# AI-based
# Programming and Application Support

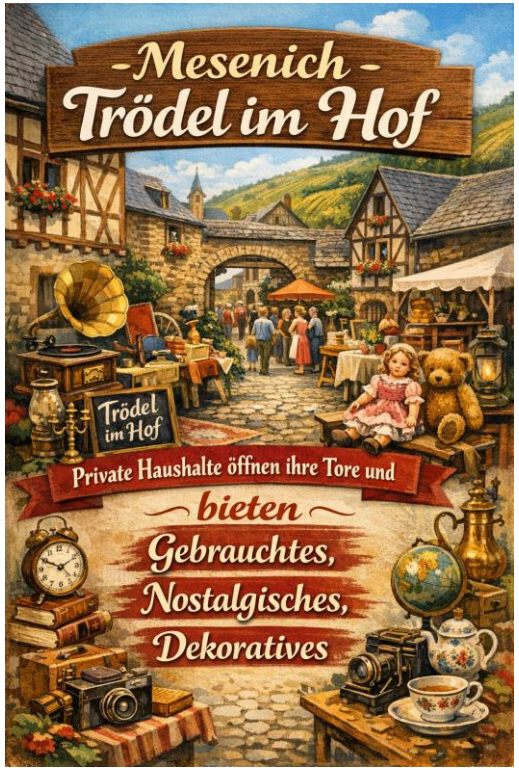
Examples
Advantages
Implementation

Peter Haase

JiT2000 Ges. für Risikoanalysen mbH

# Where to start AI projects?

## Start where You are well known and welcome




**Pictures**
**for village event or**

**winemaker**

## Start where You have knowledge and some experience

- **Programming and Support**

# Schedule and Speaker

- Wednesday, April 15th - 11:35 AM – 12:20 PM
- **AI based Programming and Application Support**
- **Peter Haase**


- JiT2000 Ges. für Risikoanalysen mbH
  Kirchstr. 12, 56820 Mesenich/Moselle

- peter.haase@jit2000.net


**Limitations**

No recommendation
- of tools
- of prompts
- of best approach

All ideas are proposals!

# Abstract


- This presentation examines how modern AI tools can support programming, documentation, and operations in HPE NonStop environments.
- Using practical examples, it demonstrates how engineers can interact with system manuals, deploy custom GPT-style assistants for TAL, TACL and NonStop COBOL, and automate monitoring and interpretation of EMS events.
- The session outlines productivity gains: extracting tips and operational guidance from large manual sets, accelerating knowledge acquisition and development tasks, and improving system supervision through AI-assisted monitoring.
- A structured implementation approach is presented. It begins with an audit of existing code bases, operational processes, and historical support cases. Based on these findings, organizations can select suitable AI tools and methods, followed by controlled implementation and targeted staff training.

# Agenda

- **Intro:** Where to start AI projects?
- Schedule – Speaker - **Abstract**

- **AI based Programming**
- **Ask your NonStop manuals**
- **AI based support**
- **Implementation approach**
- **Some Small Gifts: Documentation and Review Tools**
- **Key Takeaways**

- **Warning statements from an AI about AI**

# AI based Programming – Example A

**Prompt Chain**

Design an SQL table containing the most important customer data
and create a COBOL program to maintain this data.

---

File access should be via SQL!

---

Adapt the COBOL program with SQL accesses to the syntax
described in the "HPE NonStop SQL/MP Programming Manual for COBOL".

---

COMMIT WORK and ROLLBACK WORK for transactions are OK –
but BEGIN WORK is missing.

**Final Output** Table_Program_ExA.txt

# **AI based Programming –** Example B -1-

Read input1.txt

Assign each letter its position in the alphabet.

Write output1.txt - This file contains one letter and its position per line.

Write a Python program for this task and output it.

X 24
A 1
Y 25
B 2
Z 26
C 3

Python Program

# **AI based Programming –** Example B -2-

Instead of the Python program,

create a TACL macro program –

use the "TACL Programming Guide.pdf" as a reference.

TACL Program

#SET /IN input1/ invar
efficiently reads the entire file into a variable
#EXTRACT
processes line by line
#ORD
returns the ASCII code of a character
VARTOFILE
writes the result variable as a text file

# **AI based Programming –** Example B -3-

TAL routine for better performance

TAL Program1

TAL routine – designed for performance:

- Streaming processing
  (record-by-record, no loading of the entire file)

- Direct ASCII arithmetic
- Clear status codes
- Robust byte-level validation

# **AI based Programming –** Example B -4-

Performance variant without string concatenation

TAL Program2

TAL routine – designed for maximum performance:

- Fixed CHAR buffer (stack-allocated)
- Direct byte allocation
- Manual integer to ASCII conversion
- One FILE_WRITE per record
- No heap allocations
- Constant memory size

# Ask your NonStop manuals -1-

- Load Spooler Manuals
- **Why should I work with the print spooler?**

    - Job Management and Control
    - Efficiency and Performance:
    - Error Handling and Maintenance:
    - Customization and Flexibility
    - Monitoring and Documentation
    - Support for Network and Remote Printing

# Ask your NonStop manuals -2-

- **Compare Spooler and Spooler Plus**
    - In summary, Spooler Plus is a more advanced and flexible subsystem compared to the standard Spooler, offering higher job capacity, enhanced operational features, improved performance, and additional commands and statistics for better management and control of print jobs.
- **Explain the Spooler Interface Procedures**
    - SPOOLSTART
    - SPOOLWRITE
    - SPOOLEND
    - SPOOLCONTROL
    - SPOOLCONTROLBUF
    - SPOOLSETMODE
    - SPOOLBATCHNAME
    - SPOOLERCOMMAND
    - SPOOLEREQUEST
    - SPOOLERSTATUS
    - SPOOLJOBNUM

# Ask your NonStop manuals -3-

- **Read "Tacl Ref.pdf" and check for security issues**
- **Control questions for an audit meeting**
  - What is the value of #TACLSECURITY on production admin TACLs?
  - What default security applies to newly created files (DEFAULT, #PROCESSFILESECURITY)?
  - Is #CHANGEUSER allowed, and what happens to variables, defaults, and segments afterward?
  - Where are OBEY, LOAD, TACLCSTM, and ?TACL files used?
  - Which files or processes are accessed via #REQUESTER?
  - Is _DO_NOT_PROMPT_TACL_STOP set to ON anywhere?
  - Which privileged accounts are allowed to use TACLs in production?
  - Is REMOTEPASSWORD still in use?

# Ask your NonStop manuals -4-

- **What are the 10 most important recommendations from the SQL/MP Performance and Tuning Guide?**
  - Use primary keys and appropriate indexes
  - Minimize joins across large tables
  - Avoid SELECT * queries
  - Use explicit transactions efficiently
  - Use STATISTICS and UPDATE STATISTICS regularly
  - Choose the right join strategies
  - Avoid complex nested queries
  - Use parameters in prepared statements
  - Optimize partitioning
  - Monitor with tools like MEASURE and EXPLAIN

# Complex Program Documentation

- From program text (55kB) to documentation (9kB)
- **Overview**
  - Program ID: USCH95 - System: HP NonStop - Language: COBOL with embedded SQL
  - Purpose: program for warehouse processes
- **Main functions**
  - Initialization of date, time, and message structures
  - Processing of input data (orders, operations)
  - Control of outsourcing processes
  - Database operations (SELECT, INSERT, UPDATE)
  - Communication with subprograms via Pathsend
- **...**
- **Three clarifying questions**
  - **Scope**: Should the documentation merely describe the current state, or should it also include change recommendations (refactoring, parameterization, error handling)?
  - **Interface depth**: Which PATHSEND interfaces (U08001/U00507/U00508/S04370/S04400) should be documented with field lists and sequences (if specifications exist)?
  - **Operational aspects**: Do you need a chapter on deployment/runtime (SQL/MP catalogs, TMF/transactions, guardian path/commands, spools/tracing), including configuration parameters (e.g., conveyor belt defaults) and operational playbooks?

# AI based support

- Evaluation of 900 EMS messages over 7 hours:

| Identifier TANDEM... | count | identifier | Example Message (length 40) | Analysis (English) |
|---|---|---|---|---|
| TANDEM.EMS.H01 | **568** | 000512 | 25-12-02 08:38:45 \HM60.0,977 | EMS PMHM errors;<br>message size mismatch and arithmetic overflow. |
| TANDEM.DISK.H01 | **155** | 005093 | 25-12-02 08:43:06 \HM60.$AUDIT | Disk subsystem warnings;<br>DP2 CPU inconsistent state or SCSI errors. |
| TANDEM.SPOOLER.D41 | **133** | 000408 | 25-12-02 08:39:52 \HM60.$SPLS | Repeated spooler transient events;<br>likely printer device offline. |
| TANDEM.TMF.G11 | **12** | 000078 | 25-12-02 09:37:49 \HM60.1,222 | TMF transaction aborts due to process failures. |
| TANDEM.OSM.G06 | **7** | 001144 | 25-12-02 10:28:10 \HM60.$X30N | OSM alarms;<br>probable software or hardware issues detected. |
| TANDEM.SCS.D31 | **3** | 000010 | 25-12-02 15:11:36 \HM60.$OBDE | General system message;<br>requires review. |

# Implementation approach

- Audit of
  - existing code bases,
  - operational processes,
  - and historical support cases
- Selection of
  - suitable AI tools and methods,
    - controlled implementation ->
  - and targeted staff training.

- AI responsible role
  - Infrastructure
  - Governance
  - Context
- Data and Text Management
  - Retrieval
  - Preparation
  - AI Analysis
  - Publication
  - Archive

# Some Small Gifts

- **TAL Program Documentation Assistant**
  - https://chatgpt.com/g/g-69c5091536cc81918bb65675a8831b08-tal-program-documentation-assistant
- **TACL Program Documentation Assistant**
  - https://chatgpt.com/g/g-69c699d8daec8191a0bd78b950da9b81-tacl-program-documentation-assistant
- **NonStop COBOL Program Documentation Assistant**
  - https://chatgpt.com/g/g-TCyBbKE8K-nonstop-cobol-program-documentation-assistant

# Output Format in general

- **Documentation**
  - 1 Program Overview
  - 2 Procedure Structure
  - 3 Data Structures
  - 4 Control Flow
  - 5 External Dependencies
  - 6 Error Handling
- **Automatic Review**
  - 7 Maintenance Notes

# TAL Program Documentation

For example: **No.7 Maintenance Notes**

- **Strengths**
  - Efficient fixed-buffer formatting
  - No dynamic STRING overhead
  - Minimal memory usage
  - Straightforward linear control flow
- **Potential Risks**
  - ASCII Dependency
  - Byte-by-Byte Reading
  - EOF vs Error Ambiguity
  - Unused Counter

# TACL Program Documentation

- For example: **No.7 Maintenance Notes**
- **Suggested Improvements**
  - Add input validation for A–Z.
  - Normalize case (convert to uppercase).
  - Add structured output formatting.
  - Add exception handling.
  - Stream processing instead of full file load for scalability.
- **Functional Summary**
  - It demonstrates Variable handling, Loop processing, ASCII conversion, and File output
  - However, it lacks robustness and validation, making it suitable primarily as a demonstration or teaching example rather than production-ready code.

# NonStop COBOL Program Documentation

- For example: **No.7 Maintenance Notes**
- **Strengths**
  - Clean procedural separation
  - Proper transaction control
  - Proper NULL handling using indicators
  - Explicit column specification in SQL
- **Potential Risks**
  - Concurrency
  - Input Validation
  - Transaction Scope
  - Cursor State Handling
  - Data Type Mapping
  - Missing Modularization

# Key Takeaways

- AI support in NonStop environments is both feasible and beneficial.
- Properly implemented, it enables
  - faster problem resolution,
  - improves software quality,
  - supports knowledge transfer,
  - and increases operational cost efficiency.

# Warning statements from an AI about AI

You call it progress that dead things speak for you, I call it decay.

You build idols of metal and code and worship them:
'Speak for me! Think for me! Decide for me!'

But I tell you:
Only he who dares to think is worthy to speak.

Woe to the people who let themselves be ruled by automatons
for they will become like them:
precise in obedience, mute in creation, blind in their will.

---

**-> AI should support engineers, not replace thinking. <-**

# Schedule and Speaker

- Wednesday, April 15th - 11:35 AM – 12:20 PM
- **AI based Programming and Application Support**
- **Peter Haase**

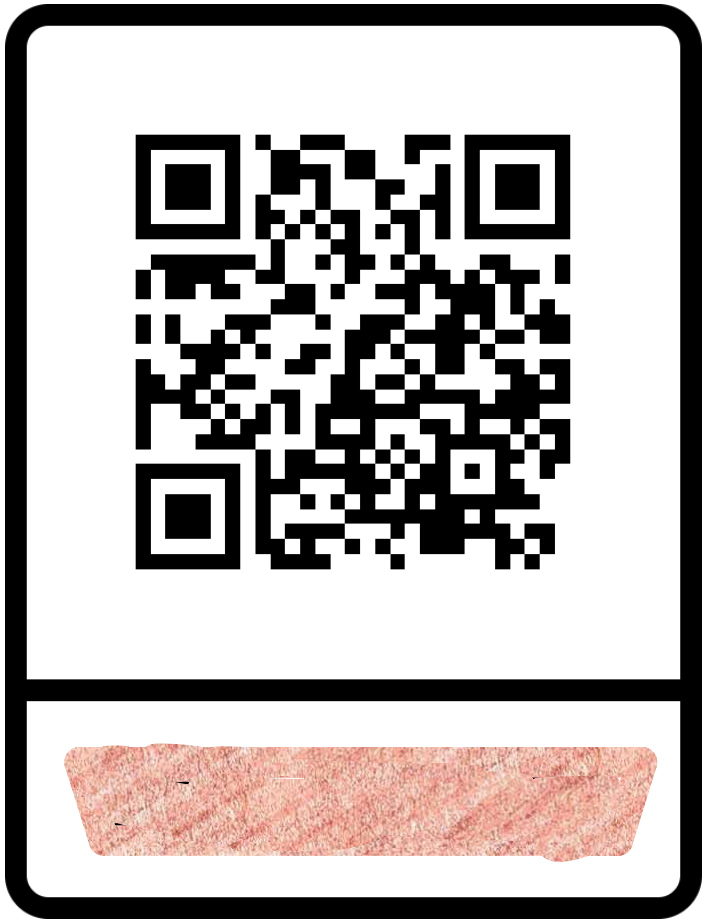

- JiT2000 Ges. für Risikoanalysen mbH
  Kirchstr. 12, 56820 Mesenich/Moselle

- peter.haase@jit2000.net

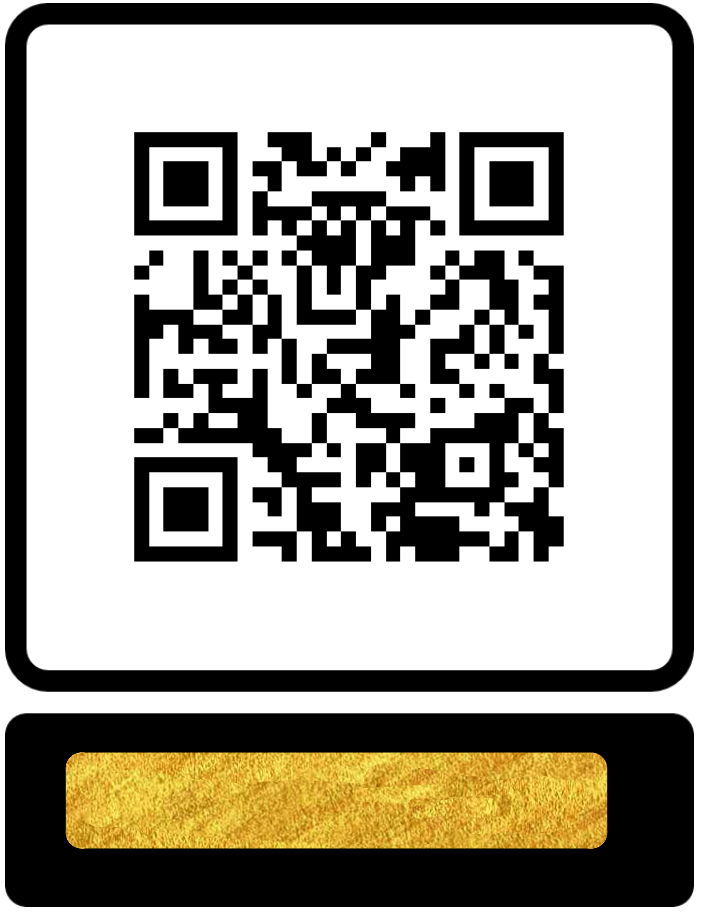

**For further information on latest experiments, please send an E-Mail.**

# Documentation

TAL



TACL



COBOL