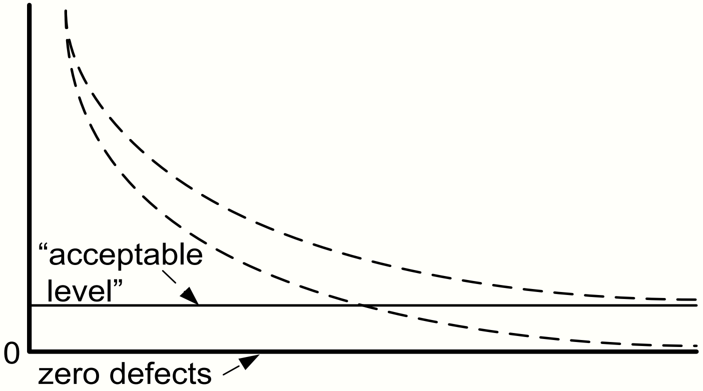


How to move towards Zero Defects

Niels Malotaux:
In my experience the 'zero defects' attitude results in 50% less defects almost overnight

Niels Malotaux

Copyright © 2022 by Niels Malotaux niels@malotaux.eu www.malotaux.eu/?id=zerodefects

Niels Malotaux

- Independent Project and Organizational Coach
- Expert in helping projects and organizations to quickly become
  - More effective - doing the right things better
  - More efficient - doing the right things better in less time
  - More predictable - delivering as needed
- Getting projects back on track
- Embedded Systems architect (electronics/firmware)
- Project types
  electronic products, firmware, software,
  space, road, rail, telecom,
  industrial control, parking system

Result Management

Do we deliver Zero Defect software ? ✓ yes ✗ no

- How many defects are acceptable ?
- Apparently the requirements specify a certain number of defects
- Do you check that the required number has been produced ?

In your projects

- How much time is spent putting defects in ?
- How much time is spent trying to find and fix them ?
- Do you sometimes get repeated issues ? ✓ yes ✗ no
- How much time is spent on defect prevention ?

Better quality costs less

What is a defect ?

A defect is the cause of a problem

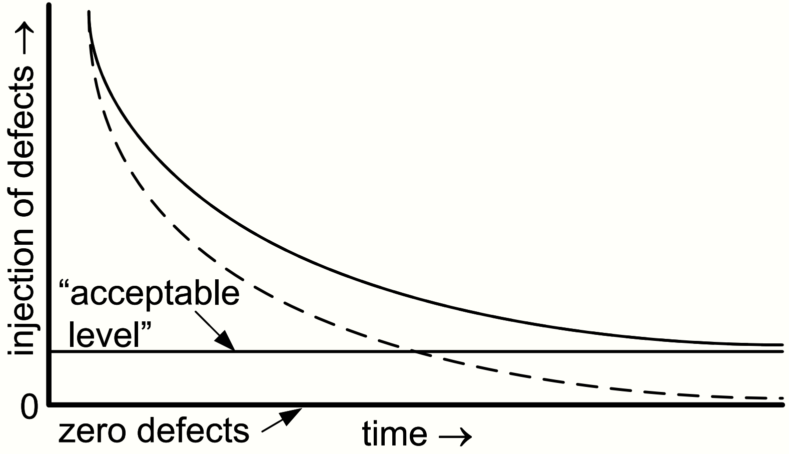
experienced by any of the stakeholders

while relying on our results

Is Zero Defects possible ? ✓ ✗ yes no

We aren’t perfect,
but the customer shouldn't find out

- Zero Defects is an asymptote



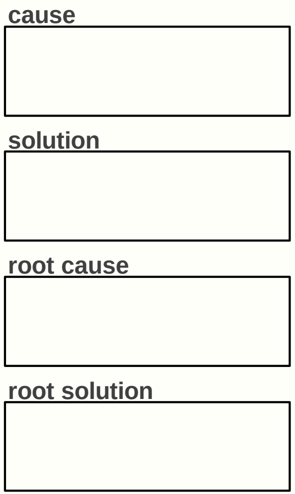
Zero Defects
= no hassle

Hassle: problem, unnecessary difficulty

- When Philip Crosby started with Zero Defects in 1961, errors dropped by 40% almost immediately

- AQL > Zero means that the organization has settled on a level of incompetence
Causing a hassle other people have to live with

Prevention: Root Cause Analysis



cause

solution

root cause

root solution

- Is Root Cause Analysis routinely performed - every time ?
- What is the difference between *Cause* and *Root Cause* ?

- Cause:
  The error that caused the problem
- Root Cause:
  What caused *us* to make the error that caused the problem

- Without proper Root Cause Analysis, we're *doomed to repeat the same errors*

Does Testing ‘assure’ quality ?

Deming
(1900-1993)

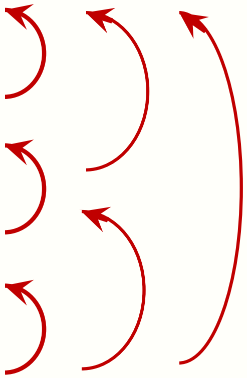
- Some testers call themselves QA: ‘Quality Assurance’
- Can testers assure quality ? ✓ ✗ yes no
- Deming:
  - Quality comes not from testing, but from *improvement of the development process*
  - Testing does *not* improve quality, nor guarantee quality
  - It’s too late
  - The quality, good or bad, is already in the product
  - *You cannot test quality into a product*
- So, how to create quality ?

Some Examples

We’re not perfect,
but the customer shouldn’t find out

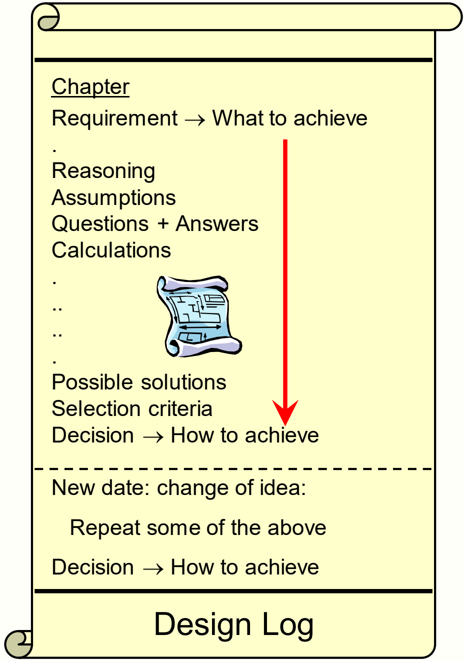
Design techniques

- Requirement
- Review
- Design
- Review
- Code
- Review



Iterate as needed

- Integration test (no questions, no issues)
- If issue in test: no Band-Aid: start all over again: Review: What’s wrong with the design ?
- Reconstruct the design (if the design description is lacking)
  - What happens if you ask "Can I see the DesignLog ?"



Chapter

Requirement → What to achieve

.

Reasoning

Assumptions

Questions + Answers

Calculations

.

..

..

.

Possible solutions

Selection criteria

Decision → How to achieve

New date: change of idea:

Repeat some of the above

Decision → How to achieve

Design Log

Cleanroom

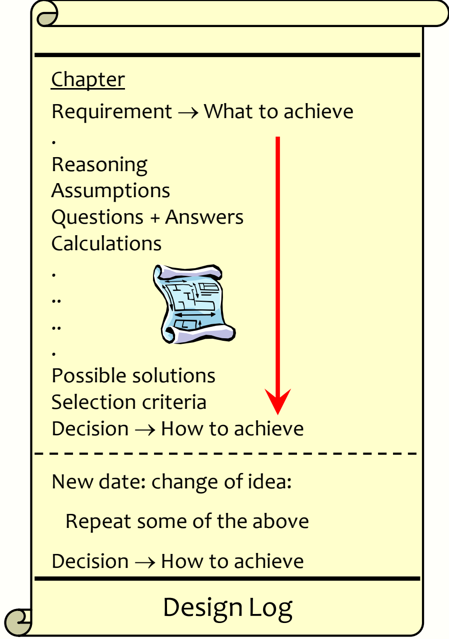
Case: In the pub

James:

Niels, this is Louise
Louise, this is Niels, who taught me about DesignLogging
Tell what happened

Louise:

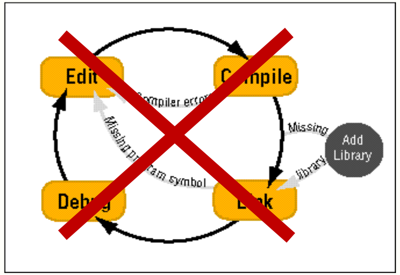
- We had only 7 days to finish some software
- We were working hard, coding, testing, coding, testing
- James said we should stop coding and go back to the design
- "We don't have time!" - "We've only 7 days!"
- James insisted
- We designed, found the problem, corrected it, cleaned up the mess
- Done in less than 7 days
- Thank you!



Chapter
Requirement → What to achieve
.
Reasoning
Assumptions
Questions + Answers
Calculations
.
..
..
.
Possible solutions
Selection criteria
Decision → How to achieve

New date: change of idea:
Repeat some of the above
Decision → How to achieve

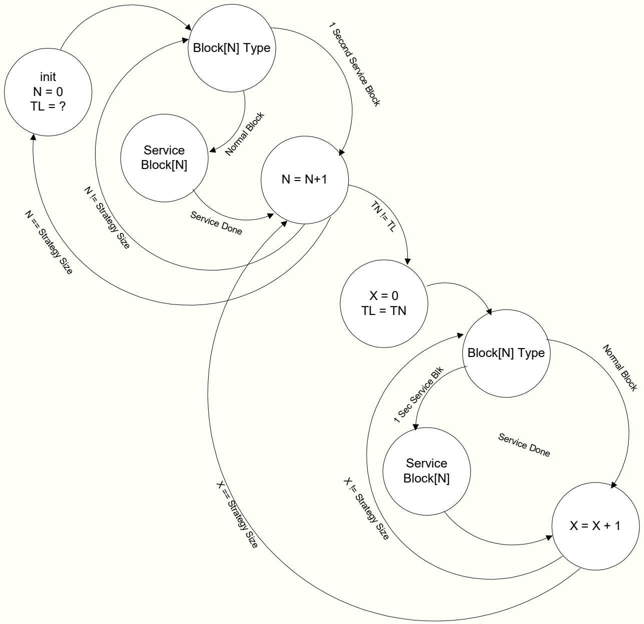
Design Log



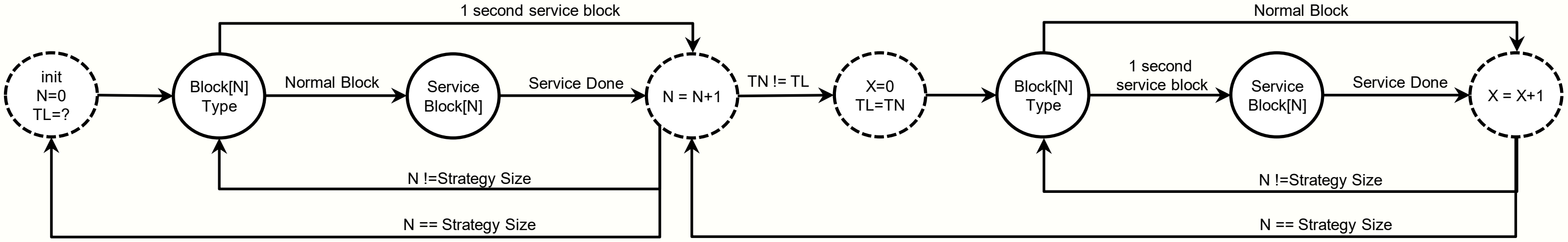
What James told me afterwards

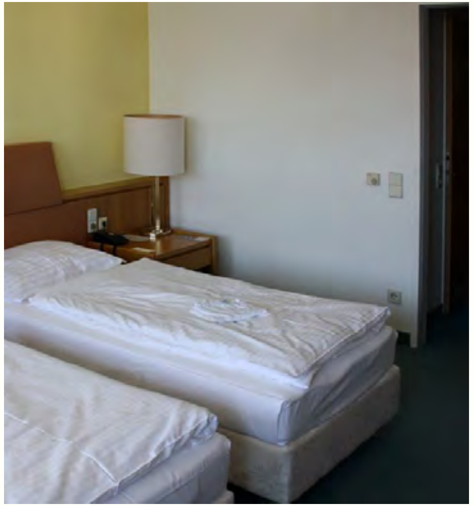
- I gave the design to two colleagues for review
- Louise corrected some minor issues
- It went into a ‘final’ review, with another colleague
- Based in his expertise, the solution was completely reworked
- Actually, two features were delivered and deployed
  - One that was design and code reviewed had no issues after deployment
  - Other one, was the source of quite some defects
- This success proved instrumental in buy-in for DesignLogs which are now embedded in the development process

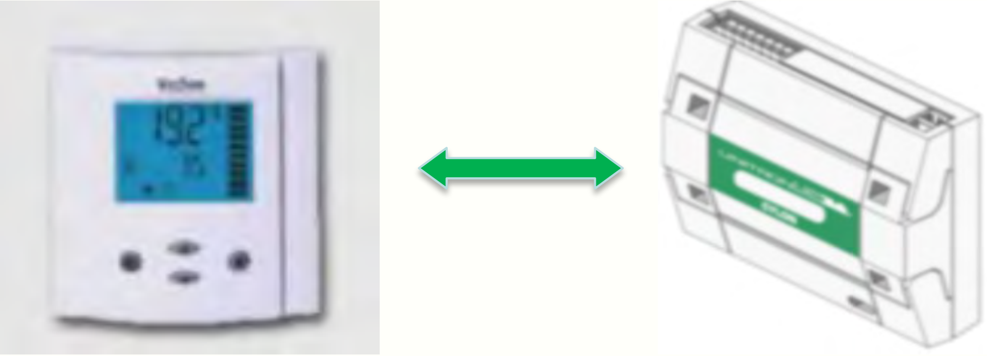
There are many ways to represent a design



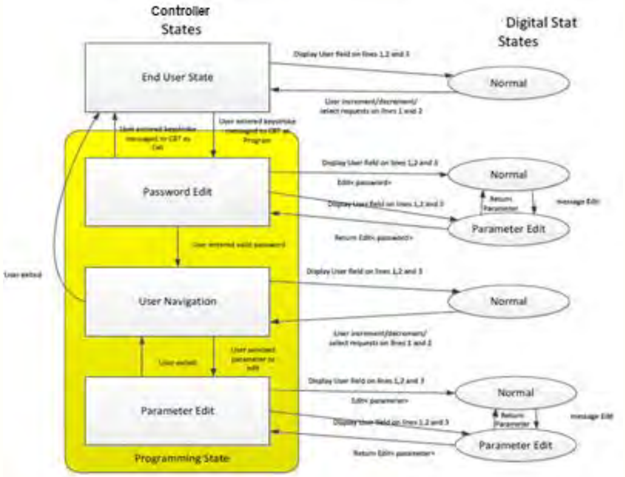
- Only few are useful
- Don't waste reviewer's time

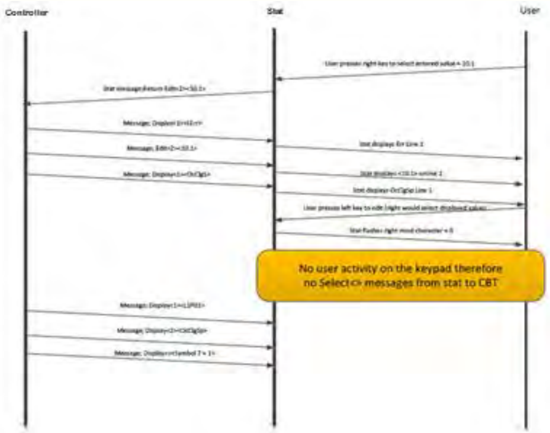






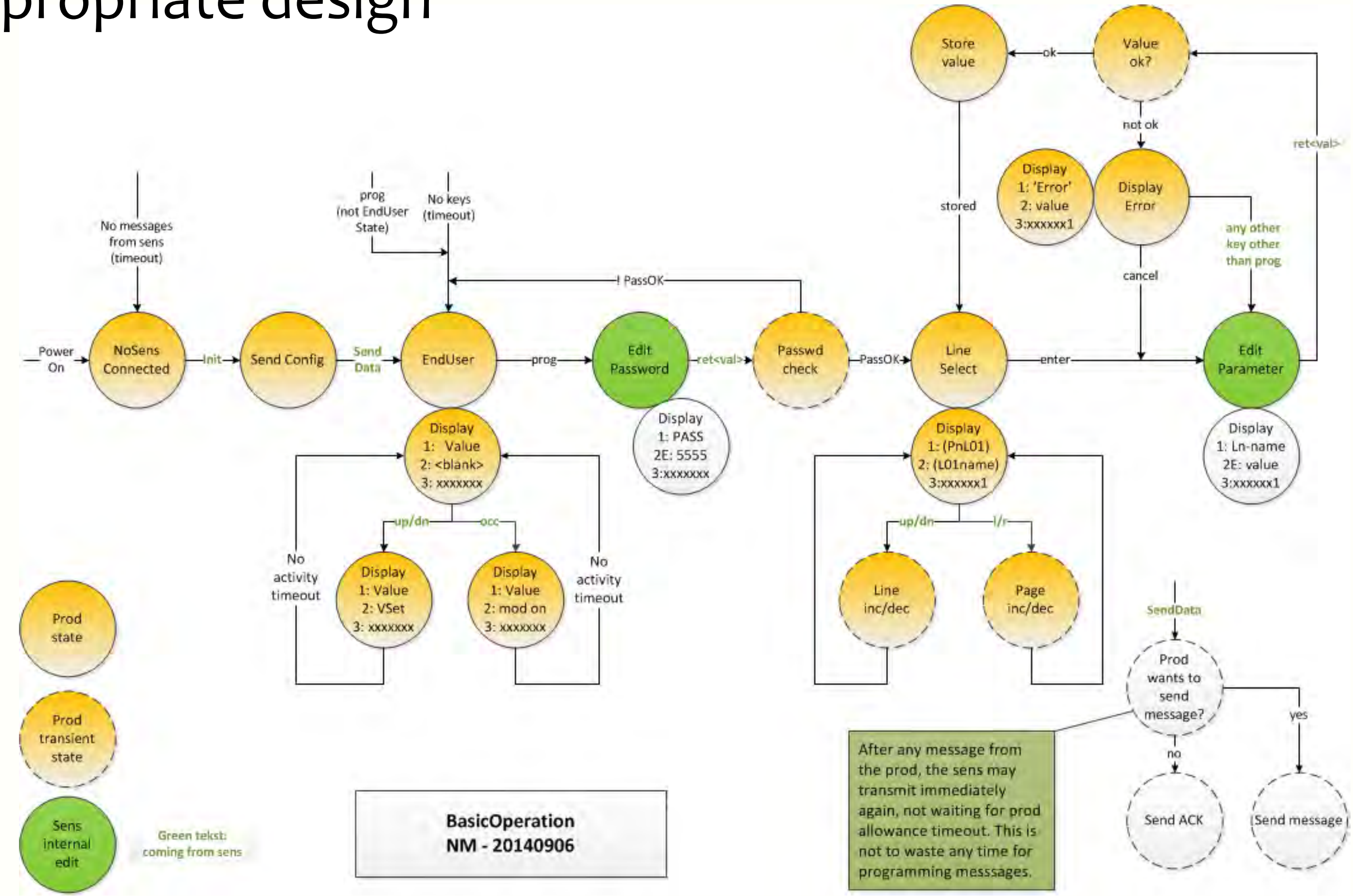
Useful design ?




Controller
Stat
User
No user activity on the keypad therefore no Select<> messages from stat to CBT

Choose the appropriate design

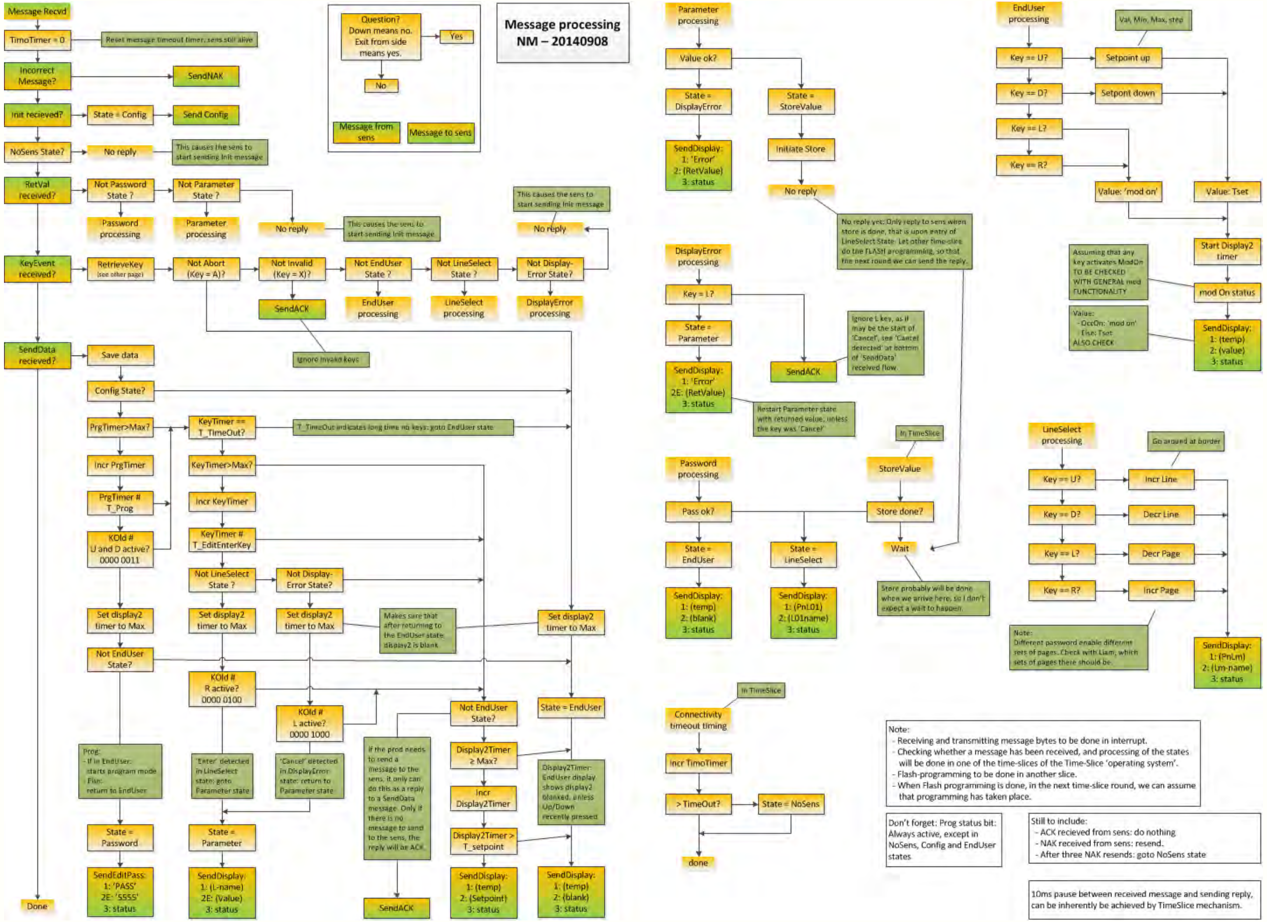
47 pages documentation
condensed into one page



After any message from the prod, the sens may transmit immediately again, not waiting for prod allowance timeout. This is not to waste any time for programming messsages.

BasicOperation
NM - 20140906

Translating into flow


Message processing
NM – 20140908
Question? Down means no. Exit from side means yes.
Yes
No
Message from sens
Message to sens
Message Recvd
TimoTimer = 0
Reset message timeout timer: sens still alive
Incorrect Message?
SendNAK
Init recieved?
State = Config
Send Config
NoSens State?
No reply
This causes the sens to start sending Init message
RetVal received?
Not Password State ?
Not Parameter State ?
Password processing
Parameter processing
No reply
This causes the sens to start sending Init message
This causes the sens to start sending Init message
No reply
KeyEvent received?
RetrieveKey (see other page)
Not Abort (Key = A)?
Not Invalid (Key = X)?
Not EndUser State ?
Not LineSelect State ?
Not Display-Error State?
SendACK
EndUser processing
LineSelect processing
DisplayError processing
Ignore invalid keys
SendData recieved?
Save data
Config State?
PrgTimer>Max?
KeyTimer == T_TimeOut?
T_TimeOut indicates long time no keys: goto EndUser state
Incr PrgTimer
KeyTimer>Max?
PrgTimer # T_Prog
Incr KeyTimer
KOld # U and D active? 0000 0011
KeyTimer # T_EditEnterKey
Not LineSelect State ?
Not Display-Error State?
Set display2 timer to Max
Set display2 timer to Max
Set display2 timer to Max
Makes sure that after returning to the EndUser state display2 is blank
Set display2 timer to Max
Not EndUser State?
KOld # R active? 0000 0100
KOld # L active? 0000 1000
Not EndUser State?
State = EndUser
Prog: - If in EndUser: starts program mode - Else: return to EndUser
'Enter' detected in LineSelect state: goto Parameter state
'Cancel' detected in DisplayError state: return to Parameter state
If the prod needs to send a message to the sens, it only can do this as a reply to a SendData message. Only if there is no message to send to the sens, the reply will be ACK.
Display2Timer ≥ Max?
Incr Display2Timer
Display2Timer > T_setpoint
Display2Timer: EndUser display shows display2 blanked, unless Up/Down recently pressed
State = Password
State = Parameter
SendEditPass: 1: 'PASS' 2E: '5555' 3: status
SendDisplay: 1: (L-name) 2E: (Value) 3: status
SendACK
SendDisplay: 1: (temp) 2: (Setpoint) 3: status
SendDisplay: 1: (temp) 2: (blank) 3: status
Done
Parameter processing
Value ok?
State = DisplayError
State = StoreValue
SendDisplay: 1: 'Error' 2: (RetValue) 3: status
Initiate Store
No reply
No reply yet: Only reply to sens when store is done, that is upon entry of LineSelect State. Let other time-slice do the FLASH programming, so that the next round we can send the reply.
DisplayError processing
Key = L?
State = Parameter
SendACK
Ignore L key, as it may be the start of 'Cancel', see 'Cancel detected' at bottom of 'SendData' received flow.
SendDisplay: 1: 'Error' 2E: (RetValue) 3: status
Restart Parameter state with returned value, unless the key was 'Cancel'
In TimeSlice
StoreValue
Password processing
Pass ok?
Store done?
State = EndUser
State = LineSelect
Wait
SendDisplay: 1: (temp) 2: (blank) 3: status
SendDisplay: 1: (PnL01) 2: (L01name) 3: status
Store probably will be done when we arrive here, so I don't expect a wait to happen.
In TimeSlice
Connectivity timeout timing
Incr TimoTimer
> TimeOut?
State = NoSens
done
EndUser processing
Val, Min, Max, step
Key == U?
Setpoint up
Key == D?
Setpont down
Key == L?
Key == R?
Value: 'mod on'
Value: Tset
Start Display2 timer
Assuming that any key activates ModOn TO BE CHECKED WITH GENERAL mod FUNCTIONALITY
mod On status
Value: - OccOn: 'mod on' - Else: Tset ALSO CHECK
SendDisplay: 1: (temp) 2: (value) 3: status
LineSelect processing
Go around at border
Key == U?
Incr Line
Key == D?
Decr Line
Key == L?
Decr Page
Key == R?
Incr Page
Note: Different password enable different sets of pages. Check with Liam, which sets of pages there should be.
SendDisplay: 1: (PnLm) 2: (Lm-name) 3: status

Note:
- Receiving and transmitting message bytes to be done in interrupt.
- Checking whether a message has been received, and processing of the states will be done in one of the time-slices of the Time-Slice 'operating system'.
- Flash-programming to be done in another slice.
- When Flash programming is done, in the next time-slice round, we can assume that programming has taken place.

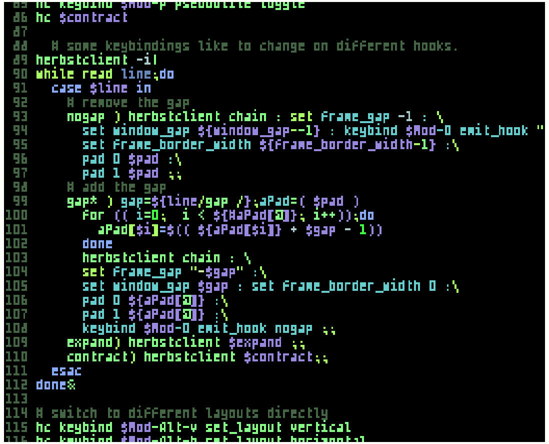
Don't forget: Prog status bit: Always active, except in NoSens, Config and EndUser states.

Still to include:
- ACK recieved from sens: do nothing
- NAK received from sens: resend.
- After three NAK resends: goto NoSens state

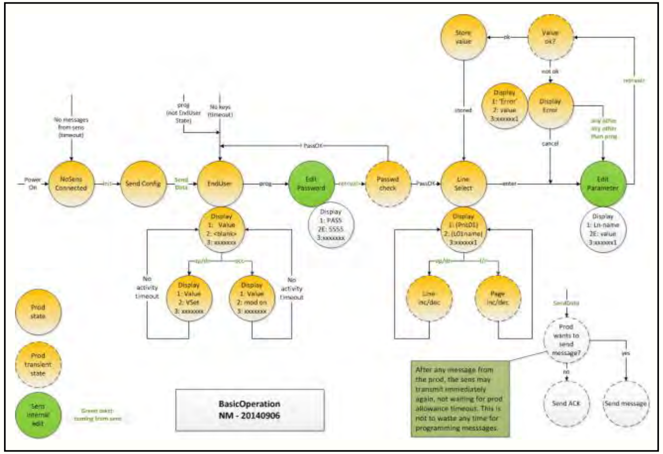
10ms pause between received message and sending reply, can be inherently be achieved by TimeSlice mechanism.

What is better than reviewing code ?

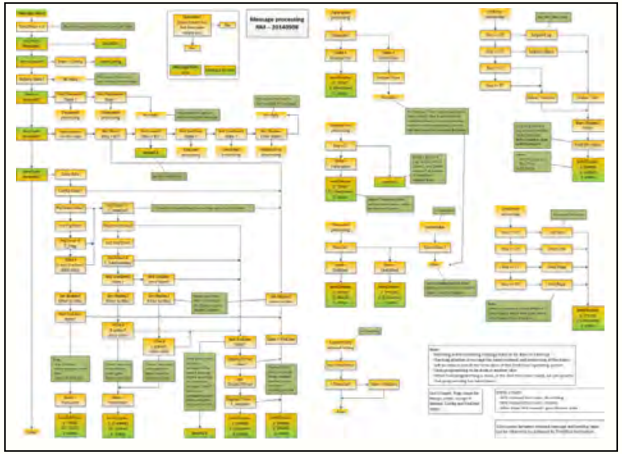
- If you review software, what do you review ?
- What is better than reviewing code ?
  - May I review the design first ?

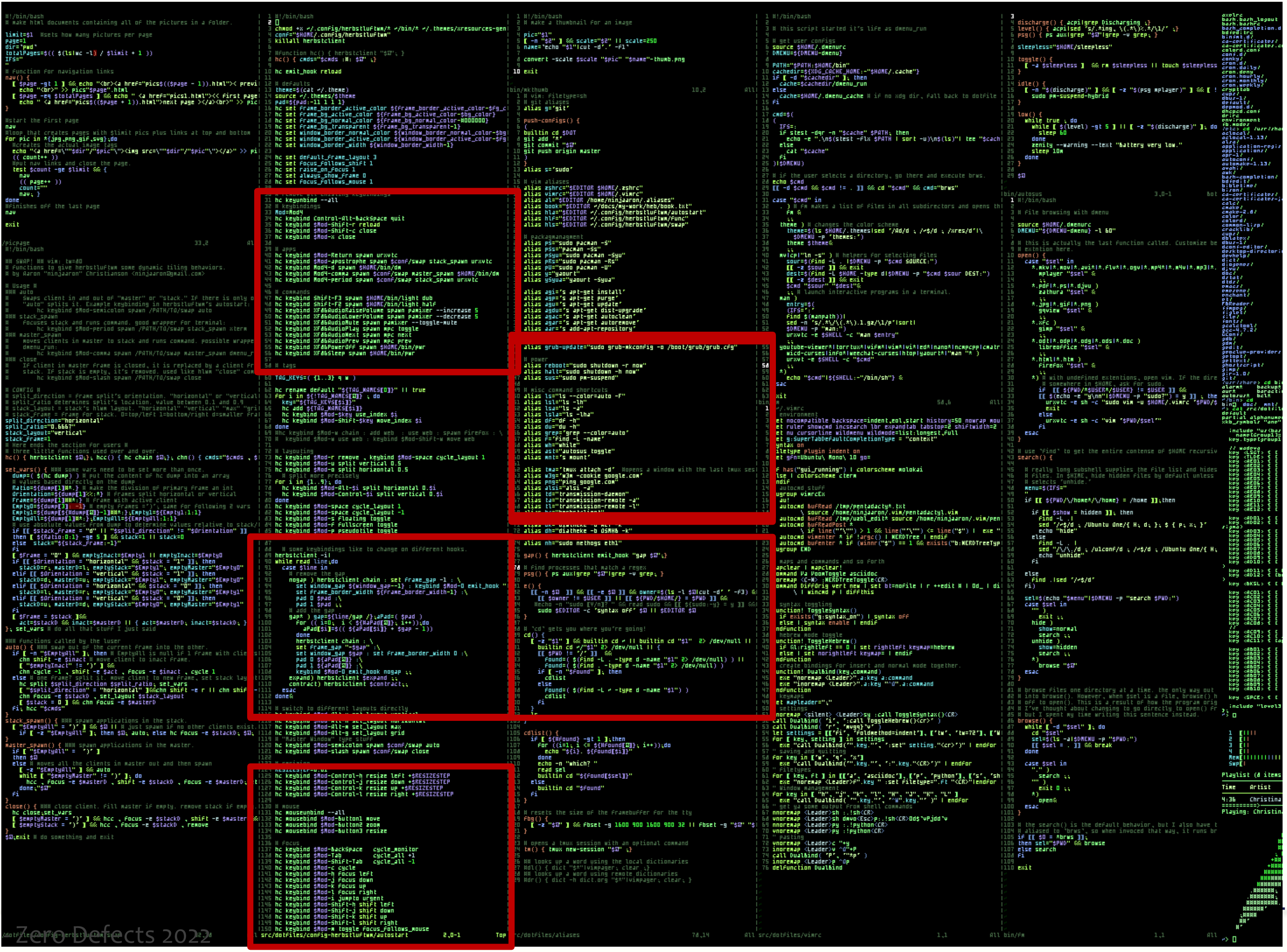


```
hc keybind $Mod-p pseudotile toggle
hc $contract

  # some keybindings like to change on different hooks.
herbstclient -il
while read line;do
  case $line in
    # remove the gap
    nogap ) herbstclient chain : set frame_gap -1 : \
      set window_gap ${window_gap--1} : keybind $Mod-0 emit_hook "
      set frame_border_width ${frame_border_width-1} :\
      pad 0 $pad :\
      pad 1 $pad ;;
    # add the gap
    gap* ) gap=${line/gap /};aPad=( $pad )
      for (( i=0;  i < ${#aPad[@]}; i++));do
        aPad[$i]=$(( ${aPad[$i]} + $gap - 1))
      done
      herbstclient chain : \
      set frame_gap "-$gap" :\
      set window_gap $gap : set frame_border_width 0 :\
      pad 0 ${aPad[@]} :\
      pad 1 ${aPad[@]} :\
      keybind $Mod-0 emit_hook nogap ;;
    expand) herbstclient $expand ;;
    contract) herbstclient $contract;;
  esac
done&

# switch to different layouts directly
hc keybind $Mod-Alt-v set_layout vertical
hc keybind $Mod-Alt-h set_layout horizontal
```



After any message from the prod, the sens may transmit immediately again, not waiting for prod allowance timeout. This is not to waste any time for programming messsages.





```
#!/bin/bash
# make html documents containing all of the pictures in a folder.

limit=$1  #sets how many pictures per page
page=1
dir=`pwd`
totalPages=$(( $(ls|wc -l) / $limit + 1 ))
IFS="
"
# function for navigation links
nav() {
  [ $page -gt 1 ] && echo "<br><a href="pics$(($page - 1)).html">< previ
    echo "<br>" >> pics"$page".html
  [ $page -eq $totalPages ] && echo " <a href="pics1.html"><< first page
    echo " <a href="pics$(($page + 1)).html">next page ></a><br>" >> pic
}

#start the first page
nav
#loop that creates pages with $limit pics plus links at top and bottom
for pic in *{jpg,png,gif,svg};do
  #creates the actual image tags
  echo "<a href=\""$dir"/"$pic"\"><img src=\""$dir"/"$pic"\"></a>" >> pi
  (( count++ ))
  #put nav links and close the page.
  test $count -ge $limit && {
    nav
    (( page++ ))
    count=""
    nav; }
done
#finishes off the last page
nav

exit


/picpage                                                33,2          Al
```

```
#!/bin/bash

## SWAP! ## vim: tw=80
# functions to give herbstluftwm some dynamic tiling behaviors.
# by Aaron "ninjaaron" Christianson <ninjaaron@gmail.com>

# Usage #
### auto
#    Swaps client in and out of "master" or "stack." If there is only o
#    "auto" splits it. Example keybinding in herbstluftwm's autostart:
#        hc keybind $Mod-semicolon spawn /PATH/TO/swap auto
### stack_spawn
#    focuses stack and runs command. good wrapper for terminal:
#        hc keybind $Mod-period spawn /PATH/TO/swap stack_spawn xterm
### master_spawn
#    moves clients in master to stack and runs command. possible wrappe
#    dmenu_run:
#        hc keybind $Mod-comma spawn /PATH/TO/swap master_spawn dmenu_r
### close
#    If client in master frame is closed, it is replaced by a client fr
#    stack. If stack is empty, it's removed. used like hlwm "close" com
#        hc keybind $Mod-slash spawn /PATH/TO/swap close

# CONFIG #
# split_direction = frame split's orientation. "horizontal" or "vertical
# split_ratio determines split's location. value between 0.1 and 0.9
# stack_layout = stack's hlwm layout. "horizontal" "vertical" "max" "gri
# stack_frame = frame for stack. 0=top/left 1=bottom/right d=smaller fra
split_direction="horizontal"
split_ratio="0.6667"
stack_layout="vertical"
stack_frame=1
# Here ends the section for users #
# three little functions used over and over.
hc() { herbstclient $@;}; hcc() { hc chain $@;}; chn() { cmds="$cmds , $

set_vars() { ### some vars need to be set more than once.
 dump=( $(hc dump) ) # put the content of hc dump into an array
 # values based directly on the dump
 Ratio=${dump[1]#*.} # make the division of primary frame an int
 Orientation=${dump[1]%%:*} # frames split horizontal or vertical
 Frame=${dump[1]##*:} # frame with active client
 Empty0=${dump[3]: -1} # empty frames =')'; same for following 2 vars
 Empty1=${dump[${#dump[@]}-1]##*:};Empty1=${Empty1:1:1}
 EmptyAll=${dump[1]#*:};EmptyAll=${EmptyAll:1:1}
 # use absolute values from dump to determine values relative to stack/
 if [[ $stack_frame = "d" || "$split_direction" != "$Orientation" ]]
 then [ ${Ratio:0:1} -ge 5 ] && stack=1 || stack=0
 else  stack="${stack_frame:-1}"
 fi
 [ $Frame = "0" ] && emptyInact=$Empty1 || emptyInact=$Empty0
 if [[ $Orientation = "horizontal" && $stack = "1" ]]; then
   stackD=r; masterD=l; emptyStack="$Empty1"; emptyMaster="$Empty0"
 elif [[ $Orientation = "vertical" && $stack = "1" ]]; then
   stackD=d; masterD=u; emptyStack="$Empty1"; emptyMaster="$Empty0"
 elif [[ $Orientation = "horizontal" && $stack = "0" ]]; then
   stackD=l; masterD=r; emptyStack="$Empty0"; emptyMaster="$Empty1"
 elif [[ $Orientation = "vertical" && $stack = "0" ]]; then
   stackD=u; masterD=d; emptyStack="$Empty0"; emptyMaster="$Empty1"
 fi
 [ $Frame = $stack ]&&
   act=$stackD && inact=$masterD || { act=$masterD; inact=$stackD; }
}; set_vars # do all that stuff I just said

### functions called by the luser
auto() { ### swap out of the current frame into the other.
 if [ -n "$EmptyAll" ]; then # EmptyAll is null if 1 frame with clie s
   chn shift -e $inact # move client to inact frame.
   [ "$emptyInact" != ')' ] &&
   chn cycle -1 , shift -e $act , focus -e $inact , cycle 1
 else # one frame? split it. move client to new frame, set stack lay t
   hc split $split_direction $split_ratio; set_vars
   [ "$split_direction" = "horizontal" ]&&chn shift -e r || chn shif  -
   chn focus -e $stackD , set_layout $stack_layout
   [ $stack = 0 ] && chn focus -e $masterD
 fi; hcc "$cmds"
}
stack_spawn() { ### spawn applications in the stack.
 [ "$EmptyAll" = ')' ] && $@ || # just spawn if no other clients exist.
   if [ -z "$EmptyAll" ]; then $@; auto; else hc focus -e $stackD; $@;
}
master_spawn() { ### spawn applications in the master.
 if [ "$EmptyAll" =  ')' ]
 then $@
 else # moves all the clients in master out and then spawn
   [ -z "$EmptyAll" ] && auto
   while [ "$emptyMaster" != ')' ]; do
     hcc , focus -e $masterD , shift -e $stackD , focus -e $masterD; t
   done;"$@"
 fi
}
close() { ### close client. fill master if empty. remove stack if emp  .
 hc close;set_vars
 [ $emptyMaster = ')' ] && hcc , focus -e $stackD , shift -e $master &
 [ $emptyStack = ')' ] && hcc , focus -e $stackD , remove
}
$@;exit # do something and exit
```

```
#!/bin/bash

chmod +x ~/.config/herbstluftwm/* ~/bin/* ~/.themes/xresources-gen
conf="$HOME/.config/herbstluftwm"
killall herbstclient

#function hc() { herbstclient "$@"; }
hc() { cmds="$cmds :#: $@" ;}

hc emit_hook reload

# defaults
theme=$(cat ~/.theme)
source ~/.themes/$theme
pad=${pad:-11 1 1 1}
hc set frame_border_active_color ${frame_border_active_color-$fg_c
hc set frame_bg_active_color ${frame_bg_active_color-$bg_color}
hc set frame_bg_normal_color ${frame_bg_normal_color-#000000}
hc set frame_bg_transparent ${frame_bg_transparent-1}
hc set window_border_normal_color ${window_border_normal_color-$bg
hc set window_border_active_color ${window_border_active_color-$fg
hc set window_border_width ${window_border_width-1}

hc set default_frame_layout 3
hc set focus_follows_shift 1
hc set raise_on_focus 1
hc set always_show_frame 0
hc set focus_follows_mouse 1

hc keyunbind --all
# keybindings
Mod=Mod4
hc keybind Control-Alt-BackSpace quit
hc keybind $Mod-Shift-r reload
hc keybind $Mod-Shift-c close
hc keybind $Mod-x close

# apps
hc keybind $Mod-Return spawn urxvtc
hc keybind $Mod-apostrophe spawn $conf/swap stack_spawn urxvtc
hc keybind Mod4-d spawn $HOME/bin/dm
hc keybind Mod4-comma spawn $conf/swap master_spawn $HOME/bin/dm
hc keybind Mod4-period spawn $conf/swap stack_spawn urxvtc

# commands
hc keybind Shift-F3 spawn $HOME/bin/light dub
hc keybind Shift-F2 spawn $HOME/bin/light half
hc keybind XF86AudioRaiseVolume spawn pamixer --increase 5
hc keybind XF86AudioLowerVolume spawn pamixer --decrease 5
hc keybind XF86AudioMute spawn pamixer --toggle-mute
hc keybind XF86AudioPlay spawn mpc toggle
hc keybind XF86AudioNext spawn mpc next
hc keybind XF86AudioPrev spawn mpc prev
hc keybind XF86PowerOff spawn $HOME/bin/pwr
hc keybind XF86Sleep spawn $HOME/bin/pwr

# tags
TAG_KEYS=( {1..3} q w )

hc rename default "${TAG_NAMES[0]}" || true
for i in ${!TAG_NAMES[@]} ; do
  key="${TAG_KEYS[$i]}"
  hc add ${TAG_NAMES[$i]}
  hc keybind $Mod-$key use_index $i
  hc keybind $Mod-Shift-$key move_index $i
done
#hc keybind $Mod-w chain : add web  : use web : spawn firefox : \
#  keybind $Mod-w use web : keybind $Mod-Shift-w move web

# layouting
hc keybind $Mod-r remove , keybind $Mod-space cycle_layout 1
hc keybind $Mod-u split vertical 0.5
hc keybind $Mod-o split horizontal 0.5
  # split more precisely
for i in {1..9}; do
  hc keybind $Mod-Alt-$i split horizontal 0.$i
  hc keybind $Mod-Control-$i split vertical 0.$i
done
hc keybind $Mod-space cycle_layout 1
hc keybind $Mod-space cycle_layout -1
hc keybind $Mod-s floating toggle
hc keybind $Mod-f fullscreen toggle
hc keybind $Mod-p pseudotile toggle

  # some keybindings like to change on different hooks.
herbstclient -i|
while read line;do
  case $line in
    # remove the gap
    nogap ) herbstclient chain : set frame_gap -1 : \
      set window_gap ${window_gap--1} : keybind $Mod-0 emit_hook "
      set frame_border_width ${frame_border_width-1} :\
      pad 0 $pad :\
      pad 1 $pad ;;
    # add the gap
    gap* ) gap=${line/gap /};aPad=( $pad )
      for (( i=0;  i < ${#aPad[@]}; i++));do
        aPad[$i]=$(( ${aPad[$i]} + $gap - 1))
      done
      herbstclient chain : \
      set frame_gap "-$gap" :\
      set window_gap $gap : set frame_border_width 0 :\
      pad 0 ${aPad[@]} :\
      pad 1 ${aPad[@]} :\
      keybind $Mod-0 emit_hook nogap ;;
    expand) herbstclient $expand ;;
    contract) herbstclient $contract;;
  esac
done&

# switch to different layouts directly
hc keybind $Mod-Alt-m set_layout max
hc keybind $Mod-Alt-g set_layout grid
# "Master Window" type stuff
hc keybind $Mod-semicolon spawn $conf/swap auto
hc keybind $Mod-slash spawn $conf/swap close

hc keybind $Mod-Control-h resize left +$RESIZESTEP
hc keybind $Mod-Control-j resize down +$RESIZESTEP
hc keybind $Mod-Control-k resize up +$RESIZESTEP
hc keybind $Mod-Control-l resize right +$RESIZESTEP

# mouse
hc mouseunbind --all
hc mousebind $Mod-Button1 move
hc mousebind $Mod-Button2 zoom
hc mousebind $Mod-Button3 resize

# focus
hc keybind $Mod-BackSpace   cycle_monitor
hc keybind $Mod-Tab         cycle_all +1
hc keybind $Mod-Shift-Tab   cycle_all -1
hc keybind $Mod-c cycle
hc keybind $Mod-h focus left
hc keybind $Mod-j focus down
hc keybind $Mod-k focus up
hc keybind $Mod-l focus right
hc keybind $Mod-i jumpto urgent
hc keybind $Mod-Shift-h shift left
hc keybind $Mod-Shift-j shift down
hc keybind $Mod-Shift-k shift up
hc keybind $Mod-Shift-l shift right
hc keybind $Mod-m toggle focus_follows_mouse
src/dotfiles/config-herbstluftwm/autostart              2,0-1          Top
```

```
#!/bin/bash
# make a thumbnail for an image

pic="$1"
[ -n "$2" ] && scale="$2" || scale=250
name=`echo "$1"|cut -d'.' -f1`

convert -scale $scale "$pic" "$name"-thumb.png

exit

bin/mkthumb                                             10,2          All
```

```
# vim: filetype=sh
# git aliases
alias g='git'

push-configs() {
(
builtin cd $DOT
git add '*'
git commit "$@"
git push origin master
)
}
alias s='sudo'

# vim aliases
alias zshrc="$EDITOR $HOME/.zshrc"
alias vimrc="$EDITOR $HOME/.vimrc"
alias al="$EDITOR /home/ninjaaron/.aliases"
alias book="$EDITOR ~/docs/my-work/heb/book.txt"
alias hla="$EDITOR ~/.config/herbstluftwm/autostart"
alias hlf="$EDITOR ~/.config/herbstluftwm/func"
alias hls="$EDITOR ~/.config/herbstluftwm/swap"

# packagmanagment
alias pS="sudo pacman -S"
alias pSs="pacman -Ss"
alias pSyu="sudo pacman -Syu"
alias pRs="sudo pacman -Rs"
alias pU="sudo pacman -U"
alias y="yaourt"
alias ySyua="yaourt -Syua"

alias agi='s apt-get install'
alias agp='s apt-get purge'
alias agu='s apt-get update'
alias agdu='s apt-get dist-upgrade'
alias agac='s apt-get autoclean'
alias agar='s apt-get autoremove'
alias aar='s add-apt-repository'

alias grub-update="sudo grub-mkconfig -o /boot/grub/grub.cfg"

# power
alias reboot="sudo shutdown -r now"
alias halt="sudo shutdown -h now"
alias sus="sudo pm-suspend"

# misc command shortcuts
alias ls="ls --color=auto -F"
alias lsl="ls -lh"
alias lsa="ls -a"
alias lsla="ls -lha"
alias df="df -h"
alias du="du -h"
alias grep='grep --color=auto'
alias f='find -L -name'
alias wh="while"
alias ast="autosus toggle"
alias mnt='s mount'

alias tma='tmux attach -d'  #opens a window with the last tmux ses
alias w3g='w3m -cookie google.com'
alias png="ping google.com"
alias alsi="alsi -a"
alias td="transmission-daemon"
alias ta="transmission-remote -a"
alias tl="transmission-remote -l"

alias dh="diatheke -b OSMHB -k"

alias nh="sudo nethogs eth1"

gap() { herbstclient emit_hook "gap $@";}

# find processes that match a regex
psg() { ps aux|grep "$@"|grep -v grep; }

e() {
  [[ -n $@  ]] && [[ -e $@ ]] && owner=$(ls -l $@|cut -d' ' -f3) &
    [[ $owner != $USER ]] || [[ ${PWD/$HOME/} = $PWD ]] &&
   #echo -n "sudo [Y/n]? " && read sudo && [[ ${sudo:-y} = y ]] &&
    sudo $EDITOR -c 'syntax off' $@ || $EDITOR $@
}

# `cd` gets you where you're going!
cd() {
  [ -z "$1" ] && builtin cd ~ || builtin cd "$1"  2> /dev/null ||
    builtin cd ~/"$1" 2> /dev/null || {
    [[ $PWD != "/" ]]  &&
      found=( $(find -L . -type d -name "$1" 2> /dev/null) ) ||
      found=( $(find . -type d -name "$1" 2> /dev/null) )
    if [ -n "$found" ]; then
      cdlist
    else
      found=( $(find -L ~ -type d -name "$1") )
      cdlist
    fi
  }
}

cdlist() {
  if [ ${#found} -gt 1 ];then
    for ((i=1; i <= ${#found[@]}; i++));do
      echo "${i}. ${found[$i]}"
    done
    echo -n "which? "
    read sel
    builtin cd "${found[$sel]}"
  else
    builtin cd "$found"
  fi
}

# resets the size of the framebuffer for the tty
fbg() {
  [ -z "$@" ] && fbset -g 1600 900 1600 900 32 || fbset -g "$@" "$
}

# opens a tmux session with an optional command
tm() { tmux new-session "$@" ;}

## looks up a word using the local dictionaries
#dl() { dict "$*"|vimpager; clear ;}
## looks up a word using remote dictionaries
#dr() { dict -h dict.org "$*"|vimpager; clear; }

src/dotfiles/aliases                                    78,14         All
```

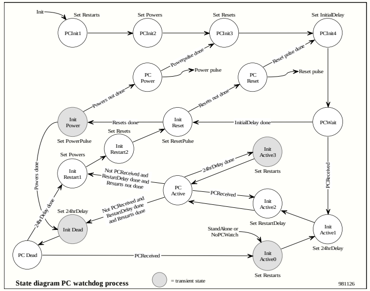
```
#!/bin/bash

# this script started it's life as dmenu_run

# get user configs
source $HOME/.dmenurc
DMENU=${DMENU-dmenu}

PATH="$PATH:$HOME/bin"
cachedir=${XDG_CACHE_HOME:-"$HOME/.cache"}
if [ -d "$cachedir" ]; then
  cache=$cachedir/dmenu_run
else
  cache=$HOME/.dmenu_cache # if no xdg dir, fall back to dotfile
fi

cmd=$(
(
  IFS=:
  if stest -dqr -n "$cache" $PATH; then
    echo -e ".\n$(stest -flx $PATH | sort -u)\n$(ls)"| tee "$cach
  else
    cat "$cache"
  fi
)|$DMENU)

# if the user selects a directory, go there and execute brws.
echo $cmd
[[ -d $cmd && $cmd != . ]] && cd "$cmd" && cmd="brws"

case "$cmd" in
  . ) # fm makes a list of files in all subdirectors and opens th
    fm &
    ;;
  theme ) # changes the color scheme
    theme=$(ls $HOME/.themes|sed '/#d/d ; /~$/d ; /xres/d'|\
      $DMENU -p 'themes:')
    theme $theme&
    ;;
  mv|cp|"ln -s" ) # helpers for selecting files
    sour=$(find -L . |$DMENU -p "$cmd SOURCE:")
    [[ -z $sour ]] && exit
    dest=$(find -L $HOME -type d|$DMENU -p "$cmd $sour DEST:")
    [[ -z $dest ]] && exit
    $cmd "$sour" "$dest"&
    ;; # launch interactive programs in a terminal.
  man )
    entry=$(
    (IFS=':'
    find $(manpath))|
    sed -n 's/.*\/\(.*\).1.gz/\1/p'|sort|
    $DMENU -p "man:")
    urxvtc -e $SHELL -c "man $entry"
    ;;
  youtube-viewer*|torrtux*|vifm*|vim*|vi*|ed*|nano*|ncmpcpp|cmatr
    wicd-curses|info*|weechat-curses|htop|yaourt*|"man "* )
    urxvt -e $SHELL -c "$cmd"
    ;;
  *)
    echo "$cmd"|${SHELL:-"/bin/sh"} &
esac

exit
bin/dm                                                  58,6          All
```

```
~/.vimrc
 environment
et nocompatible backspace=indent,eol,start history=50 nowrap mou
et ruler showcmd incsearch lbr expandtab tabstop=2 shiftwidth=2
et nu cursorline wildmenu wildmode=list:longest,full
et g:SuperTabDefaultCompletionType = "context"
yntax on
iletype plugin indent on
et gfn=Ubuntu\ Mono\ 10 go=

f has("gui_running") | colorscheme molokai
lse | colorscheme cterm
ndif
 autocmd stuff
ugroup vimrcEx
 au!
 autocmd BufRead /tmp/pentadacty*.txt
       \ source /home/ninjaaron/.vim/pentadactyl.vim
 autocmd BufRead /tmp/uzbl_edit* source /home/ninjaaron/.vim/pen
 autocmd BufReadPost *
       \ if line("'\"") > 1 && line("'\"") <= line("$") |  exe "
 autocmd vimenter * if !argc() | NERDTree | endif
 autocmd bufenter * if (winnr("$") == 1 && exists("b:NERDTreeTyp
ugroup END

 maps and commands and so forth
apclear | mapclear!
ommand Va VoomToggle asciidoc
oremap <C-N> :NERDTreeToggle<CR>
ommand DiffOrig vert new | set bt=nofile | r ++edit # | 0d_ | di
      \ | wincmd p | diffthis

 syntax toggling
unction! ToggleSyntax()
 if exists("g:syntax_on") | syntax off
 else | syntax enable | endif
ndfunction
 hebrew mode toggle
unction! ToggleHebrew()
 if &l:rightleft == 0 | set rightleft keymap=hebrew
 else | set norightleft keymap= | endif
ndfunction
 create bindings for insert and normal mode together.
unction! DualBind(key,command)
 exe "noremap <Leader>".a:key a:command
 exe "inoremap <Leader>".a:key "^O".a:command
ndfunction
 keymaps
et mapleader=";"
 settings
noremap <silent> <Leader>sy :call ToggleSyntax()<CR>
call DualBind( 'i', ':call ToggleHebrew()<cr>' )
call DualBind( 'r', 'mvgq}`v' )
let settings = [['fi', 'foldmethod=indent'], ['tw', 'tw=72'], ['W
for [ key, setting ] in settings
  exe "call DualBind('".key."', ':set" setting."<cr>')" | endfor
" saving and quitting
for key in ['w', 'q', 'x']
  exe "call DualBind('".key."', ':".key."<CR>')" | endfor
" filetypes
for [ key, ft ] in [['a', 'asciidoc'], ['p', 'python'], ['s', 'sh
  exe "noremap <Leader>f".key " :set filetype=".ft "<CR>"|endfor
" Window management
for key in [ "h", "j", "k", "l", "H", "J", "K", "L" ]
  exe "call DualBind( '".key."', '^W".key."' )" | endfor
" get ya some output from shell commands
nnoremap <Leader>sh :.!sh<CR>
vnoremap <Leader>sh dmvo<Esc>p:.!sh<CR>0d$`vPjdd`v
nnoremap <Leader>py :.!python<CR>
vnoremap <Leader>py :!python<CR>
" pasting
vnoremap <Leader>c "+y
inoremap <Leader>v ^O"+P
call DualBind( 'P', '"*p' )
inoremap <Leader>p ^Op
delfunction DualBind

All src/dotfiles/vimrc                                  1,1
```

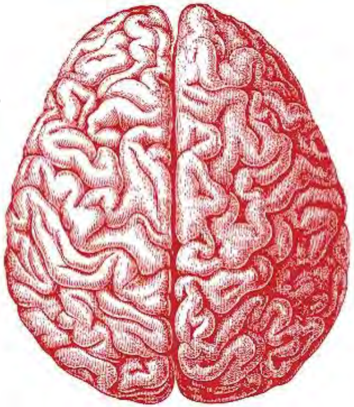
```

discharge() { acpi|grep Discharging ;}
level() { acpi|sed 's/.*ing, \(.*\)%.*/\1/' ;}
psg() { ps aux|grep "$@"|grep -v grep; }

sleepless="$HOME/sleepless"

toggle() {
  [ -a $sleepless ]  && rm $sleepless || touch $sleepless
}

idle() {
  [ -n "$(discharge)" ] && [ -z "$(psg mplayer)" ] && [ !
    sudo pm-suspend-hybrid
}

low() {
  while true ; do
    while [ $(level) -gt 5 ] || [ -z "$(discharge)" ]; do
      sleep 60
    done
    zenity --warning --text "Battery very low."
    sleep 10m
  done
}

$@

bin/autosus                                             3,0-1         Bot
```

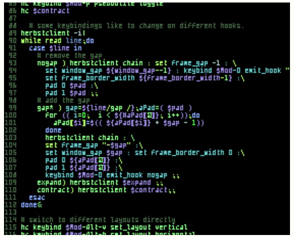
```
#!/bin/bash

# file browsing with dmenu

source $HOME/.dmenurc
DMENU="${DMENU-dmenu} -l 60"

# this is actually the last function called. Customize be
# extntion here.
open() {
  case "$sel" in
    *.mkv|*.mov|*.avi*|*.flv*|*.ogv|*.mp4*|*.m4v|*.mp3|*.
      mplayer "$sel" &
      ;;
    *.pdf|*.ps|*.djvu )
      zathura "$sel" &
      ;;
    *.jpg|*.gif|*.png )
      gqview "$sel" &
      ;;
    *.xfc )
      gimp "$sel" &
      ;;
    *.odt|*.odp|*.odg|*.ods|*.doc )
      libreoffice "$sel" &
      ;;
    *.html|*.htm )
      firefox "$sel" &
      ;;
    *) # with undefined extentions, open vim. If the dire
       # somewhere in $HOME, ask for sudo.
      if  [[ ${PWD/*$USER*/$USER} != $USER ]] &&
      [[ $(echo -e "y\nn"|$DMENU -p "sudo?") = y ]] ; the
        urxvtc -e sh -c "sudo vim -u $HOME/.vimrc '$PWD/$
        exit
      else
        urxvtc -e sh -c "vim '$PWD/$sel'"
      fi
  esac
}

# use `find` to get the entire contense of $HOME recursiv
search() {

  # really long subshell supplies the file list and hides
  # files. In $HIME, hide hidden files by defautl unless
  # selects 'unhide.'
  menu=$(IFS="
  "
  if [[ ${PWD/\/home*/\/home} = /home ]];then

    if [[ $show = hidden ]]; then
      find -L .|
      sed '/~$/d ; /Ubuntu One/{ H; d; }; $ { p; x; }'
      echo "hide"
    else
      find -L . |
      sed "/\/\./d ; /ulconf/d ; /~$/d ; /Ubuntu One/{ H;
      echo "unhide"
    fi

  else
    find .|sed '/~$/d'
  fi)

  sel=$(echo "$menu"|$DMENU -p "search $PWD:")
  case $sel in
    "" )
      exit ;;
    hide )
      show=normal
      search ;;
    unhide )
      show=hidden
      search ;;
    *)
      browse "$@"
  esac
}

# browse files one directory at a time. the only way out
# into browse(). However, when $sel is a file, browse() h
# off to open(). This is a result of how the program orig
# I've thought about changing to go directly to open() fr
# but I spent my time writing this sentence instead.
browse() {
  while [ -d "$sel" ]; do
    cd "$sel"
    sel=$(ls -a|$DMENU -p "$PWD:")
    [[ $sel = . ]] && break
  done

  case $sel in
    "." )
      search ;;
    "" )
      exit 0 ;;
    *)
      open&
  esac
}

# the search() is the default behavior, but I also have t
# aliased to `brws`, so when invoced that way, it runs br
if [[ $0 = *brws ]];
then sel="$PWD" && browse
else search
fi

exit

All bin/fm                                              1,1           All
```

```
axelrc
bash.bash_logout
bash.bashrc
bash_completion.d
bdfedit.rc
binfmt.d/
ca-certificates/
ca-certificates.c
colord.conf
conf.d/
conky/
cron.d/
cron.daily/
cron.deny
cron.hourly/
cron.monthly/
cron.weekly/
crypttab
cups/
dbus-1/
default/
depmod.d/
dhcpcd.conf
drirc
environment
fb.modes
/etc> cd /usr/shar
aclocal/
aclocal-1.13/
alsa/
application-regis
applications/
apr-1/
autoconf/
automake-1.13/
avahi/
awk/
bash-completion/
bdfedit/
bibletime/
bison/
ca-certificates/
ca-certificates-j
calc/
cmake/
cmake-2.8/
color/
colord/
common-lisp/
cracklib/
cups/
dblatex/
dbus-1/
dconf-editor/
desktop-directori
devhelp/
dict/
diagones/
djvu/
doc/
dstat/
dtds/
emacs/
emesene/
enchant/
et/
FBReader/
ffmpeg/
figlet/
file/
fonts/
gcalctool/
gcc-4.7.2/
GConf/
gdb/
gdm/
gedit/
geoclue-providers
getopt/
gettext/
ghostscript/
gimp/
gir-1.0/
git/
/usr/share> cd bi
alarm*     backup*
aur*       baracti
autosus*   bcl*
~/bin> cd
bin@  doc/  mnt/
~> cat src/dotfil
default
partial alphanumer
xkb_symbols "ano"

  include "us(bas
    name[Group1]=
  key.type[Group1

  // wadova
  key <LSGT> { [
  key <TLDE> { [
  key <AE01> { [
  key <AE02> { [
  key <AE03> { [
  key <AE04> { [
  key <AE05> { [
  key <AE06> { [
  key <AE07> { [
  key <AE08> { [
  key <AE09> { [
  key <AE10> { [
  key <AE11> { [
  key <AE12> { [

  key <AD01> { [
  key <AD02> { [
sigma>
  key <AD03> { [
  key <AD04> { [
  key <AD05> { [
  key <AD06> { [
  key <AD07> { [
  key <AD08> { [
  key <AD09> { [
  key <AD10> { [
>
  key <AD11> { [
  key <AD12> { [
>
  key <BKSL> { [

  key <AC01> { [
  key <AC02> { [
  key <AC03> { [
  key <AC04> { [
  key <AC05> { [
  key <AC06> { [
  key <AC07> { [
  key <AC08> { [
  //
  key <AC09> { [
  key <AC10> { [
  key <AC11> { [

  key <AB01> { [
  key <AB02> { [
  key <AB03> { [
  key <AB04> { [
  key <AB05> { [
  key <AB06> { [
  key <AB07> { [
  key <AB08> { [
  key <AB09> { [
  key <AB10> { [

  key <SPCE> { [

  include "level3
};
~> 

  1  [||||
  2  [||
  3  [||
  4  [||
  Mem[||||||||||
  Swp[|

Playlist (8 items
Time    Artist
4:36    Christina
==========>
Playing: Christin

~> 
```

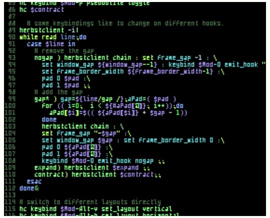


State diagram PC watchdog process

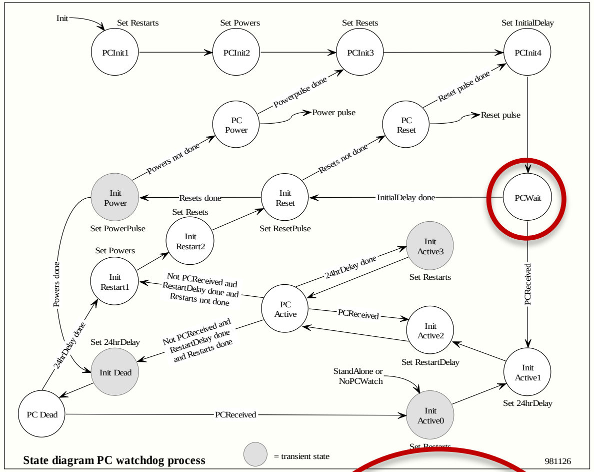


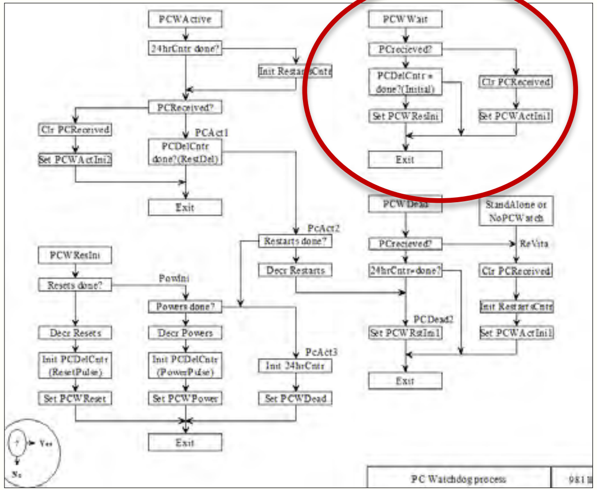


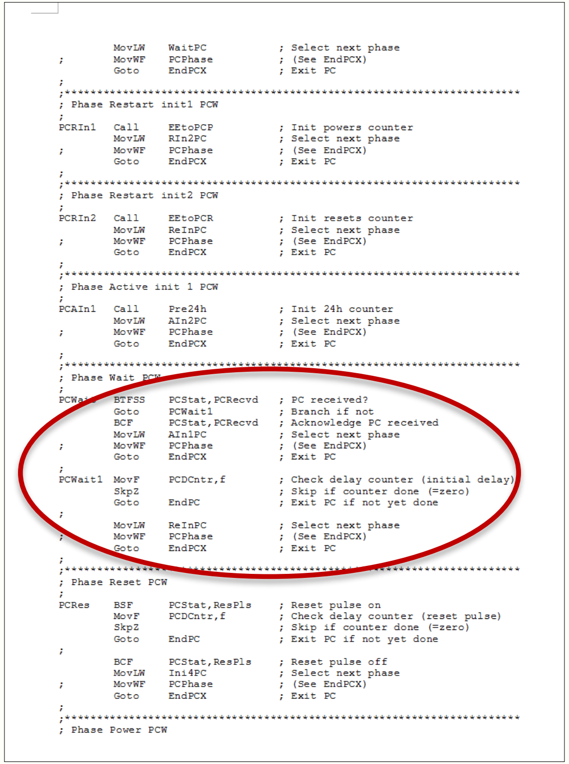
```
hc keybind $Mod-p pseudotile toggle
hc $contract

  # some keybindings like to change on different hooks.
herbstclient -il
while read line;do
  case $line in
    # remove the gap
    nogap ) herbstclient chain : set frame_gap -1 : \
      set window_gap ${window_gap--1} : keybind $Mod-O emit_hook "
      set frame_border_width ${frame_border_width-1} :\
      pad 0 $pad :\
      pad 1 $pad ;;
    # add the gap
    gap* ) gap=${line/gap /};aPad=( $pad )
      for (( i=0;  i < ${#aPad[@]}; i++));do
        aPad[$i]=$(( ${aPad[$i]} + $gap - 1))
      done
      herbstclient chain : \
      set frame_gap "-$gap" :\
      set window_gap $gap : set frame_border_width 0 :\
      pad 0 ${aPad[0]} :\
      pad 1 ${aPad[0]} :\
      keybind $Mod-O emit_hook nogap ;;
    expand) herbstclient $expand ;;
    contract) herbstclient $contract;;
  esac
done&

# switch to different layouts directly
hc keybind $Mod-Alt-v set_layout vertical
hc keybind $Mod-Alt-h set_layout horizontal
```



```
hc keybind $Mod-p pseudotile toggle
hc $contract

  # some keybindings like to change on different hooks.
herbstclient -il
while read line;do
  case $line in
    # remove the gap
    nogap ) herbstclient chain : set frame_gap -1 : \
      set window_gap ${window_gap--1} : keybind $Mod-O emit_hook "
      set frame_border_width ${frame_border_width-1} :\
      pad 0 $pad :\
      pad 1 $pad ;;
    # add the gap
    gap* ) gap=${line/gap /};aPad=( $pad )
      for (( i=0;  i < ${#aPad[@]}; i++));do
        aPad[$i]=$(( ${aPad[$i]} + $gap - 1))
      done
      herbstclient chain : \
      set frame_gap "-$gap" :\
      set window_gap $gap : set frame_border_width 0 :\
      pad 0 ${aPad[0]} :\
      pad 1 ${aPad[0]} :\
      keybind $Mod-O emit_hook nogap ;;
    expand) herbstclient $expand ;;
    contract) herbstclient $contract;;
  esac
done&

# switch to different layouts directly
hc keybind $Mod-Alt-v set_layout vertical
hc keybind $Mod-Alt-h set_layout horizontal
```







```
          MovLW    WaitPC             ; Select next phase
;         MovWF    PCPhase            ; (See EndPCX)
          Goto     EndPCX             ; Exit PC
;
;**********************************************************************
; Phase Restart init1 PCW
;
PCRIn1    Call     EEtoPCP            ; Init powers counter
          MovLW    RIn2PC             ; Select next phase
;         MovWF    PCPhase            ; (See EndPCX)
          Goto     EndPCX             ; Exit PC
;
;**********************************************************************
; Phase Restart init2 PCW
;
PCRIn2    Call     EEtoPCR            ; Init resets counter
          MovLW    ReInPC             ; Select next phase
;         MovWF    PCPhase            ; (See EndPCX)
          Goto     EndPCX             ; Exit PC
;
;**********************************************************************
; Phase Active init 1 PCW
;
PCAIn1    Call     Pre24h             ; Init 24h counter
          MovLW    AIn2PC             ; Select next phase
;         MovWF    PCPhase            ; (See EndPCX)
          Goto     EndPCX             ; Exit PC
;
;**********************************************************************
; Phase Wait PCW
;
PCWait    BTFSS    PCStat,PCRecvd     ; PC received?
          Goto     PCWait1            ; Branch if not
          BCF      PCStat,PCRecvd     ; Acknowledge PC received
          MovLW    AIn1PC             ; Select next phase
;         MovWF    PCPhase            ; (See EndPCX)
          Goto     EndPCX             ; Exit PC
;
PCWait1   MovF     PCDCntr,f          ; Check delay counter (initial delay)
          SkpZ                        ; Skip if counter done (=zero)
          Goto     EndPC              ; Exit PC if not yet done
;
          MovLW    ReInPC             ; Select next phase
;         MovWF    PCPhase            ; (See EndPCX)
          Goto     EndPCX             ; Exit PC
;
;**********************************************************************
; Phase Reset PCW
;
PCRes     BSF      PCStat,ResPls      ; Reset pulse on
          MovF     PCDCntr,f          ; Check delay counter (reset pulse)
          SkpZ                        ; Skip if counter done (=zero)
          Goto     EndPC              ; Exit PC if not yet done
;
          BCF      PCStat,ResPls      ; Reset pulse off
          MovLW    Ini4PC             ; Select next phase
;         MovWF    PCPhase            ; (See EndPCX)
          Goto     EndPCX             ; Exit PC
;
;**********************************************************************
; Phase Power PCW
```

Case: Scrum Sprint Planning

- What is the measure of success for the coming sprint ?
- “What a strange question !
  We're Agile, so we deliver working software. Don't you know ?”

- Note: Users are not waiting for software:
  they just need *improved performance* of what they’re doing

- How about a requirement for 'Demo': *No Questions - No Issues*
- How's that possible !!?

- They actually succeeded !

Demo ??

- Give the delivery to the stakeholders
- Zip your mouth

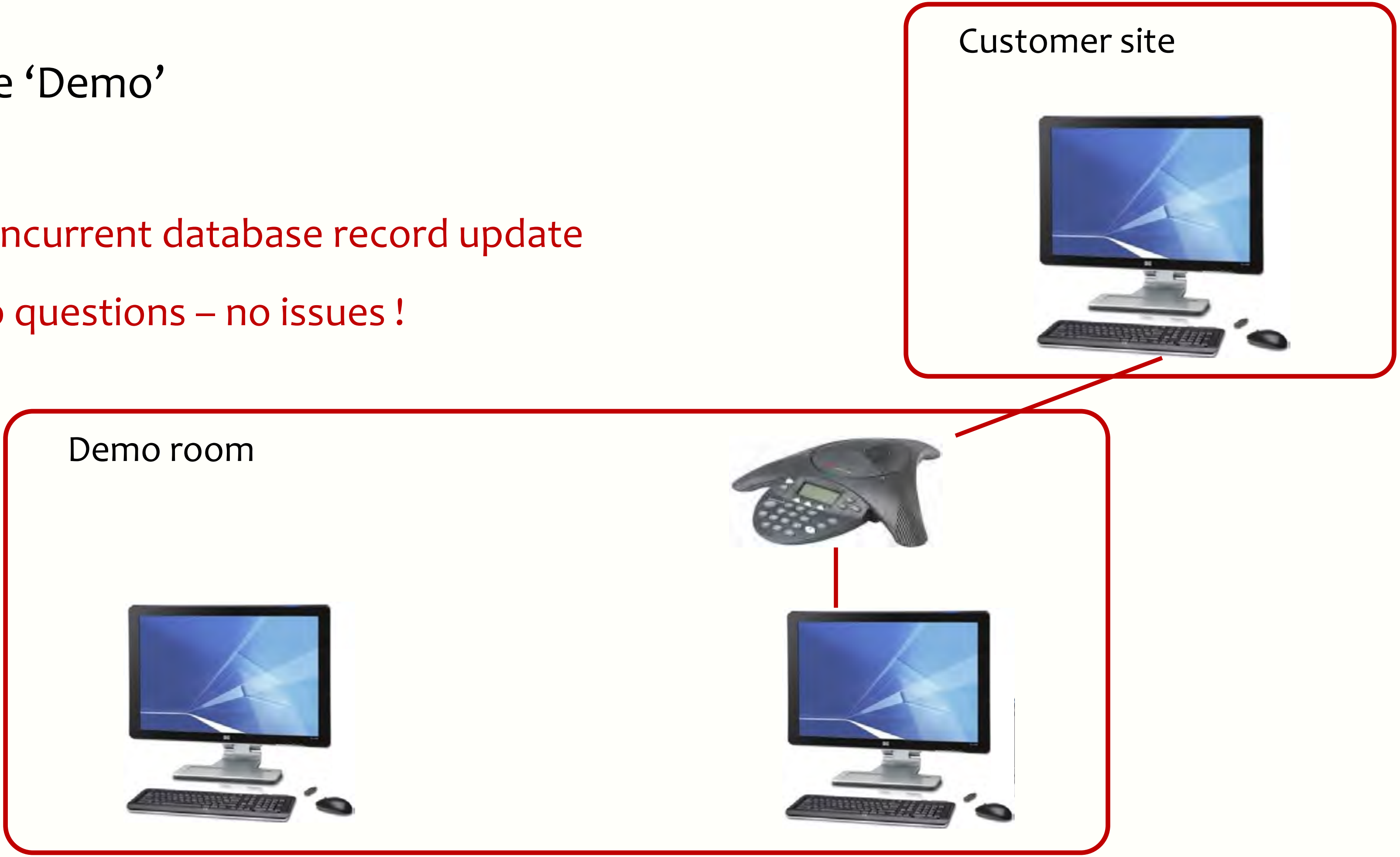
- Keep your hands handcuffed on your back
- and o-b-s-e-r-v-e what happens
- Seeing what the stakeholders actually do, provides real feedback
- Then we can 'talk business' with the stakeholders

- Is this what you do ?

The ‘Demo’

Concurrent database record update

No questions – no issues !



Customer site

Demo room

Delivery Strategy Suggestions

- What we deliver will be used by the appropriate users immediately, within one week not making them less efficient than before
- If a delivery isn't used immediately, we analyse and close the gap so that it will start being used (otherwise we don't get feedback)
- The proof of the pudding is when it's eaten and found tasty, by them, not by us
- The users determine success, and whether they want to pay (we don't have to tell them, but it should be our attitude)
- Would you dare to deliver no-cure-no-pay ?

Case: How much legwork is being done in your project ?

- Requirements/specifications were trashed out with product management
- Technical analysis was done and
- Detail design for the first delivery

- At the first delivery:
- James: How is the delivery? (quality versus expectation)
- Adrian: It's exactly as expected,
  which is absolutely unprecedented for a first delivery
  the initial legwork has really paid off

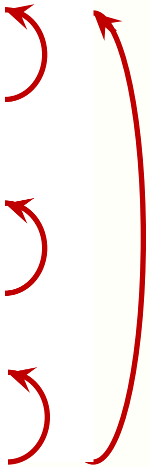
Some techniques shown

- Design
- Drawings
- DesignLog
- Review
- No Questions - No Issues

A Zero Defects attitude makes an immediate difference

Basic approach

- Design the requirement
- Review
- Design implementation
- Review
- Implement (code ?)
- Review



Iterate fast, as needed

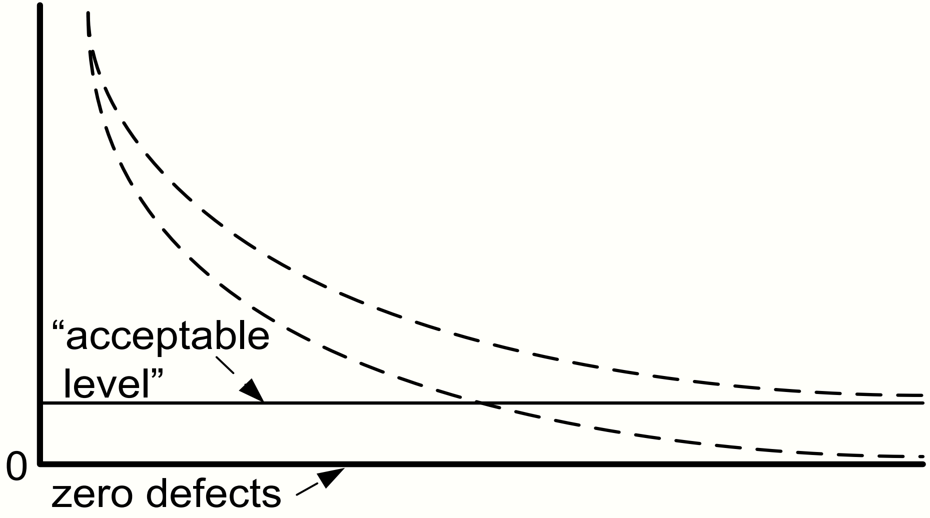
- Test doesn’t find issues (because they’re not there)

Do we deliver Zero Defect products ?

- How many defects do you think are acceptable ?
- Do the requirements specify a certain number of defects ?
- Do you check that the required number has been produced ?

- In your projects
- How much time is spent putting defects in ?
- How much time is spent trying to find and fix them ?
- Do you sometimes get repeated issues ?
- How much time is spent on defect prevention ?
- Could you use "No Questions - No Issues" ?

Better quality costs less



Approaching Zero Defects is Absolutely Possible

If in doubt, let's talk about it

Niels Malotaux

Copyright © 2022 by Niels Malotaux niels@malotaux.eu www.malotaux.eu/?id=zerodefects

More

Inquiries: niels@malotaux.eu

www.malotaux.eu/booklets

1 Evolutionary Project Management Methods (2001)
Issues to solve, and first experience with the Evo Planning approach

2 How Quality is Assured by Evolutionary Methods (2004)
After a lot more experience: rather mature Evo Planning process

3 Optimizing the Contribution of Testing to Project Success (2005)
How Testing fits in

3a Optimizing Quality Assurance for Better Results (2005)
Same as Booklet 3, but for non-software projects

4 Controlling Project Risk by Design (2006)
How the Evo approach solves Risk by Design (by process)

5 TimeLine: How to Get and Keep Control over Longer Periods of Time (2007)
Replaced by Booklet 7, except for the step-by-step TimeLine procedure

6 Human Behaviour in Projects (APCOSE 2008)
Human Behavioural aspects of Projects

7 How to Achieve the Most Important Requirement (2008)
Planning of longer periods of time, what to do if you don’t have enough time

8 Help ! We have a QA Problem ! (2009)
Use of TimeLine technique: How we solved a 6 month backlog in 9 weeks

9 Predictable Projects (2012) - How to deliver the Right Results at the Right Time

RS Measurable Value with Agile (Ryan Shriver - 2009)
Use of Evo Requirements and Prioritizing principles

www.malotaux.eu/inspections

Inspection pages