University of Michigan
Survey Research Center

A New Tool for Visualizing Blaise Logic

Jason Ostergren, Rhonda Ash

15th International Blaise Users Conference
Washington D.C., USA
September 2013

“Visual Blaise” Tool

- A desktop application programmed by the Health and Retirement Study (HRS)
- Officially a prototype and a temporary name*, but is already robust and useable
- “Visual Blaise” displays a graphical flowchart and some additional metadata
- The “Visual Blaise” flowchart can be edited

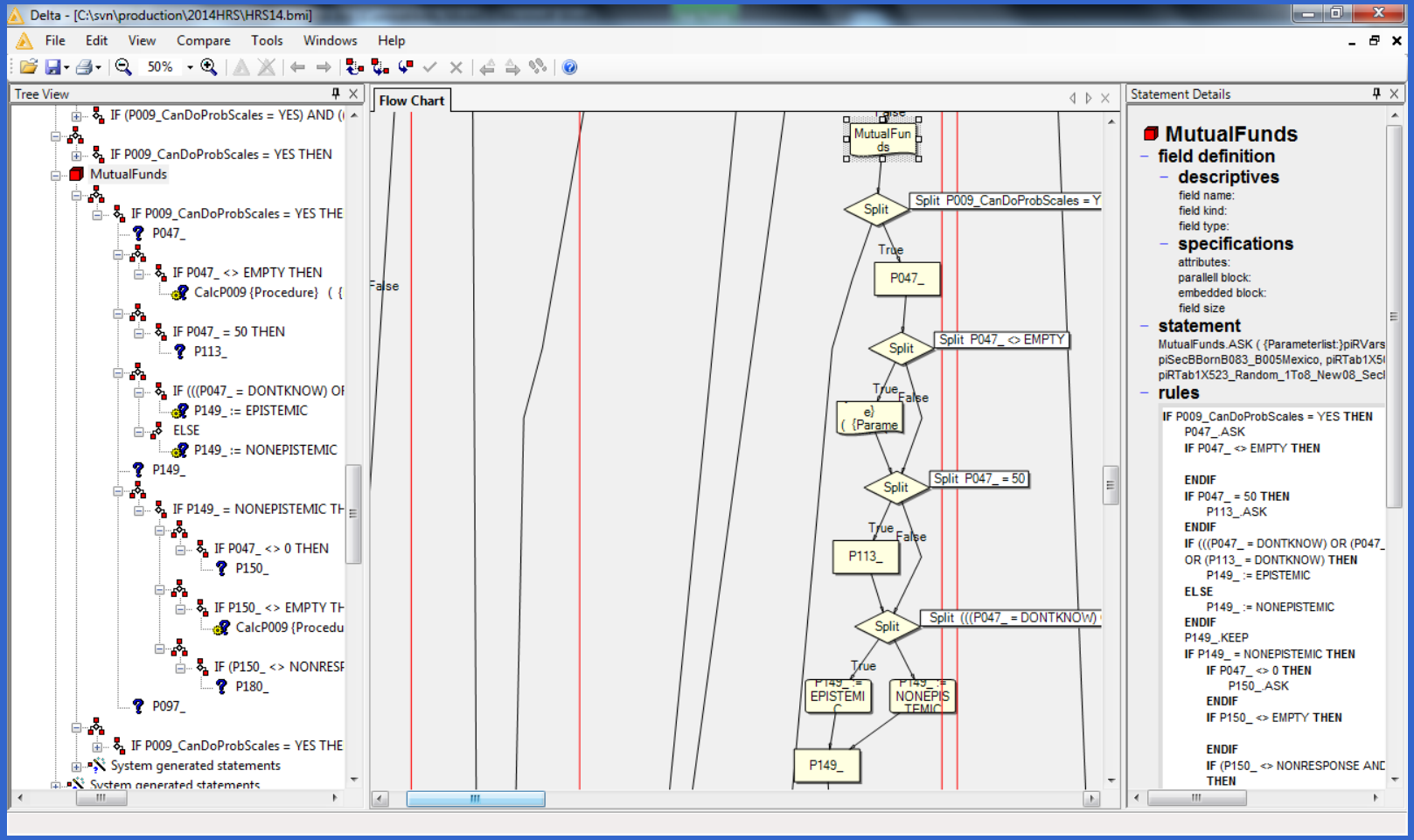
Impetus for Developing

2011 redesign of HRS pension section highlighted deficiencies in our ability to document flow (resulting in much design by demo)

Pension sequence flow is particularly tricky because respondents' understanding of their pensions is often sketchy leading to wrong path

Redesign included many flow problems, such as inserting precarious mechanisms like escape hatches to avoid inappropriate and frustrating questions and redirect to relevant ones

Visualization Tools (e.g. Delta)



Solution Concept

A flowchart with reconfigurable nodes:

- Horizontal flow (fits most monitors better)
- Shallowest logic at top (always asked statements are visible at a glance)
- Colors, patterns, motion aid understanding
- “True arrows” always angled the same way
- “Else arrows” point down (grouping elseifs)
- View only one block at a time
- Robust navigation and find options

Importing Metadata

- Does not use API directly
- Imports from Blaise Rules XML (see IBUC 2010)
- XML also produced by MQDS (since April 2012)
- Abridged import option (intuitive, but permanent)

Internal Object Structure

- Import converts metadata to linked statement objects
- Designed to allow easy insertion and removal
- Link from each object to “Next statement”
- IF and FOR link also to “True statement”
- ELSE and ELSEIF link to “Else statement”
- Collection of linked statements for each Block/Procedure on route

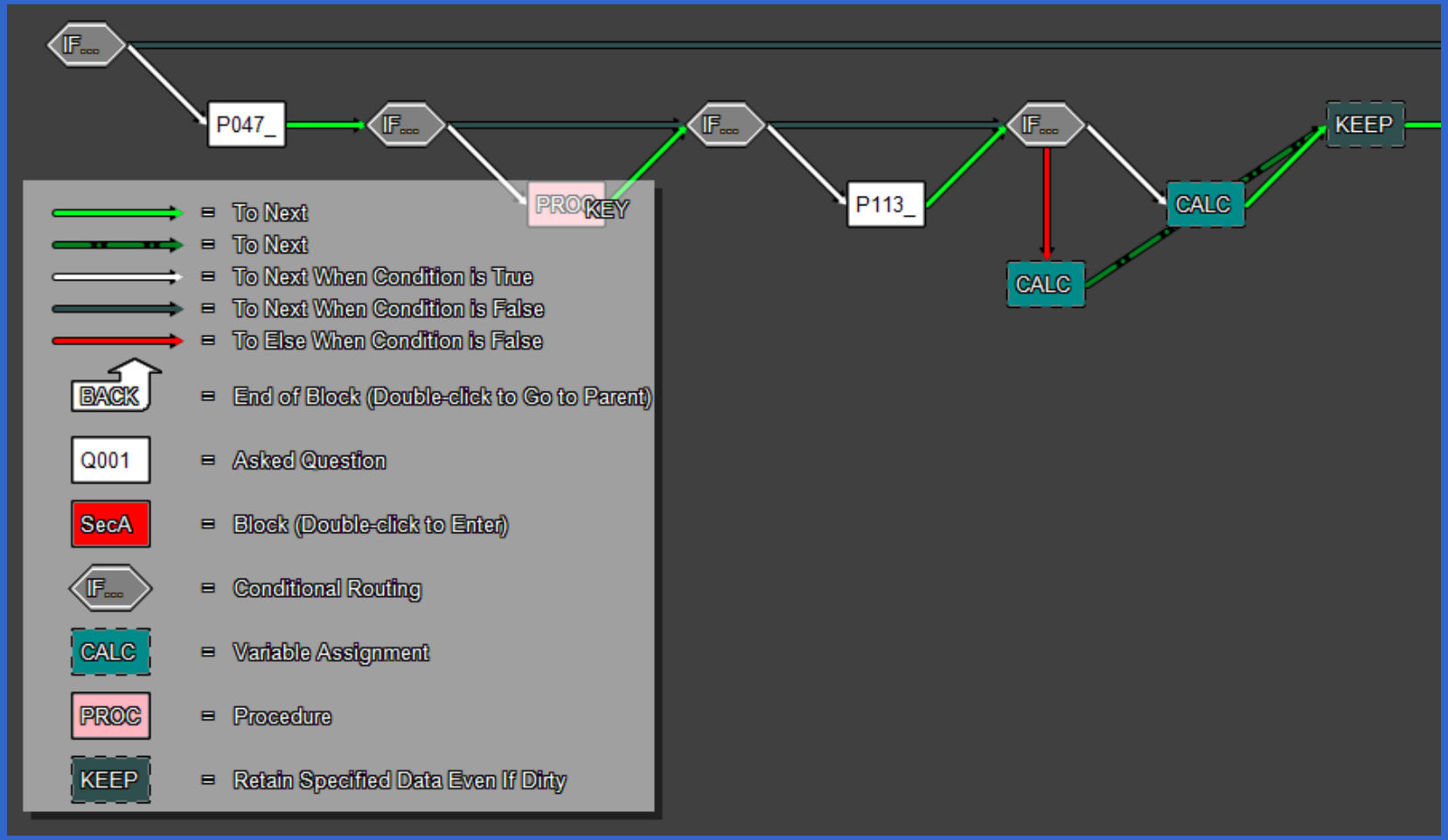
Editing Visually

- Select a single statement:
- Drag and drop a single statement to a new location
- Insert a statement
- Select multiple (contiguous) statements
- Cut and paste statement(s) to a new location
- Cut statement(s) from one block and paste in another
- Delete statement(s)

Output

- Once a datamodel is imported, changes can be saved to and loaded from a native file format for sharing and transportability
- Export changed blocks to a text file showing the rules section(s) in formatted and syntactically correct Blaise code

Demo


IF...
P047_
IF...
PROC KEY
IF...
P113_
IF...
CALC
CALC
KEEP
= To Next
= To Next
= To Next When Condition is True
= To Next When Condition is False
= To Else When Condition is False
BACK
= End of Block (Double-click to Go to Parent)
Q001
= Asked Question
SecA
= Block (Double-click to Enter)
IF...
= Conditional Routing
CALC
= Variable Assignment
PROC
= Procedure
KEEP
= Retain Specified Data Even If Dirty

Conclusion

- Beginning to make use of this tool, but still developing
- Find feature and quick block navigation turning out to be very important (because it combines the advantages of the Blaise database view, which is block based and easy to navigate, with the statements view, which is important for the logic we care about here, but is hard to navigate)
- May have other uses, such as debugging?