

#### overview

- a house
- motivations
- options
- products

### a house ...



... in a quiet neighbourhood

# ... has a specific architecture



# a house - 20 years of renovation projects piling up



# a house – where to start with renewing?

- leave everything as it is?
- only new wallpaper?
- new windows?
- reface with thermal insulation?
- new central heating?
- new centralized water supply system?
- change floor plans?
- or just move out?
- it just depends, how long you want to stay in this house

### motivations

tear it down & build a new house using standards



#### motivations

or stick to it, due to

- excellent service levels
- inexpensive operating cost
- excellent security in regards to external attacks
- unmatched scalability
- individual business applications

#### motivations

- expected cost reductions
- better maintenance & easier extensions
- generation problems / e.g. COBOL programmers
- user acceptance
- hard-/software discontinuation
- new functions
- simplification
- workflow optimization

### options

- options
  - option 1: "pimp my screen"
  - option 2: "redesign"
  - option 3: "fully fledged modernizing"
- products:











### option 1 – "pimp my screen"

### 1. the most simple method: "pimp my screen"

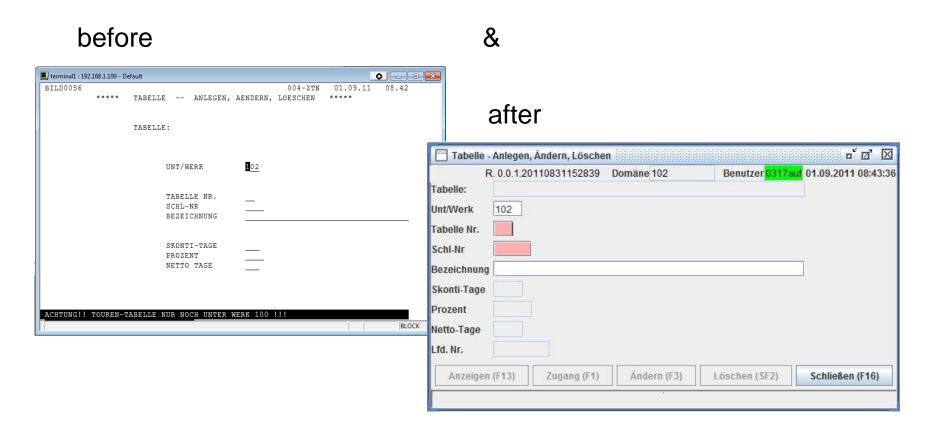
goal: better user acceptance, web functionality

other advantages: none

method: 1:1 client code migration to Java or to HTML



# "pimp my screen"



### option 2 - "redesign"

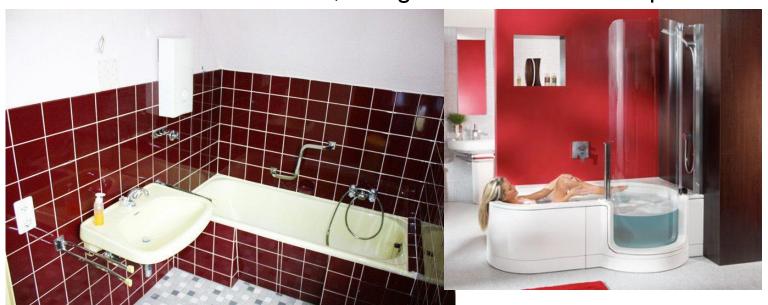
# 2. combining basic dialogues and creating new & more complex dialogues

goal: better user acceptance

other advantages: simplified user navigation

method: implemeting new client apps based on Java

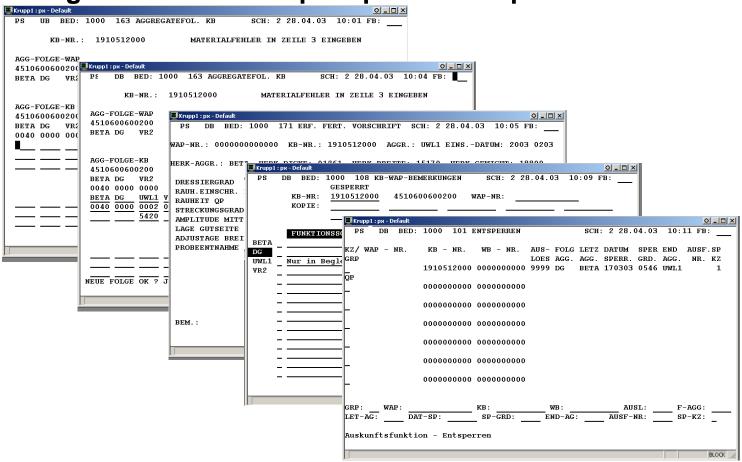
or .Net; using toolkits for faster implementation





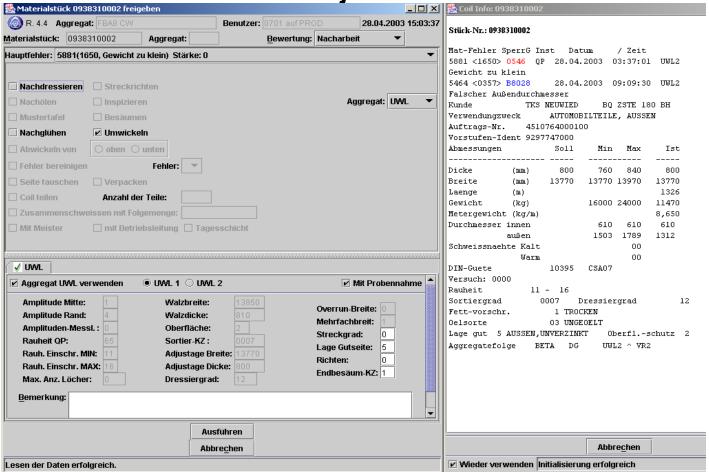
#### before: (example form ITUG 2003)

a logical transaction split up into 5 simple transactions ...



#### after: (example from ITUG 2003)

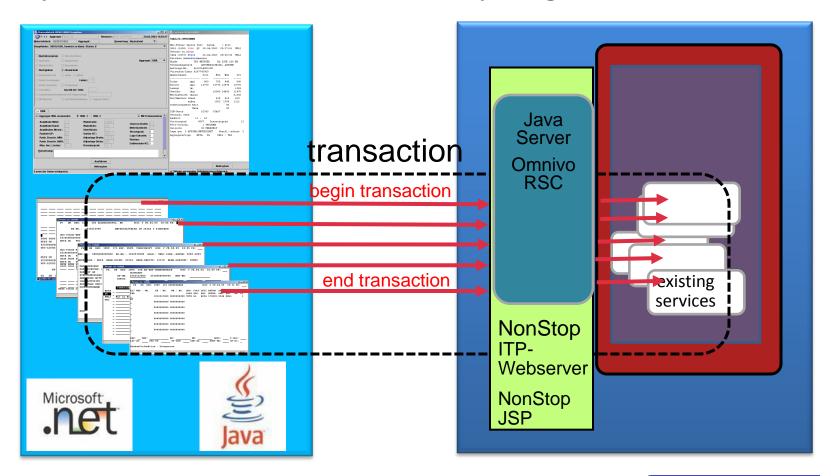
... are now converted into just one transaction







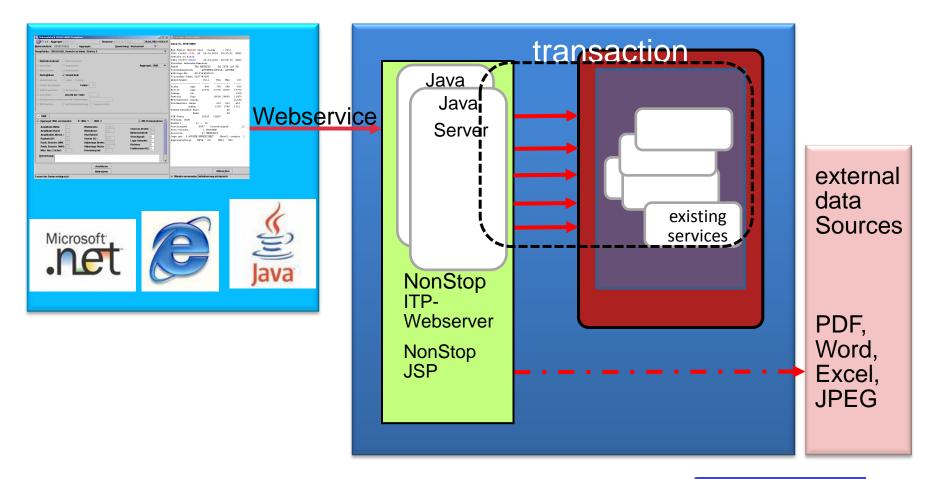
# combining functions within a client, using TomCat on NonStop, only one scaleable Java bean for everything





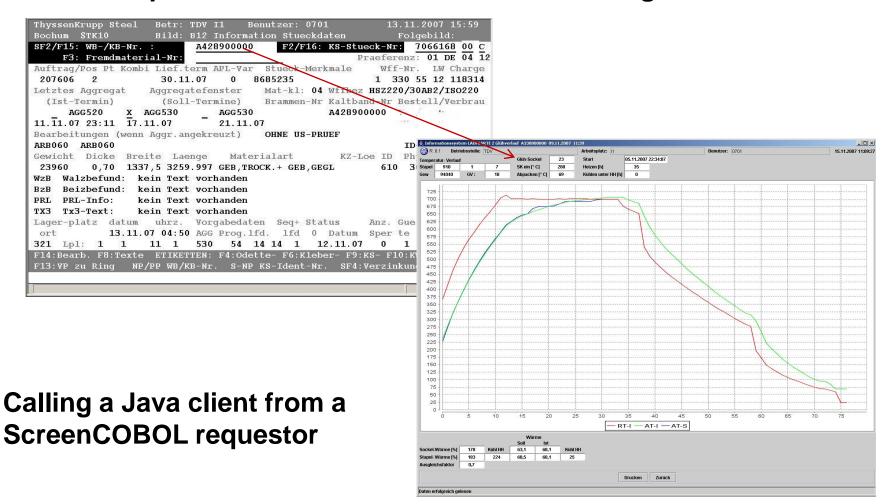


# combining functions within a server, individual Java beans



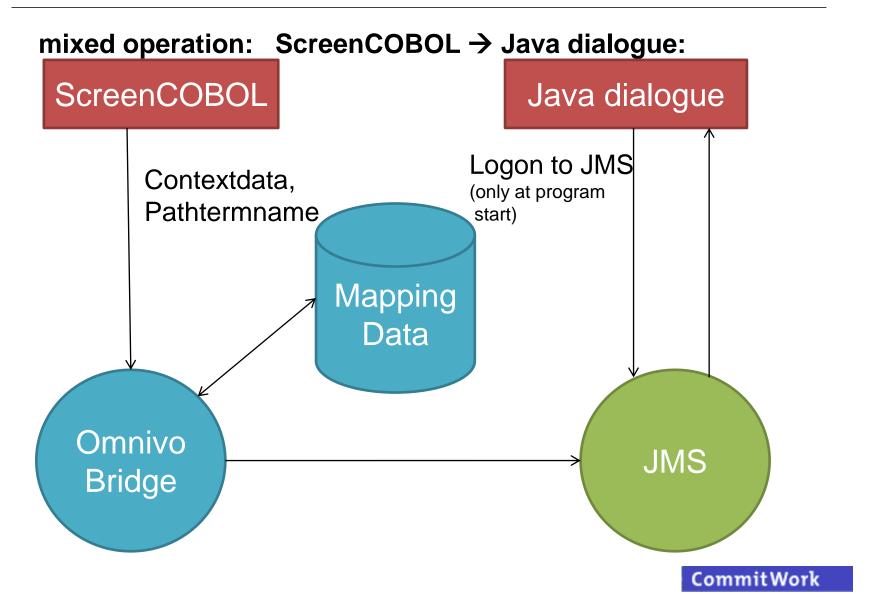


#### mixed operation: ScreenCOBOL → Java dialogue:









### option 3 – "fully fledged modernizing"

#### 3. involving a SOA-stack layer on a NonStop Server

goal: highest flexibility for all requirements on client- and server side



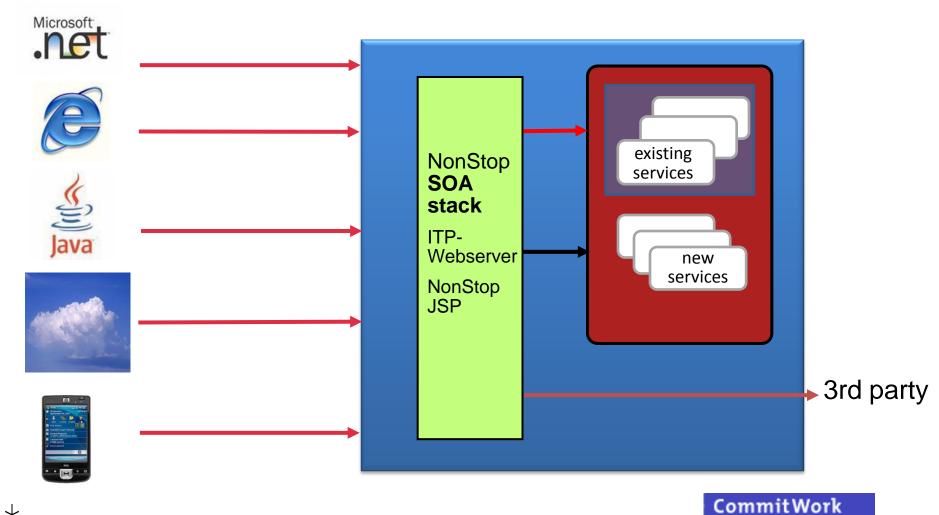




# "fully fledged modernizing"



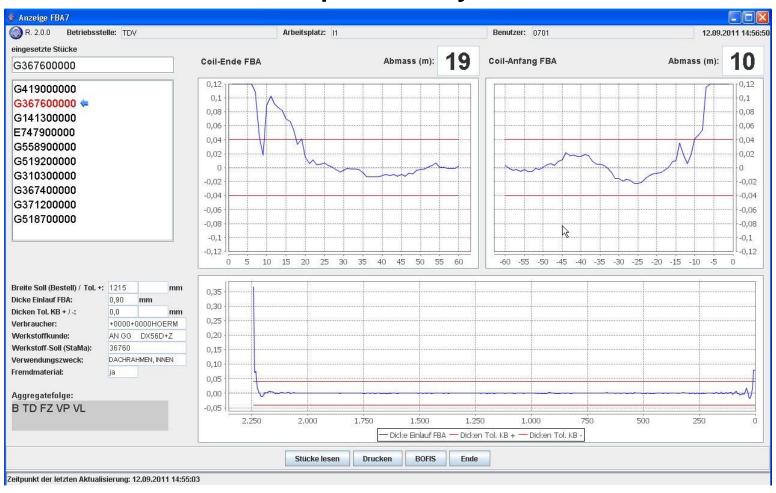
#### open for most everything



# "fully fledged modernizing" ComnivoBridge



#### one function – services provided by several servers

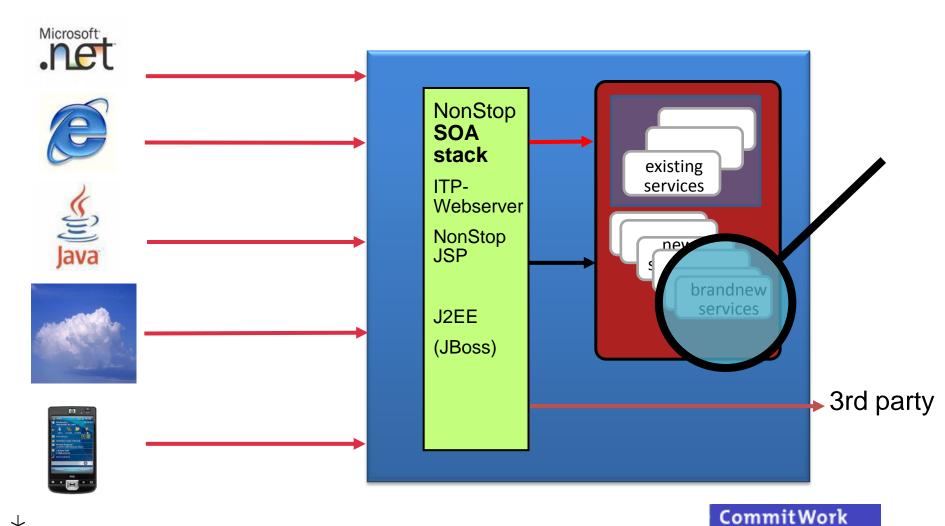




# "fully fledged modernizing"

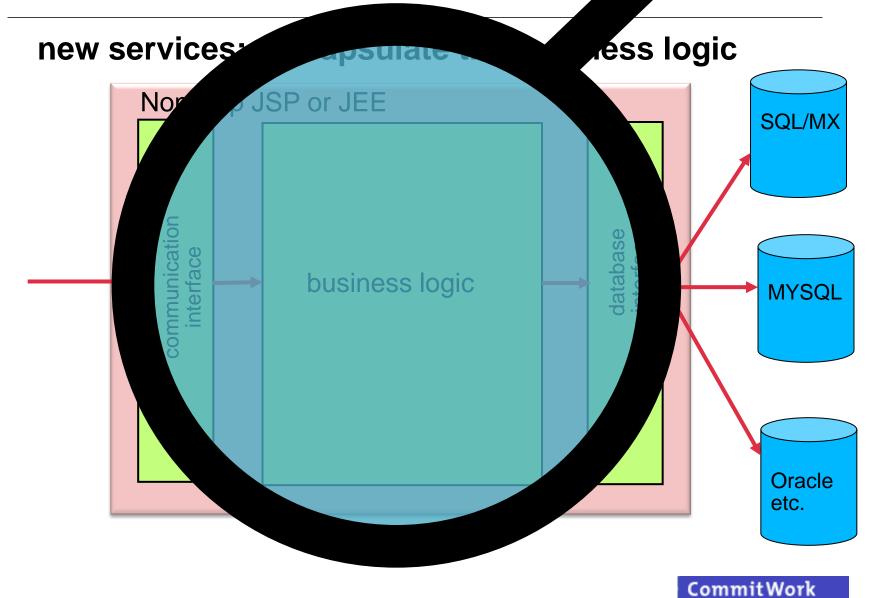


#### smooth migration



# "fully fledged modernizing

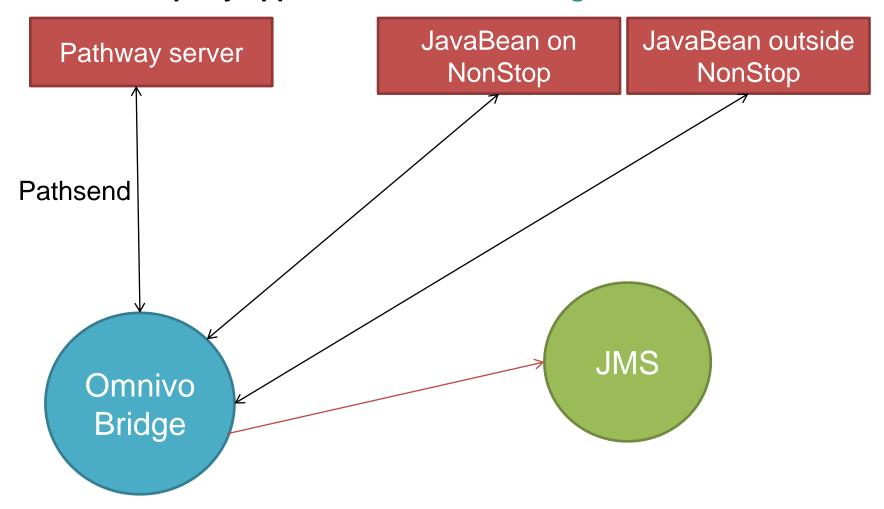




# "fully fledged modernizing"

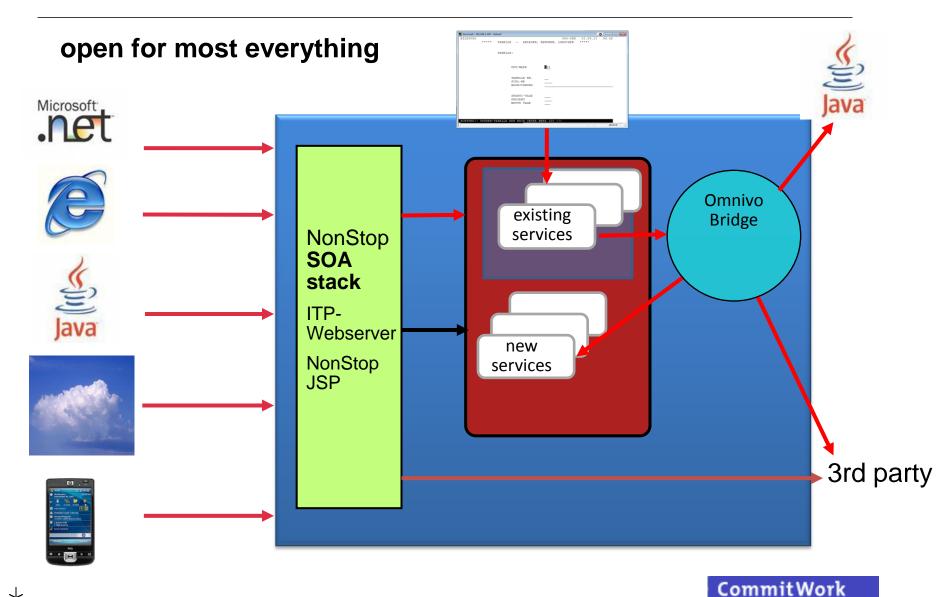


#### access to 3rd party applications: OmnivoBridge enables even more...



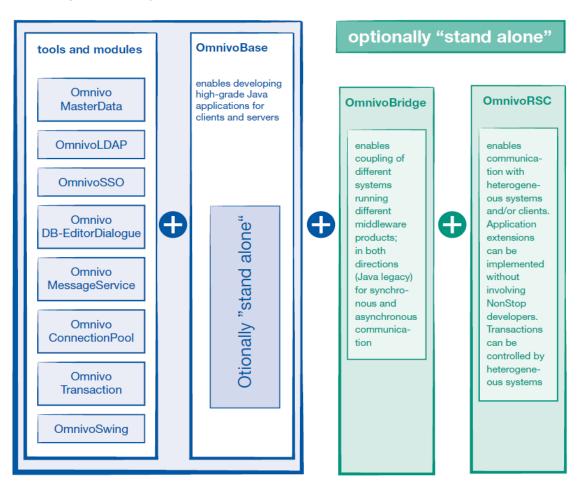
# "fully fledged modernizing"







#### one family - unlimited possibilities



\* Developers supported by CommitWork's Eclipse plug-ins

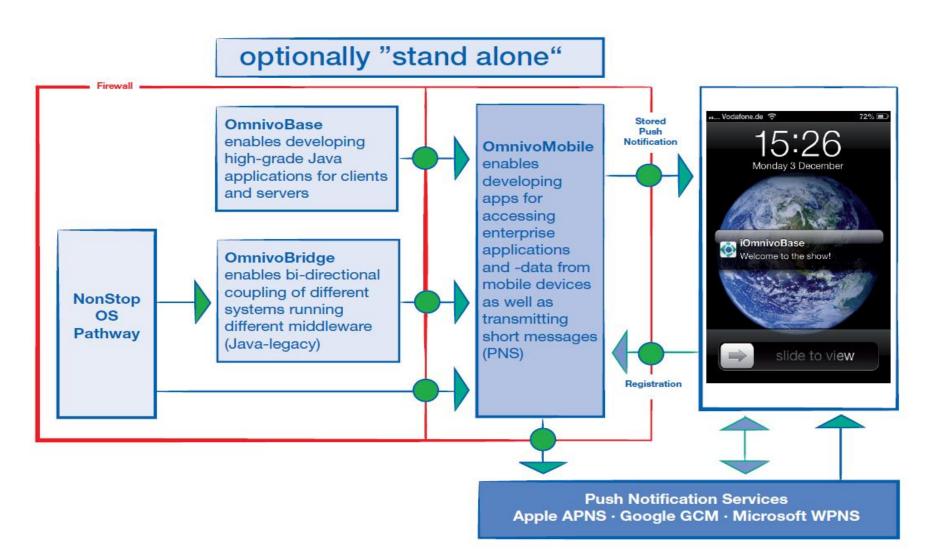




and in the end, mobile access to the NonStop

### **OmnivoMobile**





#### conclusion

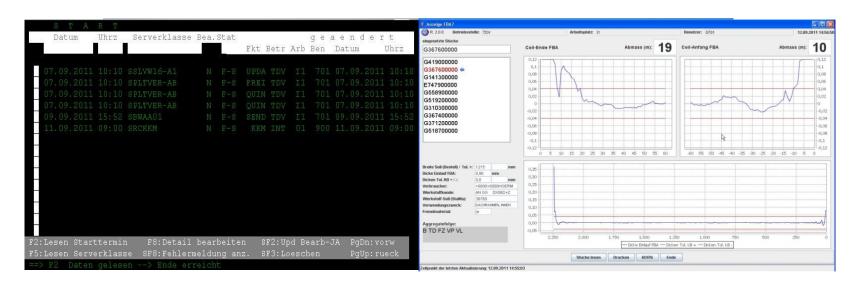
- modernizing is more than just colored screens
- HP NonStop servers should also carry new & modern software
- investment protection by encapsulating the business logic
- smooth migration final goal: eliminating any vendor lock-in
- utilizing new NonStop server resources for new functions
- customers can start with just one CommitWork product and later grow "function-wise" with additional products
- proven in practice, please ask for references

### you want to see more

- Mobile access to the NonStop
- Eclipse plugin for Cobol85, ScreenCobol and Tal
- Please come to a demo

#### conclusion

### modernizing, but done properly!



more information? Visit us at: <a href="http://www.commitwork.com">http://www.commitwork.com</a>

or contact
Hans-Fried Kirschbaum
<a href="mailto:kirschbaum@commitwork.de">kirschbaum@commitwork.de</a>
+49 231 94116910



