung von Versionskontrollsystemen (Git / Subversion))

APL-Migrationen nach Dyalog APL

- Migrations-Toolbox
  - Stufe 1: Usercommand]in
    → Erkennung von Code-Änderungen bei Quellsystem-Updates
  - Stufe 2: automatische Anpassungen (Textersetzungen)
    - Korrektur von Fehlern im Zuge von Imports mit]in
    - Zeichenersetzungen falsch übertragener EBCDIC-Zeichen
    - Ersetzung von Highminus, Doublequotes, …
    - Protokollierung potentiell anzupassender Codestellen
  - Stufe 3: manuelle Anpassungen
- Code in Textfiles (LINK)
  - Namenskonflikte infolge Case-Sensitivität beheben
  - Einsatz von Versionskontrollsystemen zur Unterstützung bei Migrations-Schleifen

APL-Migrationen nach Dyalog APL

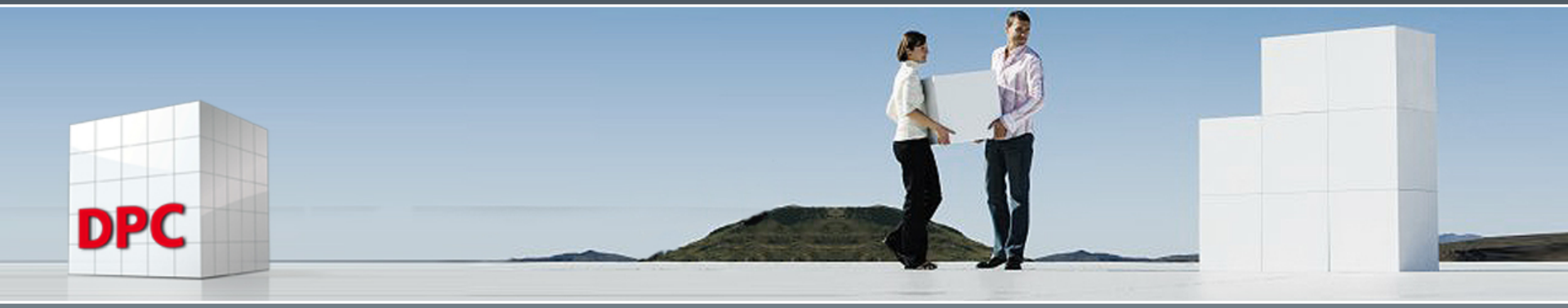
- Besonderheiten
  - Migration Level `⎕ML←3`
  - Defaults für Systemvariablen, z.B. `⎕CT`
  - optionales linkes Argument `RES←{LARG} dyFN RARG`
  - GUI
    - Redesign anstelle von 1:1 Migration (mit `⎕WC`)
    - 1:1 Ersetzung mit `ΔWI`
  - Komponentendateien
    - Umsetzung in dcf-Dateien
    - Zugriff auf sf-Dateien mittels APL+Win Runtime
  - Schnittstellen zu ext. Routinen, Betriebssystem, ...

APL-Migrationen nach Dyalog APL

- Sprachunterschiede
  - Systemfunktionen und –variablen
  - Lokalisierung von Systemvariablen
  - Kontrollstrukturen
  - Error Handling
  - Schnittstellen und Dateisystem
  - ⍠ Format by Example
  - /⍤ Replicate Each
  - Doppeltes Hochkomma (`V←"APL-Germany"`)
  - Ungültige Namen (z.B. highminus `var¯XY`)
  - `A B[2]` Indexing (Bindungsstärke)
  - Inneres Produkt `f.g` bei nicht skalaren Funktionen
  - `^/` auf leeren, geschachtelten Strukturen

Demo

- Toolbox zur APL-Migration:
  - `]load F:\tmp\MigUtils`
- APL2 Migration: EXAMPLES.apl
  - APL2:
    ```
    )load 'F:\tmp\DEMO_APL2.APL'
    )out 'F:\tmp\DEMO_A.ATF'
    ```
  - Dyalog:
    ```
    'APL2PC' mig_import_atf 'F:\tmp\...\DEMO_APL2.atf'
    ```
- APL+Win: Lib 2: DATES.w3
  - APL+Win:
    ```
    )load 'DEMO_APL_PLUS.w3‘
    ]out 'F:\tmp\DEMO_APL_PLUS.ATF‘
    ```
  - Dyalog: `'APLPLUS' mig_import_atf 'DEMO_APL_PLUS.atf'`



APL-Migrationen

Danke / Fragen?

Jürgen Wiedemann
DPC

APL-Germany
08. November 2024