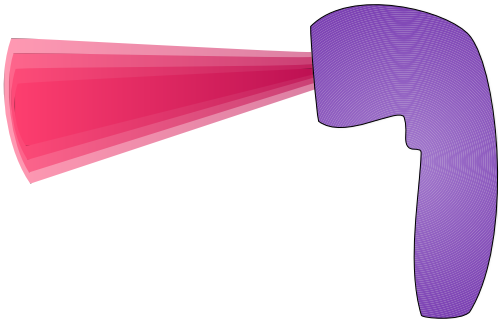
Box-moving Functional Editor in Barcode solution Program

- Company Elmicron
- Application ScanLink
- Configuration editor
- Process evaluation
- vwait
- TCL Activities
- TCL 9



	ID	Daten	Kommentar
Symbologie:	]d1	Datamatrix	Symbologie Datamatrix vom Le
Eingabedaten:		+HIBCAQ7B51/$D4467	
Strukturtyp:		HIBC	Health Industry Bar Code
Kennzeichner:	+	HIBC	Labeler ID Issuing Agency: H Interpretierte Daten: RHHIBC
Artikel:		AQ7B5	
erpackungsindex:		1	Unterste Logistische Verpack
Charge:	/$	D446	
en Sekundärcode:		7	Modulo-43 Prüfzeichen korrek
			Resümee des letzten Scans
Resümee:			HIBC UDI konform

Harald.Oehlmann@elmicron.de
Berlin/Germany
Chat: oehhar, Wiki: hao

Company Elmicron

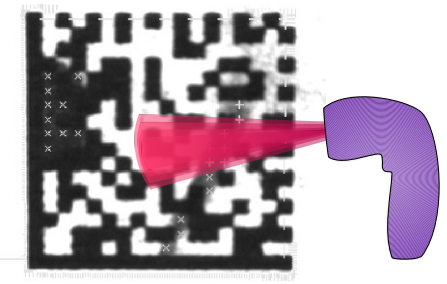
Auto ID
→ Barcode and RFID
→ ISO Standards

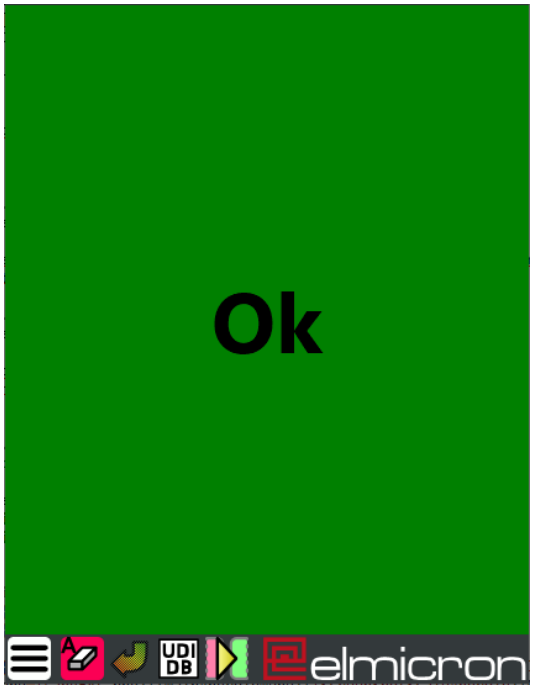
5 Employees

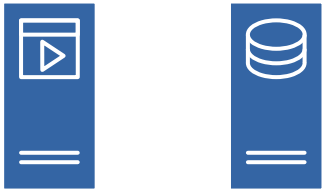
TclDevKit Licence

Application ScanLink

- Barcode verification and integration software
  Scanner -> ScanLink ->
  Keyboard emulation, Data Base, File, JSON/SOAP WebService
- Platforms: MS-Windows, Android, Linux 32 bit (on Thin Clients)
- TCL Starkits including self written C libraries for device connection
- Resizable Interface (ETCL 2019)
- Button keyboard for industrial touch-screen devices

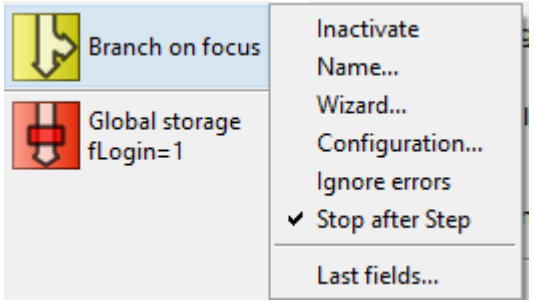




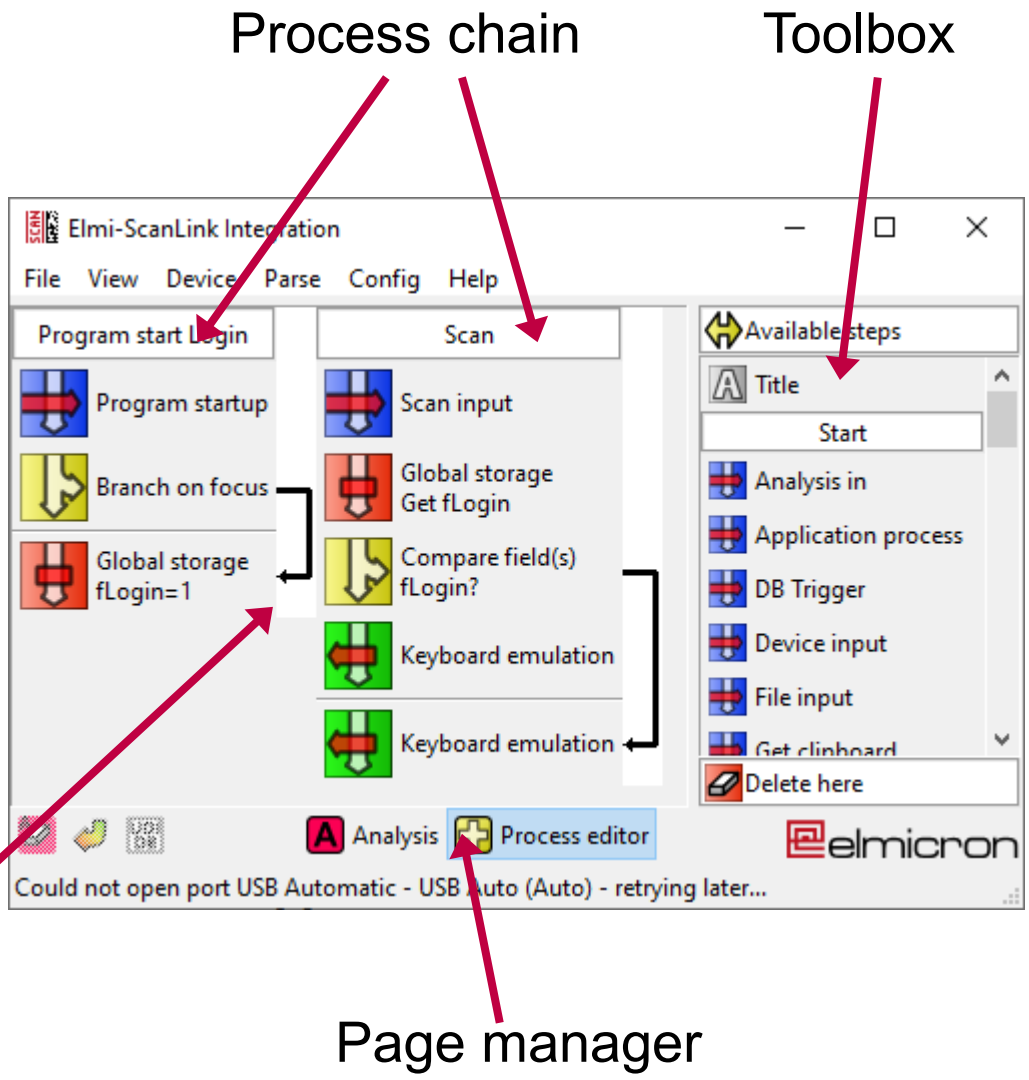


Box-moving configuration editor

- Processing steps from top to down including branch/jumps and stops
- Drag and drop interface (from Bwidget) for boxes and arrows
- menu with configuration points on right click (ttk menubuttons)



- Bubble-help everywhere
- Branch arrows in a canvas



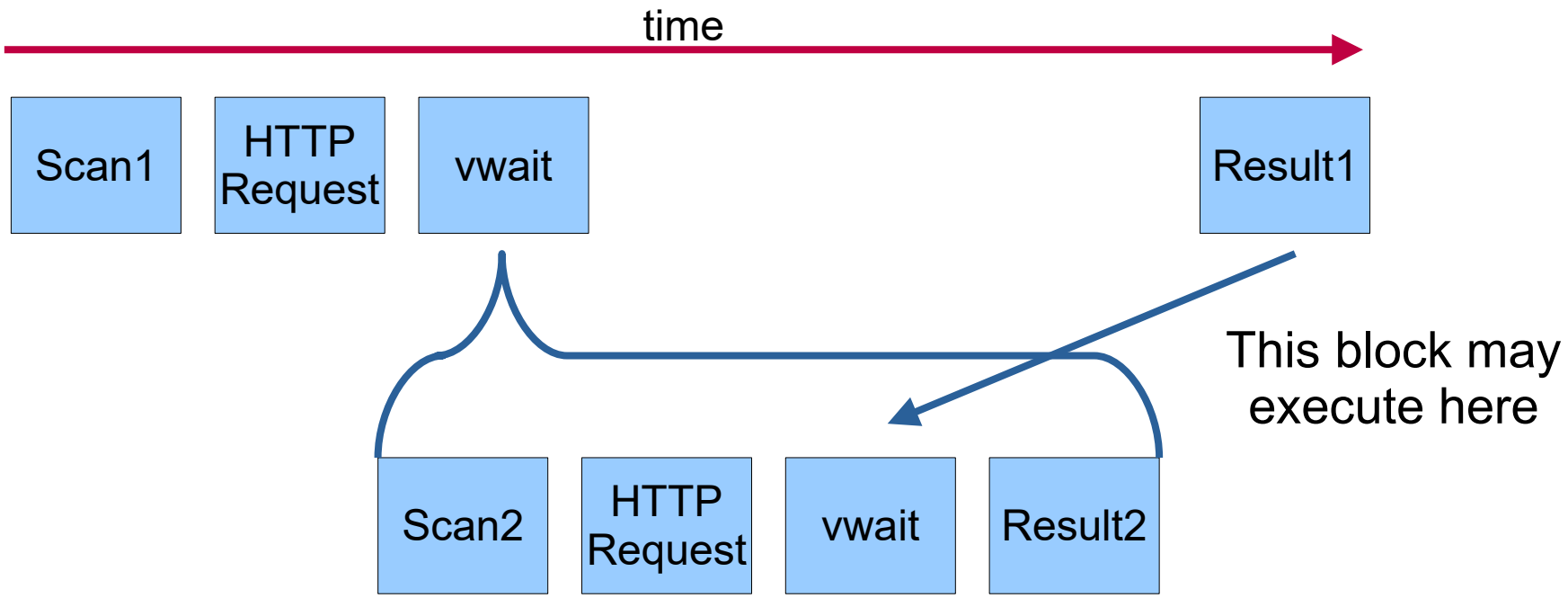
Process chains evaluation

- Follow the branch arrows
- Stop on end or end flag
- Pipe "**dData**" data dict through each step processing function by calling "**<Mod>DataIn**" proc of the module.
- Multiple process chains are started simultaneously and interlace on any wait steps.

```
while 1 {

    # >> Next step
    set fBranch [expr {
            "branch" in $dModuleOpt
            && [dict exists $dData _BranchDo]
    if { $fBranch } {
        set ClassPosCur [lsearch -exact
                $Classes\
                $Config(_BranchClass_$ClassCur)]
    } else {
        # > Check for end
        If { $ClassPosCur + 1
                        >= [llength $Classes]
                || $Config(_fStop_$ClassCur)
                || $Config(_fGroupEnd_$ClassCur)
        } {
            break
        }
        incr ClassPosCur
    }

    # >> Get new class
    set ClassCur [lindex $Classes $ClassPosCur]
    set ModuleCur $Config(_Module_$ClassCur)
    set ModuleClassCur $Config(_Class_$ClassCur)

    # >> Execute action
    set dData [${ModuleCur}DataIn\
            $ModuleClassCur $dData]
}
```

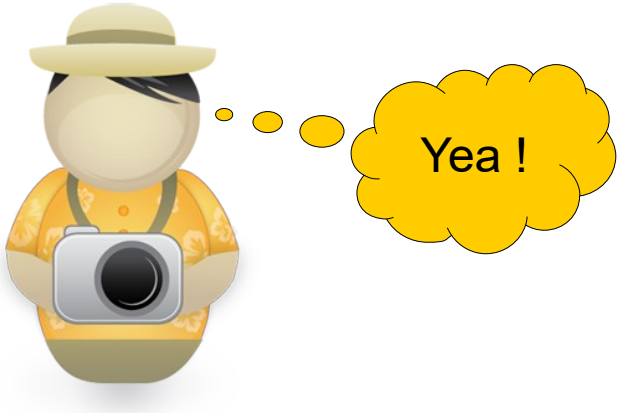
"vwait" may block a process chain

- "vwait" is very easy to use for simultaneously processing
- Nested "vwait" blocks any vwait in the caller proc and does not use eventual free time.
- "coroutines" are great but feel to complicated for me
- For known stops (Windows input) stop process chain and restart it.
- Any vwait is a problem (ex: http package), register events and return



TCL/Tk activities

- Bwidget (from Eric Boudallier)
- TCLWS (from Gerald Lester et al)
- Msgcat (from Marc Harrisson)
- TCL win socket driver: hope for Ashoks iopc
- Rivet (only support)
- TkSVG (from Christian Gollwitzer)
- MS-VC6++ PSDK2003SP1 TCL/Tk make (will die)
- Zint Bar code generator TCL interface
- TkSVG (Tk8.7 feature for Tk8.6)



TCL Wishlist

- How to get TCL 9.0 and Tk8.7 out of the door?
- Blockers: encodings
- How to help and motivate Donald to make releases?

Thank you all, you guys rock !