Communicating Complex Logic with Ease

Atanas Marchev

Understanding code isn’t just for the wizards

What I will talk about

- It is difficult to translate code to human language
- Non-technical people don’t have to understand code

Beautiful prose logic

(h) Minimum Execution Quantity.

An instruction a User may attach to an order with a Non-Displayed instruction or a Time-in-Force of Immediate-or-Cancel requiring the System to execute the order only to the extent that a minimum quantity can be satisfied. By default, an order with the Minimum Execution Quantity instruction will execute upon entry against a single order or multiple aggregated orders simultaneously. A User may alternatively specify the order not execute against multiple aggregated orders simultaneously and that the minimum quantity condition be satisfied by each individual order resting on the EDGA Book. If there are such orders, but there are also orders that do not satisfy the minimum quantity condition, the order with the Minimum Execution Quantity instruction will execute against orders resting on the EDGA Book in accordance with Rule 11.9, Order Priority, until it reaches an order that does not satisfy the minimum quantity condition, and then the remainder of the order will be posted to the EDGA Book or cancelled in accordance with the terms of the order. If, upon entry, there are no orders that satisfy the minimum quantity condition resting on the EDGA Book, the order will either be posted to the EDGA Book or cancelled in accordance with the terms of the order. Where there is insufficient size to satisfy an incoming order's minimum quantity condition, [and] that incoming order will not trade and will be[, if] posted on the EDGA Book at its limit price[, would cross an order(s) resting on the EDGA Book, the order with the minimum quantity condition will be re-priced to and ranked at the Locking Price].

However, an order with a Minimum Execution Quantity will be cancelled where, if posted, it would cross the displayed price of an order on the EDGA Book. An order to buy (sell) with a Minimum Execution Quantity instruction that is ranked in the EDGA Book will not be eligible to trade: (i) at a price equal to or above (below) any sell (buy) orders that are Displayed and that have a ranked price equal to or below (above) the price of such order with a Minimum Execution Quantity instruction; or (ii) at a price above (below) any sell (buy) order that is Non-Displayed and has a ranked price below (above) the price of such order with a Minimum Execution Quantity instruction. However, an order with a Minimum Execution Quantity instruction that crosses an order on the...

What I will talk about

- It is difficult to translate code to human language
- Non-technical people don’t have to understand code
- Expressing complex logic in prose is unreadable
- We can’t get a quick overview from code
  - What are the paths that can be taken during execution
  - How is the structure of the procedure
  - How much of it is it tested

What is a Nassi-Shneiderman Diagram(NSD)

SIGPLAN Notices 12 1973 August

technical contributions

FLOWCHART TECHNIQUES FOR STRUCTURED PROGRAMMING

I. Nassi
and
B. Shneiderman

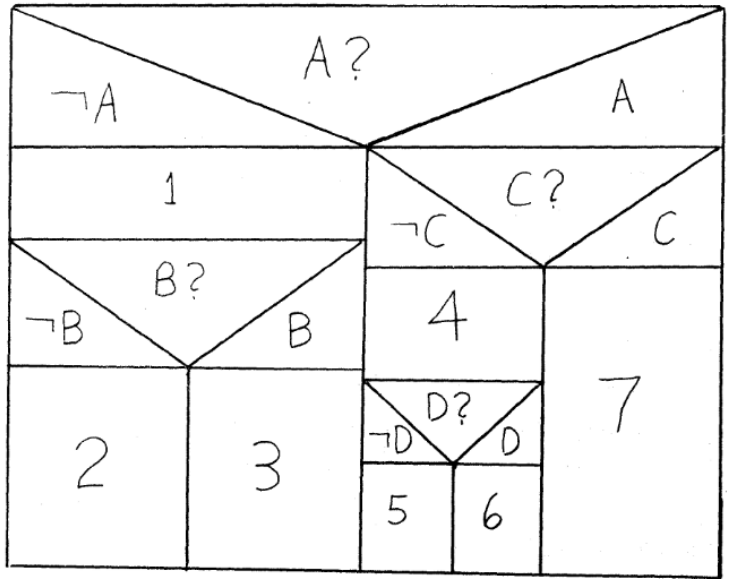
Department of Computer Science
State University of New York at Stony Brook
Stony Brook, L. I., New York

ABSTRACT

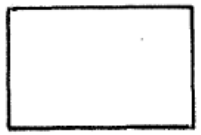
With the advent of structured programming and GOTO-less programming a method is needed to model computation in simply ordered structures, each representing a complete thought possibly defined in terms of other thoughts as yet undefined. A model is needed which prevents unrestricted transfers of control and has a control structure closer to languages amenable to structured programming. We present an attempt at such a model.

Typically, computer programs go through various phases of formulation and definition. During one of these phases a flow-chart may be drawn to describe the program at a level of abstraction

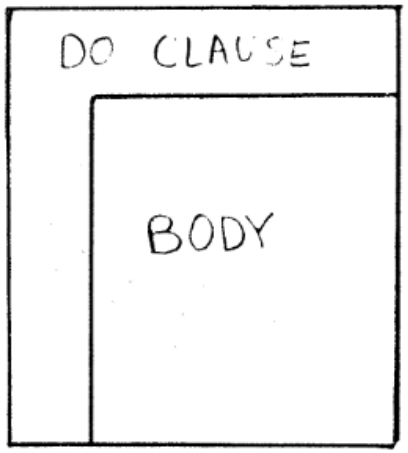
- Flowchart Techniques for Structured Programming
- Written in 1973 by I. Nassi and B. Shneiderman



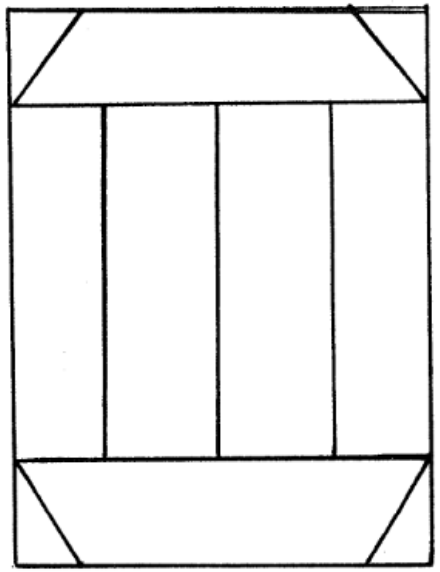
Syntax of NSD



Instruction

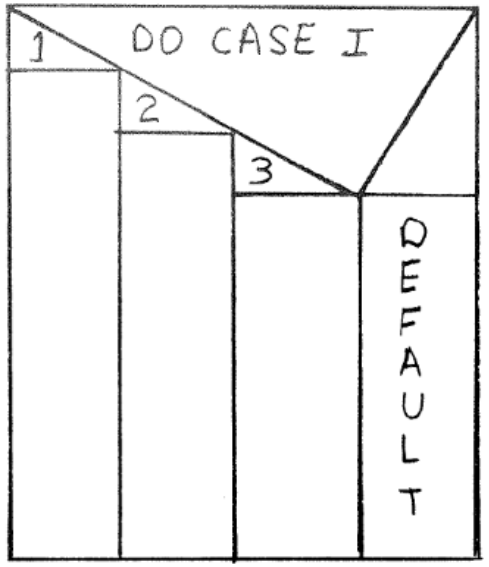


Loop

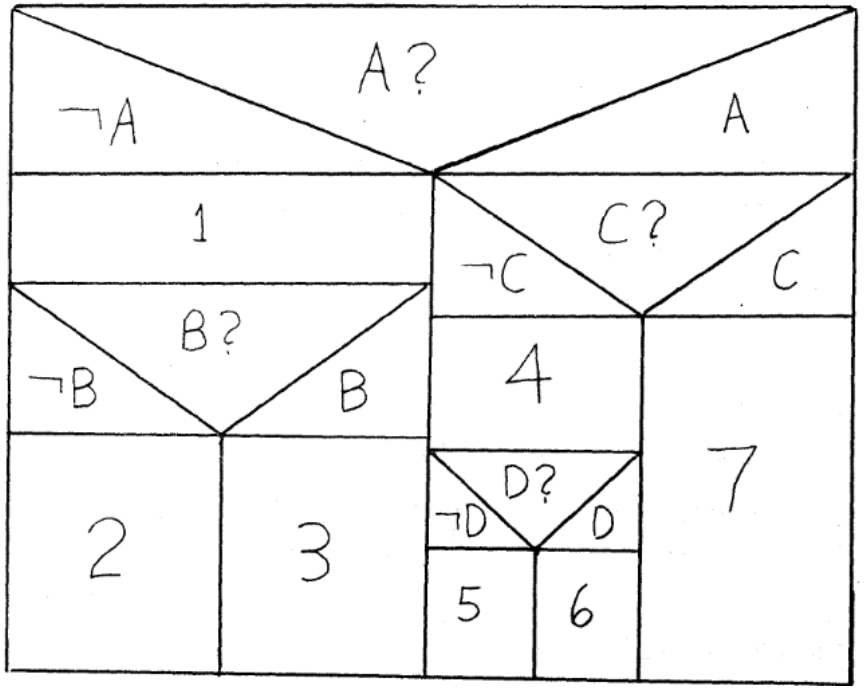


Parallel

Return



Case



Decision

How NSDs solve this problem

- Intuitive design
- If diagram is difficult to read → bad code
- We can follow code paths
  - Complete overview of entire procedure
  - Easier debugging
  - Code coverage

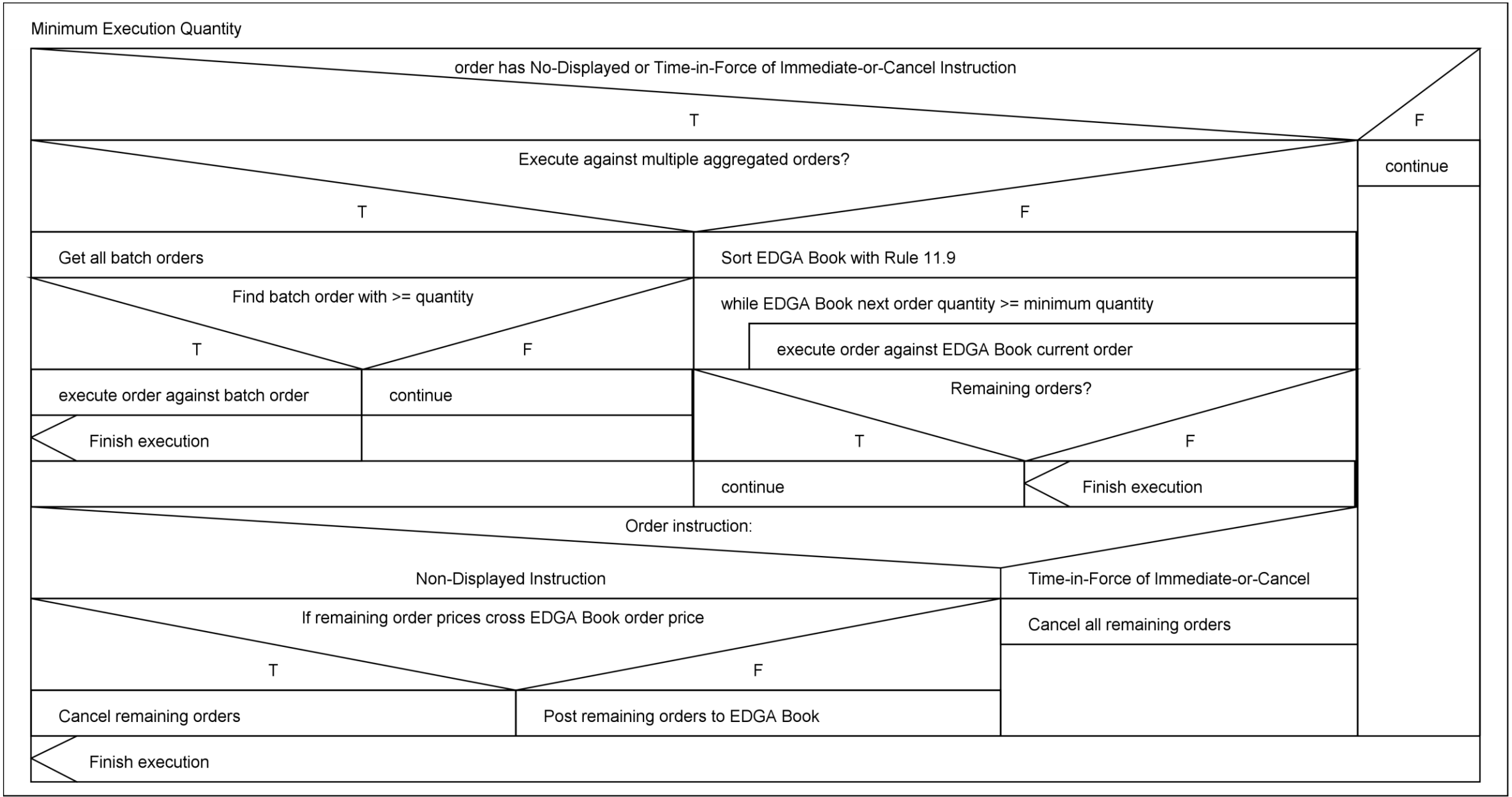
Beautiful prose logic

(h) Minimum Execution Quantity.

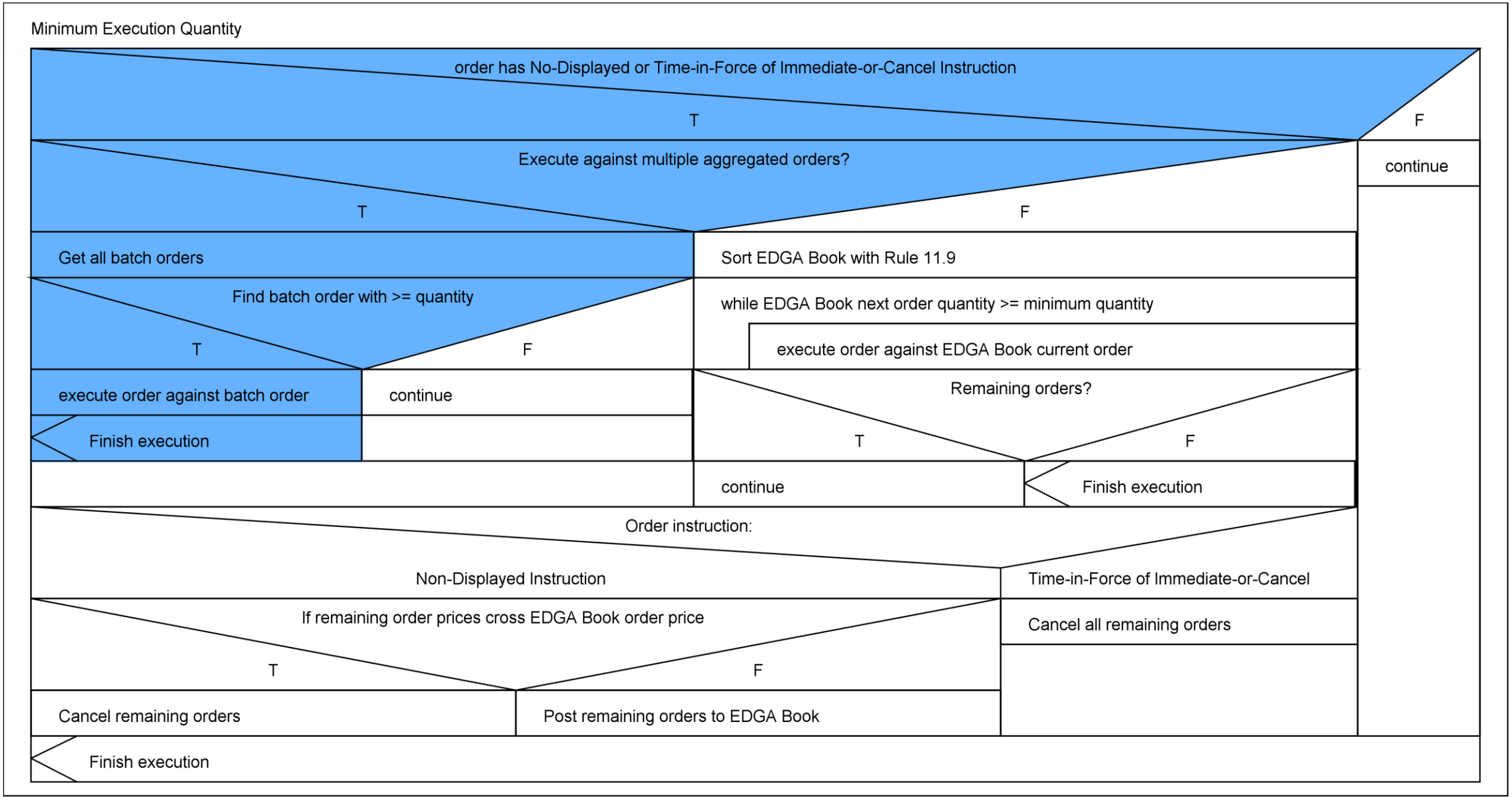
An instruction a User may attach to an order with a Non-Displayed instruction or a Time-in-Force of Immediate-or-Cancel requiring the System to execute the order only to the extent that a minimum quantity can be satisfied. By default, an order with the Minimum Execution Quantity instruction will execute upon entry against a single order or multiple aggregated orders simultaneously. A User may alternatively specify the order not execute against multiple aggregated orders simultaneously and that the minimum quantity condition be satisfied by each individual order resting on the EDGA Book. If there are such orders, but there are also orders that do not satisfy the minimum quantity condition, the order with the Minimum Execution Quantity instruction will execute against orders resting on the EDGA Book in accordance with Rule 11.9, Order Priority, until it reaches an order that does not satisfy the minimum quantity condition, and then the remainder of the order will be posted to the EDGA Book or cancelled in accordance with the terms of the order. If, upon entry, there are no orders that satisfy the minimum quantity condition resting on the EDGA Book, the order will either be posted to the EDGA Book or cancelled in accordance with the terms of the order. Where there is insufficient size to satisfy an incoming order's minimum quantity condition, [and] that incoming order will not trade and will be[, if] posted on the EDGA Book at its limit price[, would cross an order(s) resting on the EDGA Book, the order with the minimum quantity condition will be re-priced to and ranked at the Locking Price].

However, an order with a Minimum Execution Quantity will be cancelled where, if posted, it would cross the displayed price of an order on the EDGA Book. An order to buy (sell) with a Minimum Execution Quantity instruction that is ranked in the EDGA Book will not be eligible to trade: (i) at a price equal to or above (below) any sell (buy) orders that are Displayed and that have a ranked price equal to or below (above) the price of such order with a Minimum Execution Quantity instruction; or (ii) at a price above (below) any sell (buy) order that is Non-Displayed and has a ranked price below (above) the price of such order with a Minimum Execution Quantity instruction. However, an order with a Minimum Execution Quantity instruction that crosses an order on the...

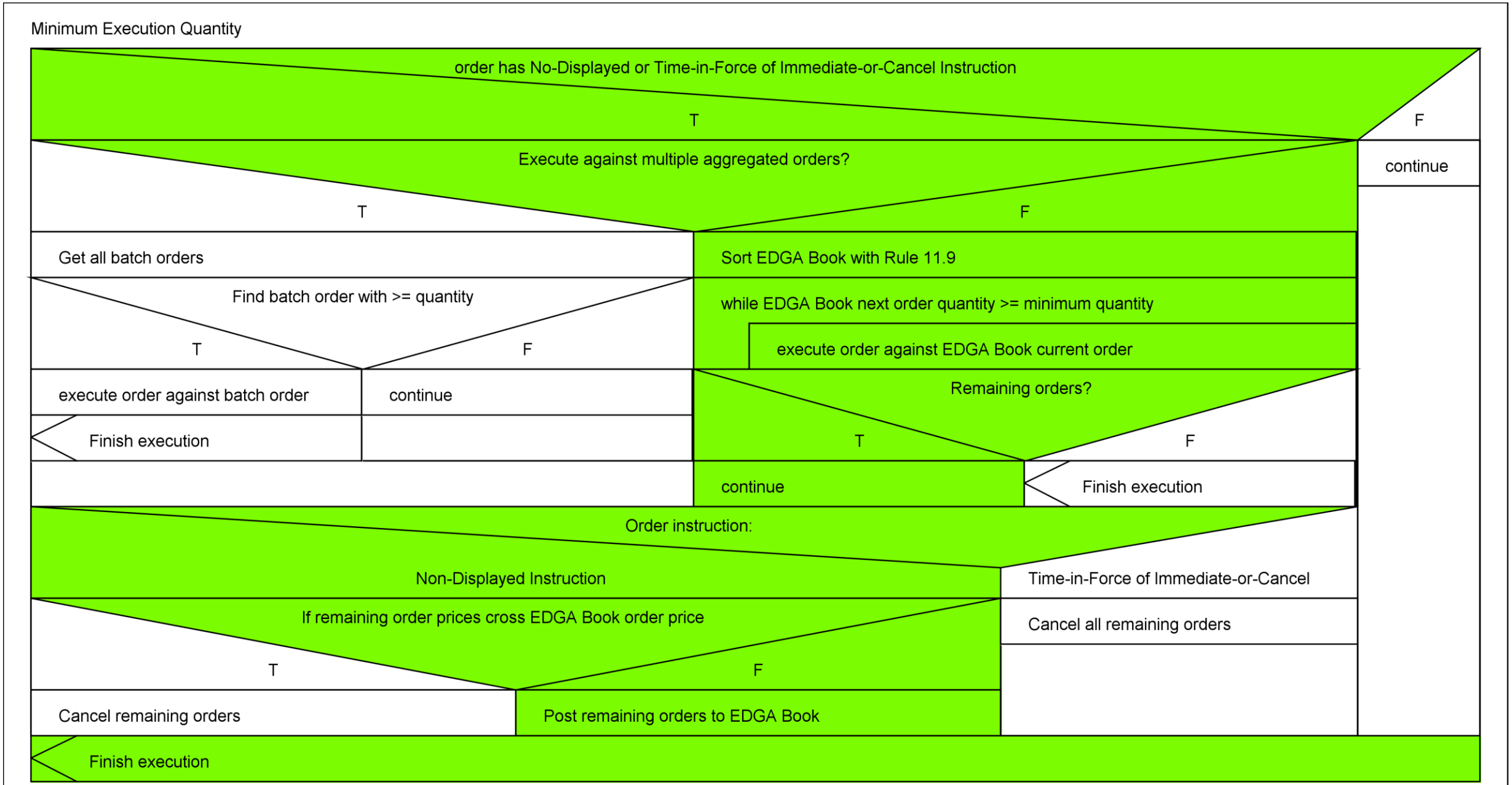
Better diagram logic



Better diagram logic



Better diagram logic



Demo

Challenges with implementation

- Storing the NSD objects
- Swing, graphical or editor level presentation
- Code coverage

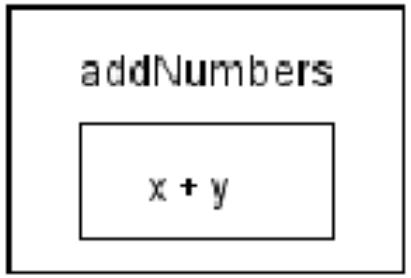
Storing the NSD objects

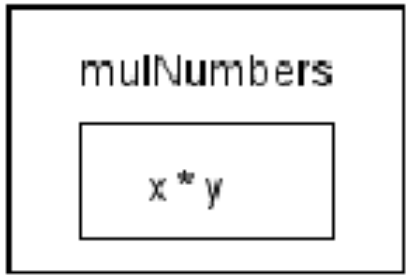
Demo2

test suite Demo2 [success]

show types:
only local declara
inherit scope from

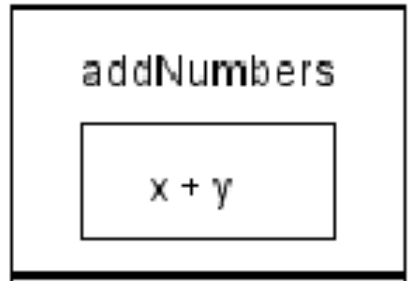
```
fun addNumbers/R(x: number, y: number) {
  x + y

}
fun mulNumbers/R(x: number, y: number) {
  x * y

}
```

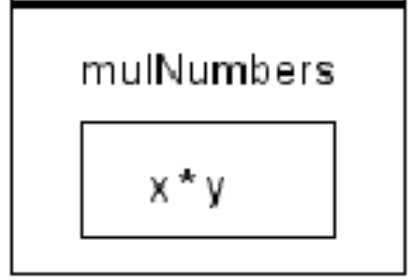




Demo2

test sui

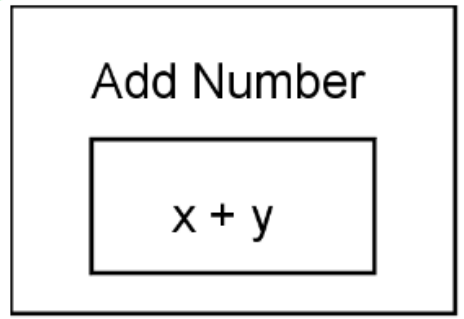




Storing the NSD objects

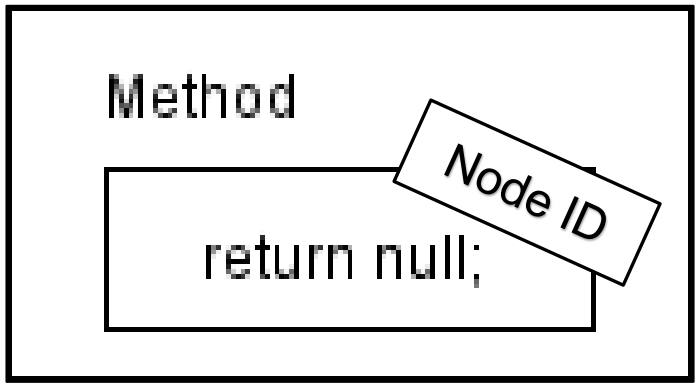
- NSD Objects represent NSD symbols
- NSD Objects to be kept as long as editor is open
  - Editing the node elsewhere wouldn't break the diagram

```
fun addNumbers/R(x : number, y : number) {
  x + y

}
```

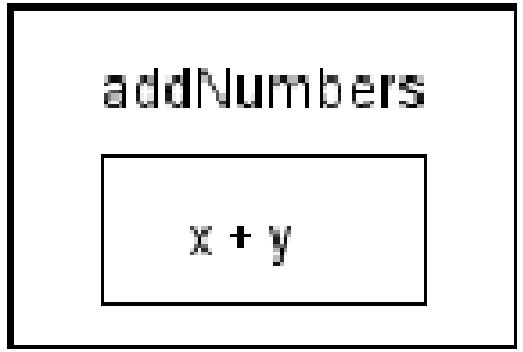


Storing the NSD objects

- NSD Objects represent NSD symbols
- NSD Objects to be kept as long as editor is open
  - Editing the node elsewhere wouldn't break the diagram

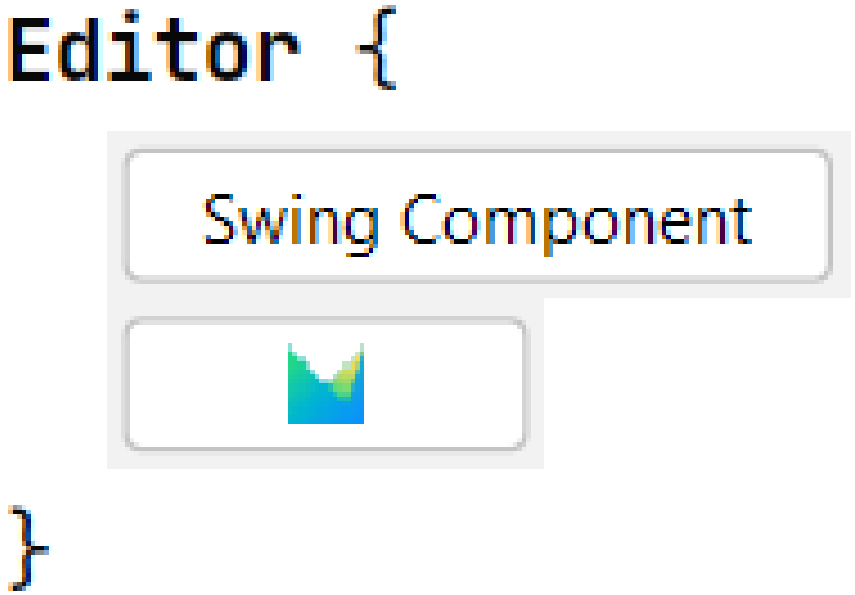



```
fun addNumbers/R(x: number, y: number) {
  x + y

}
```



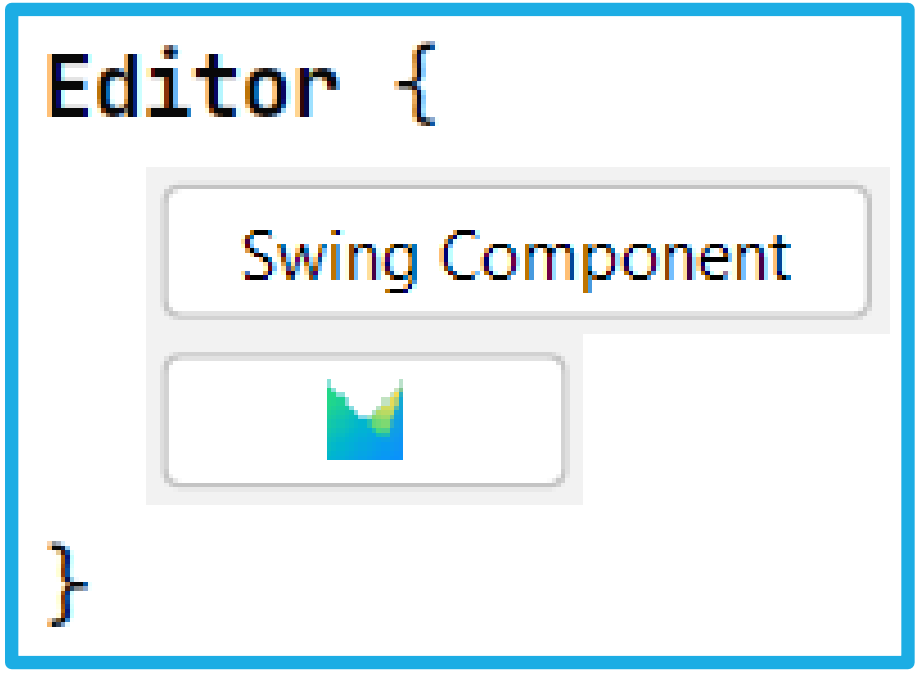
Editor Cell, Swing Components or Pure Graphical Presentation

- MPS language with editors for every NSD object
- Swing layout of NSD objects
- Image of the rendered NSD



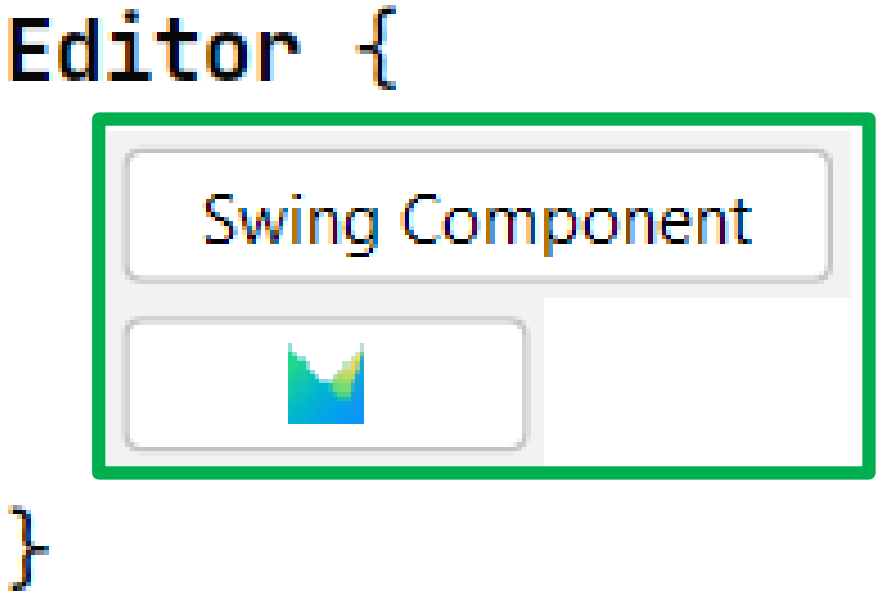
Editor Cell, Swing Components or Pure Graphical Presentation

- Editor cells for every NSD object
- Swing layout of NSD objects
- Image of the rendered NSD



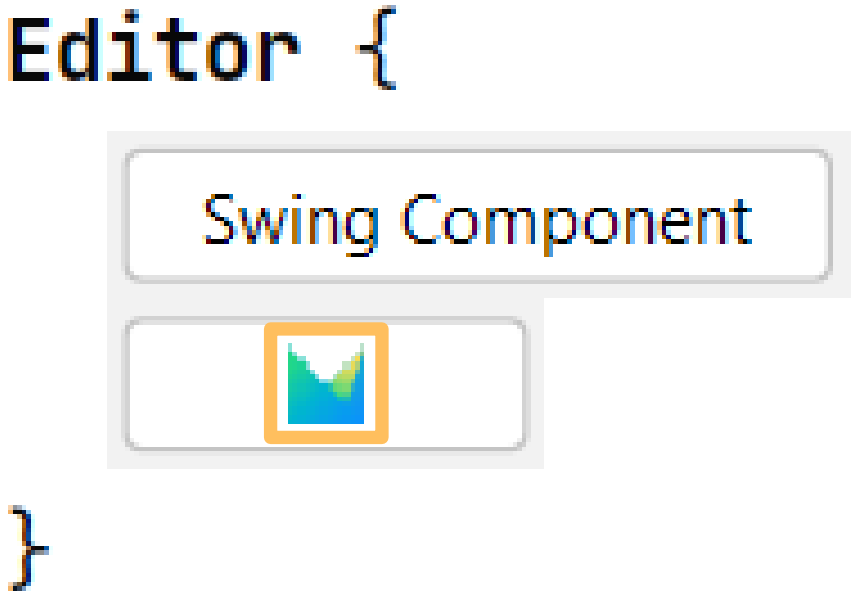
Editor Cell, Swing Components or Pure Graphical Presentation

- MPS language with editors for every NSD object
- Swing layout of NSD objects
- Image of the rendered NSD



Editor Cell, Swing Components or Pure Graphical Presentation

- MPS language with editors for every NSD object
- Swing layout of NSD objects
- Image of the rendered NSD



Represent the NSD Object as an Editor Cell

Editor Cell

- Seamless Integration
- Highly modifiable layout managers
- Best maintainability
- Embeddable cells
- In place editing
- Very difficult integration with library
- Implementation of custom cells and custom cell layouts

Represent the NSD Object as a Swing Component

Swing Component

- Good Integration with MPS
- In-built Layout Manager
- Fine-grained interaction
- Medium difficulty of integration with library
- Half step between graphical and editor level presentations

Represent the NSD Object as Pure Graphics?

Pure Graphics

- Library works on this level
- One level of presentation
- Fast deployment to MPS
- Simple to maintain
- Very simple layout manager
- Manual:
  - Drawing (Font/Size Management)
  - Interaction (Mouse Clicking)
  - Layouts

Represent the NSD Object as an Editor Cell, Swing Component or Pure Graphics?

Editor Cell

- Best Integration
- Flexible layout managers
- Best maintainability
- Embeddable cells
- In place editing
- Very difficult integration
- Implementation of custom cells and custom cell layouts

Swing Component

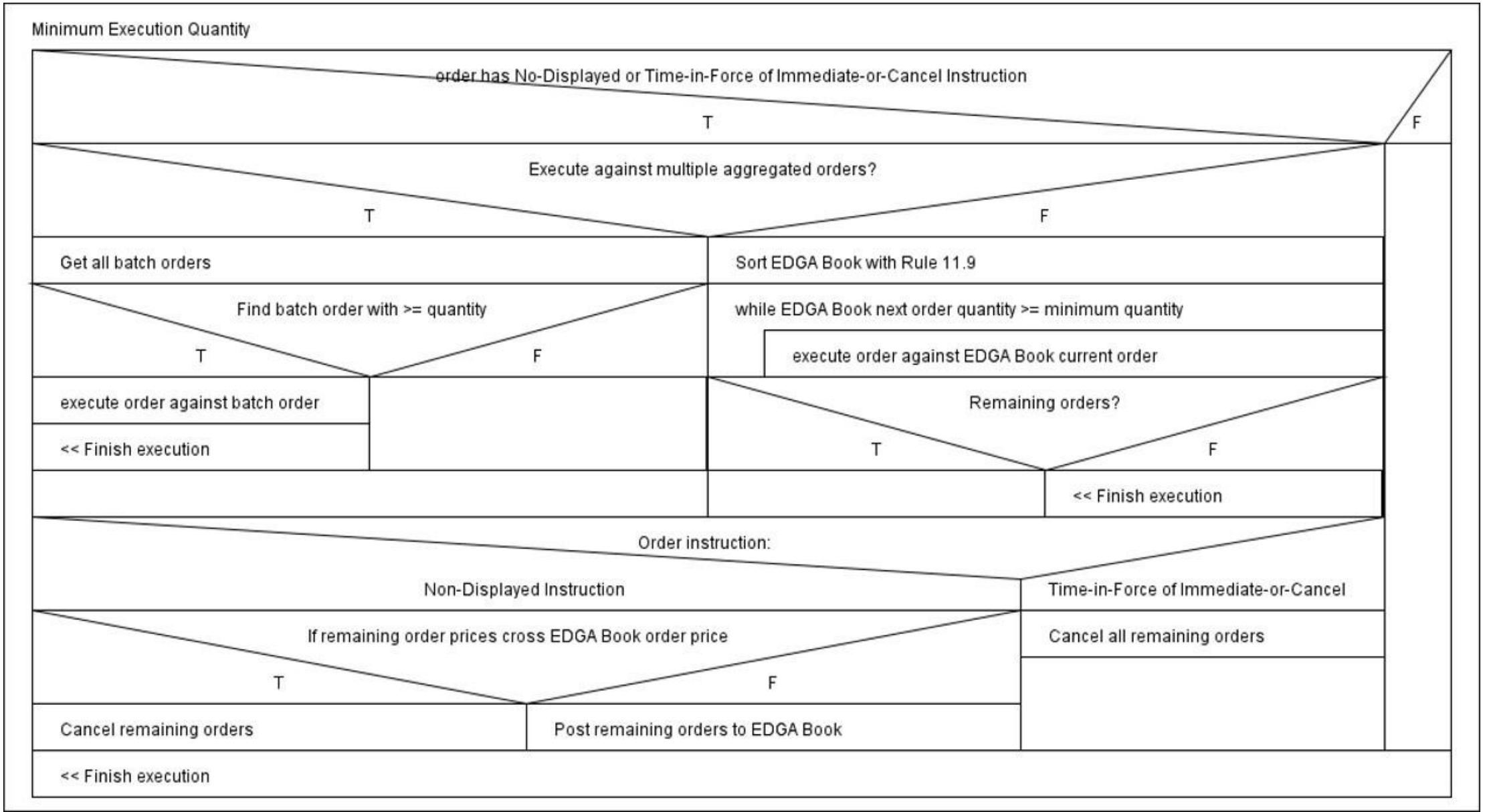
- Okay Integration
- In-built Layout Manager
- Fine-grained interaction
- Medium difficulty of integration
- Half step between graphical and editor level presentations

Pure Graphics

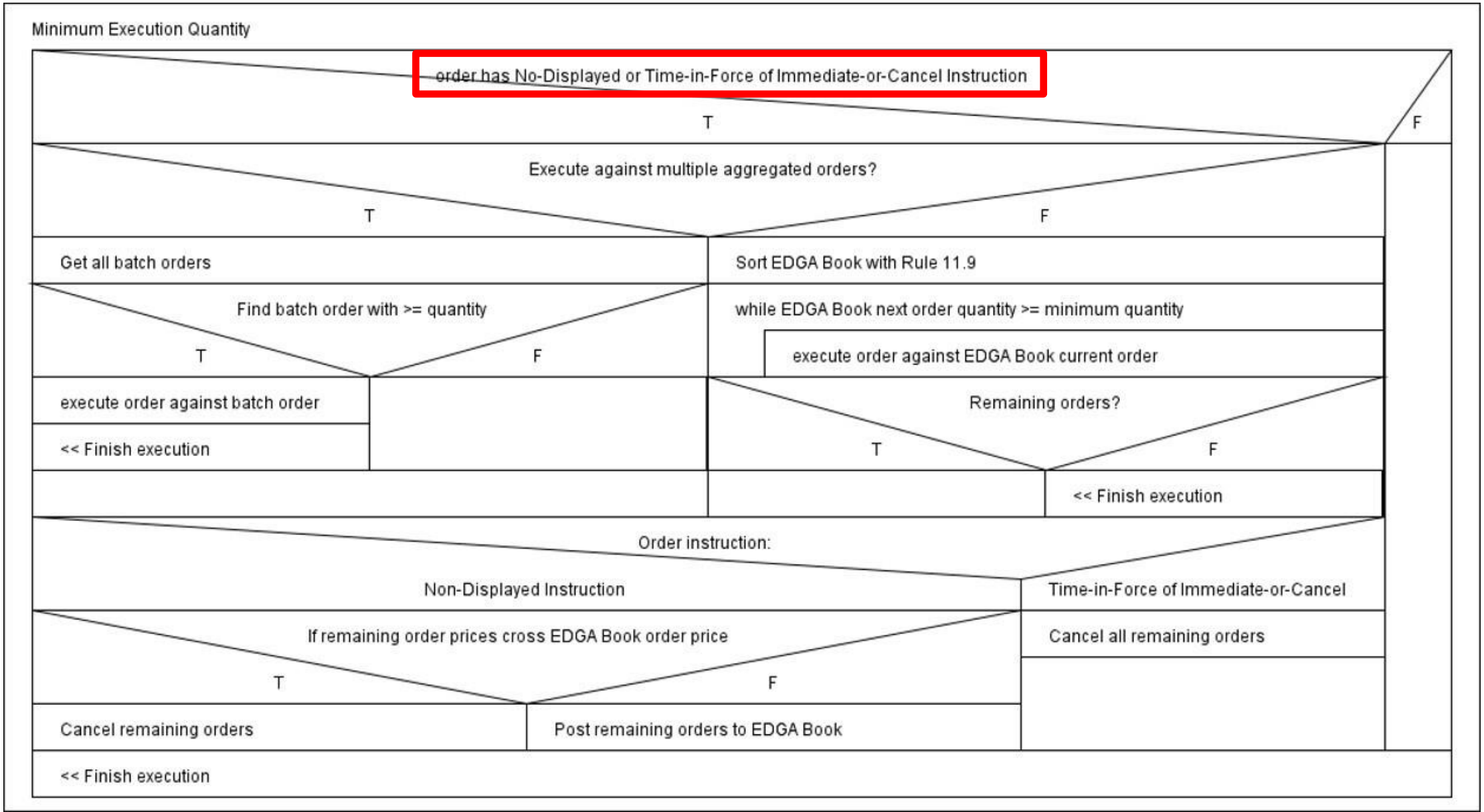
Winner

- Works out of the box
- One level of presentation
- Fast deployment to MPS
- Simple to maintain
- Very simple layout manager
- Manual:
  - Drawing (Font/Size Management)
  - Interaction (Mouse Clicking)
  - Layouts

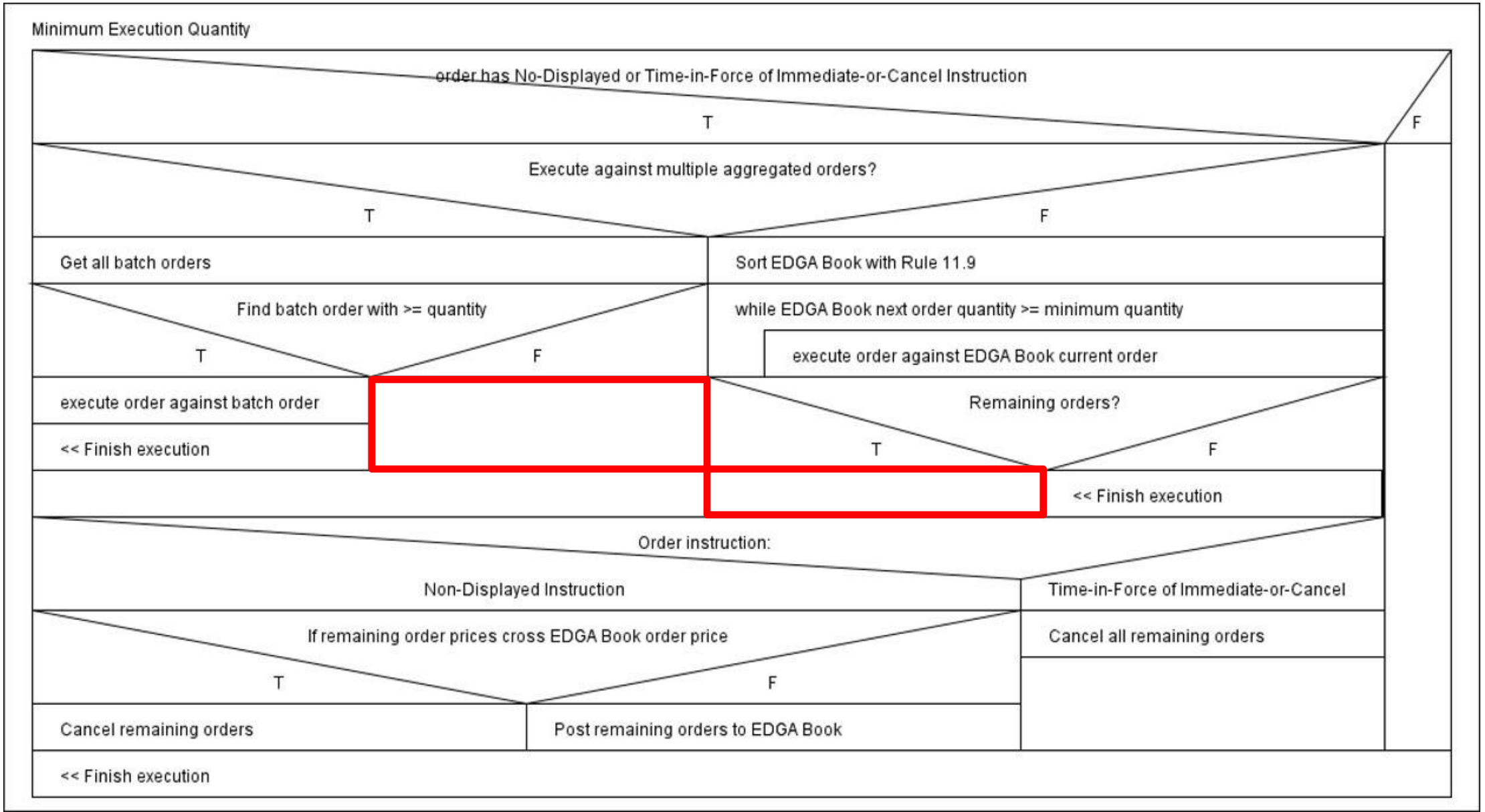
Pure Graphics Presentation



Pure Graphics Presentation



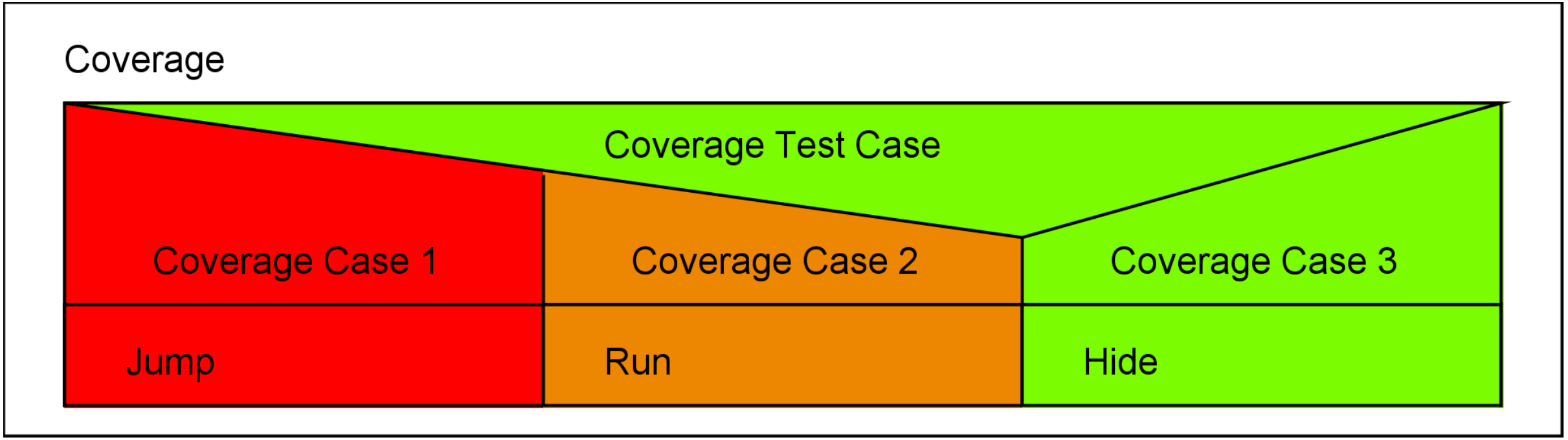
Pure Graphics Presentation



Code coverage for KernelF

- Interpreter supports coverage out-of-the-box

Code coverage side effect



Code coverage for KernelF

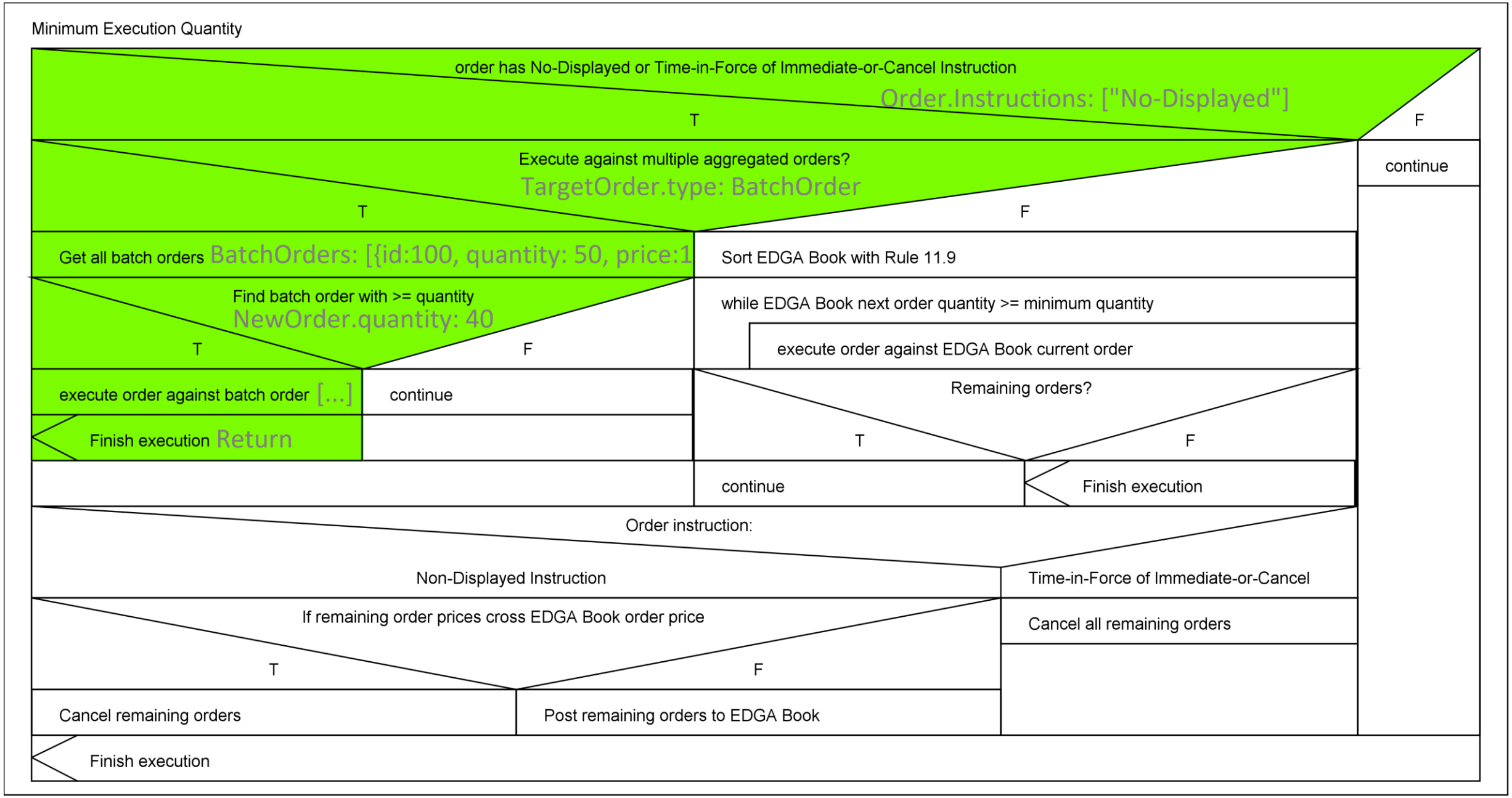
- Interpreter supports coverage out-of-the-box
- Extending the NSD plugin to other interpreted languages is fairly simple

BaseLanguage

Future

- Swing → MPS Editor cells
- Debugger integration

Debugger Integration



Questions

Links

- NSD Whitepaper - https://dl.acm.org/doi/10.1145/953349.953350
- Original library - https://github.com/meyfa/nsdlib
- My forked library - https://github.com/AtanasM98/nsdlib
- KernelF - https://voelter.de/data/pub/kernelf-reference.pdf

atanas@f1re.io