General requirements for a program visualization tool to be used in engineering of 4GL-programs

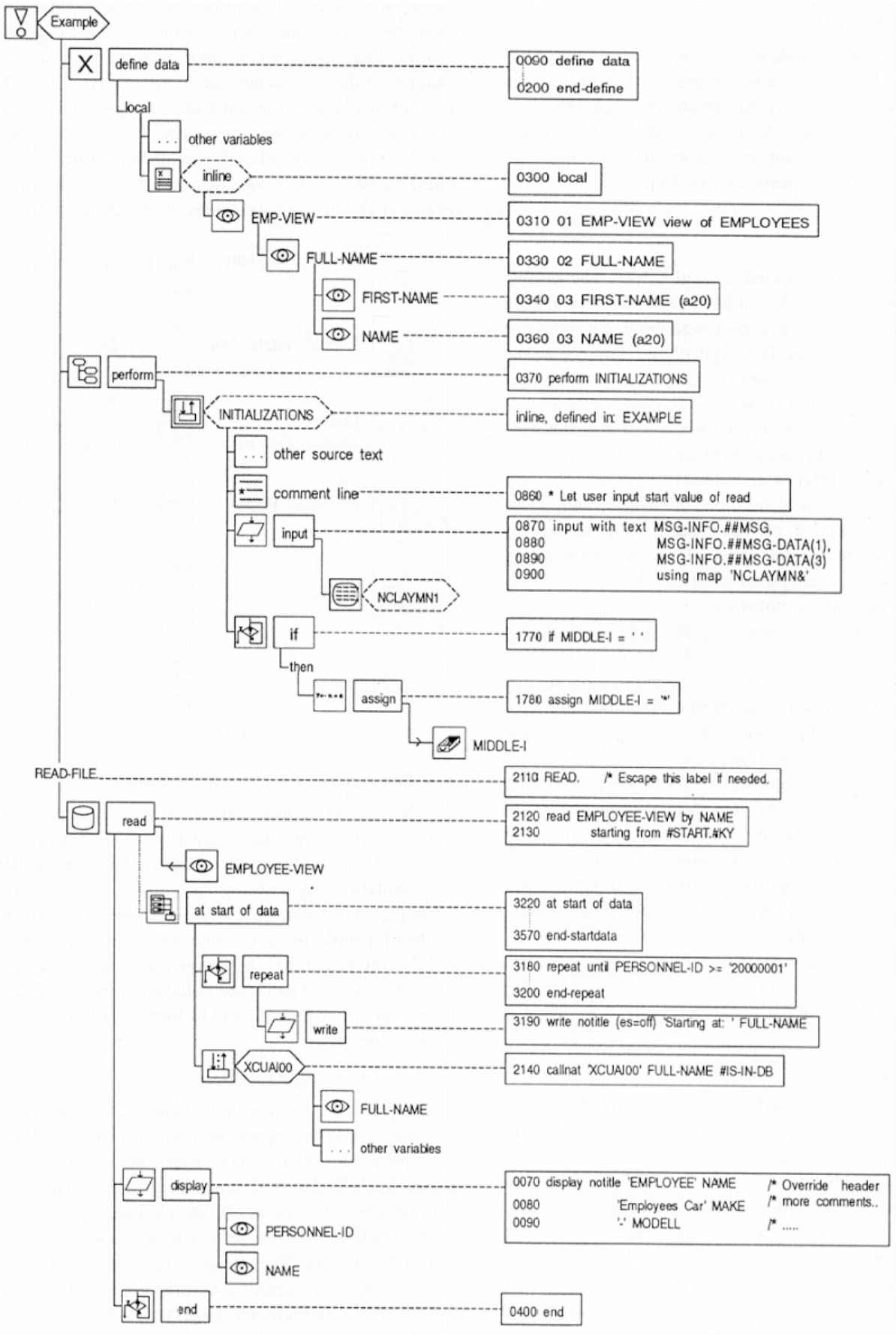
Ludwig Coulmann,.
Chair on Programming Languages & Compilers, Darmstadt University of Technology

This paper appears in: **Visual Languages, 1993., Proceedings 1993 IEEE Symposium on**
Publication Date: 24-27 Aug 1993
On page(s): 37-41
Meeting Date: 08/24/1993 - 08/27/1993
Location: Bergen, Norway
ISBN: 0-8186-3970-9
References Cited: 10
INSPEC Accession Number: 4664477
Digital Object Identifier: 10.1109/VL.1993.269576
Posted online: 2002-08-06 19:04:09.0

Abstract
Program visualization can be used profitably to help a programmer gain an understanding of the program's meaning. In our context this process is called program analysis. The paper first points out that program analysis is highly individual and is influenced by the person involved and by the aim of the process. Secondly, it describes what consequences evolve out of the program analysis characteristics for a supporting tool. Four distinct, general properties of a visualization tool are presented, emphasizing the user and his changing interests. Thirdly, concepts are given for the visualization of 4GL-programs and a specific tool is described as an example of how the outlined requirements translate to a real application. A tree is used to represent different structural relations in the program and icons at the nodes facilitate the tree's perception

---

For a sketch of a displayed NATURAL-program see next page.


Example
define data
0090 define data
0200 end-define
local
other variables
inline
0300 local
EMP-VIEW
0310 01 EMP-VIEW view of EMPLOYEES
FULL-NAME
0330 02 FULL-NAME
FIRST-NAME
0340 03 FIRST-NAME (a20)
NAME
0360 03 NAME (a20)
perform
0370 perform INITIALIZATIONS
INITIALIZATIONS
inline, defined in: EXAMPLE
other source text
comment line
0860 * Let user input start value of read
input
0870 input with text MSG-INFO.##MSG,
0880 MSG-INFO.##MSG-DATA(1),
0890 MSG-INFO.##MSG-DATA(3)
0900 using map 'NCLAYMN&'
NCLAYMN1
if
1770 if MIDDLE-I = ' '
then
assign
1780 assign MIDDLE-I = '*'
MIDDLE-I
READ-FILE
2110 READ. /* Escape this label if needed.
read
2120 read EMPLOYEE-VIEW by NAME
2130 starting from #START.#KY
EMPLOYEE-VIEW
at start of data
3220 at start of data
3570 end-startdata
repeat
3180 repeat until PERSONNEL-ID >= '20000001'
3200 end-repeat
write
3190 write notitle (es=off) 'Starting at: ' FULL-NAME
XCUAI00
2140 callnat 'XCUAI00' FULL-NAME #IS-IN-DB
FULL-NAME
other variables
display
0070 display notitle 'EMPLOYEE' NAME /* Override header
0080 'Employees Car' MAKE /* more comments..
0090 '-' MODELL /* .....
PERSONNEL-ID
NAME
end
0400 end