How to move towards Zero Defects

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

Niels Malotaux

N R Malotaux
Consultancy

niels@malotaux.nl

www.malotaux.nl

Niels Malotaux

- Independent Project and Organizational Coach
- Expert in helping optimizing performance

- Helping projects and organizations very quickly to become
  - More effective – doing the right things better
  - More efficient – doing the right things better in less time
  - Predictable – delivering as predicted
- Getting projects on track

Result Management

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

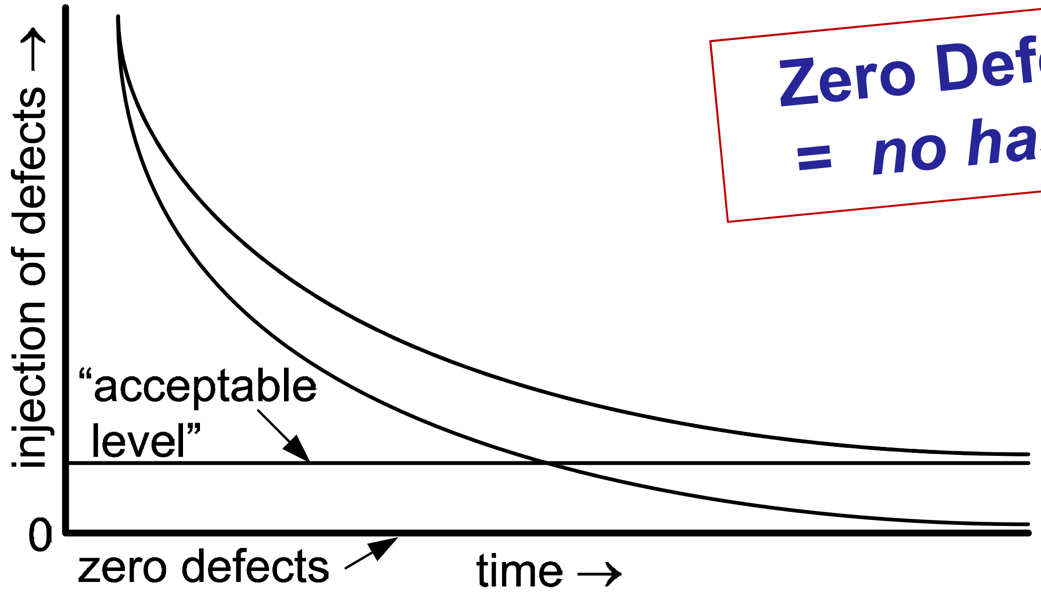
- How much time is spent putting defects in ?
- How much time is spent trying to find and fix them ?
- Do you sometimes get repeated issues ?
- How much time is spent on defect prevention ?

Root Cause Analysis

- Is Root Cause Analysis routinely performed ?
- What is the Root Cause of a defect ?
- Cause:
  The error that caused the defect
- Root Cause:
  What *caused us* to make the error that caused the defect
- Without proper RCA, we're *doomed to repeat the same errors*

What is Zero Defects

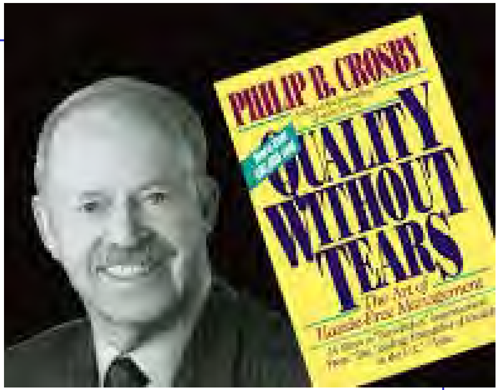
- Zero Defects is an asymptote



Zero Defects
= no hassle

- 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

Crosby (1926-2001) - Absolutes of Quality

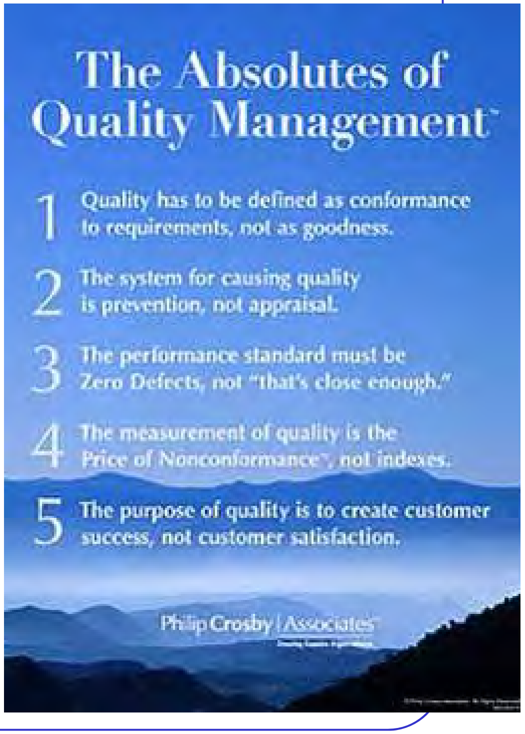


- Conformance to requirements
- Obtained through prevention
- Performance standard is zero defects
- Measured by the price of non-conformance (PONC)

Philip Crosby, 1970

- The purpose is customer success
  (not customer satisfaction)

Added by Philip Crosby Associates, 2004



The Absolutes of Quality Management™

1 Quality has to be defined as conformance to requirements, not as goodness.

2 The system for causing quality is prevention, not appraisal.

3 The performance standard must be Zero Defects, not "that's close enough."

4 The measurement of quality is the Price of Nonconformance™, not indexes.

5 The purpose of quality is to create customer success, not customer satisfaction.

Philip Crosby | Associates™

Ultimate Goal of a What We Do

Quality on Time

Delivering the Right Result at the Right Time,
wasting as little time as possible (= efficiently)

- Providing the customer with
  - what he needs
  - at the time he needs it
  - to be satisfied
  - to be more successful than he was without it
- Constrained by (win - win)
  - what the customer can afford
  - what we mutually beneficially and satisfactorily can deliver
  - in a reasonable period of time

Shouldn't Testing and QA find the defects ?

Deming

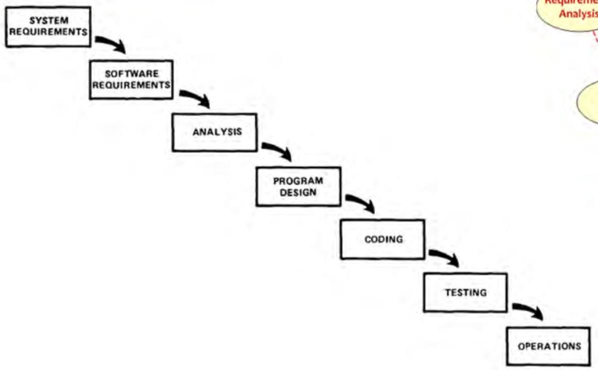
(1900-1993)

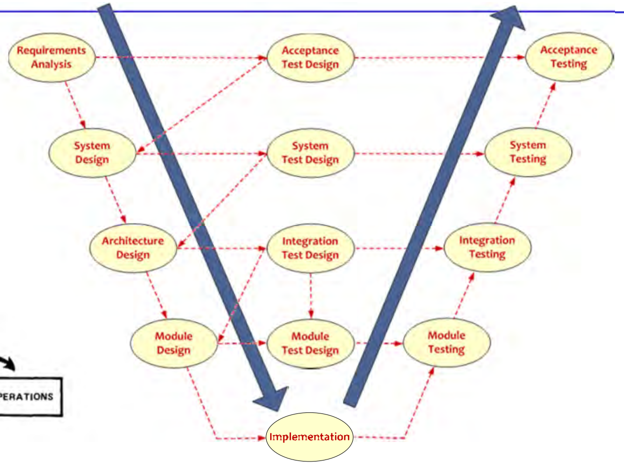
Deming:

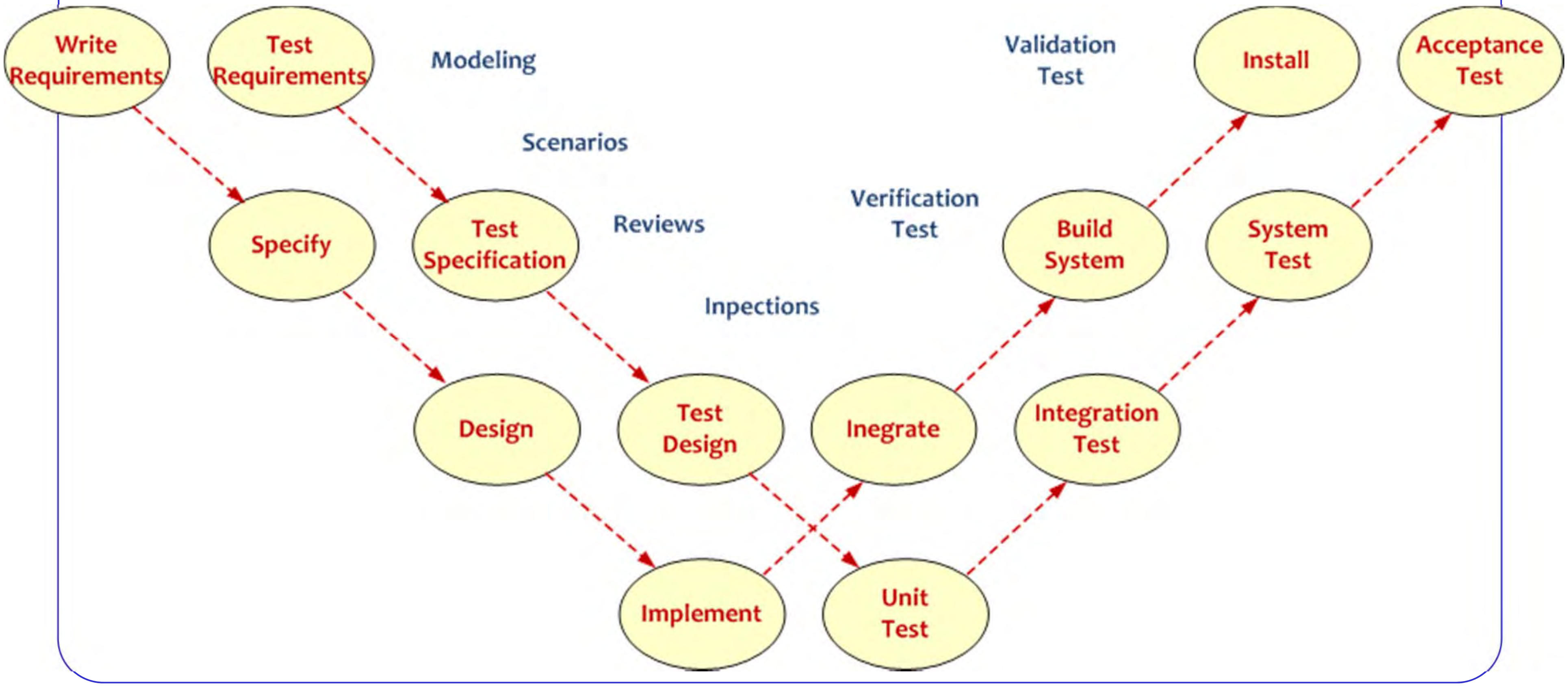
- Quality comes not from testing, but from *improvement of the development process*
- Testing does *not* improve quality, nor guarantee quality
- It’s too late
- The quality, good or bad, is already in the product
- You cannot test quality into a product

- Who delivers the quality, good or bad ?
- Testers and QA are *consultants* to development
- Testing and QA *shouldn’t delay* the delivery - How ?

W-model







Write Requirements

Test Requirements

Modeling

Validation Test

Install

Acceptance Test

Scenarios

Specify

Test Specification

Reviews

Verification Test

Build System

System Test

Inpections

Design

Test Design

Inegrate

Integration Test

Implement

Unit Test

Some Examples

Design techniques

- Design
- Review
- Code
- Review

Iterate as needed

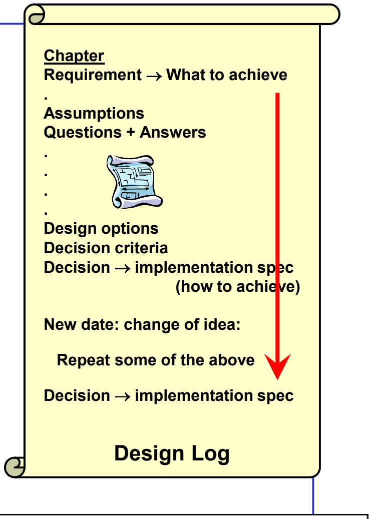
- Test (no questions, no issues)
- If issue in test: no Band-Aid: start all over again:
  Review: What's wrong with the design ?
- Reconstruct the design (if the design description is lacking)
- QA to review the DesignLog for more efficiently helping the developers: Ask "Can we see the DesignLog?"

Chapter
Requirement → What to achieve
.
Assumptions
Questions + Answers
.
.
.
.
Design options
Decision criteria
Decision → implementation spec
(how to achieve)

New date: change of idea:
Repeat some of the above
Decision → implementation spec

Design Log

In the pub



Chapter
Requirement → What to achieve
.
Assumptions
Questions + Answers
.
.
.
.
Design options
Decision criteria
Decision → implementation spec
(how to achieve)

New date: change of idea:

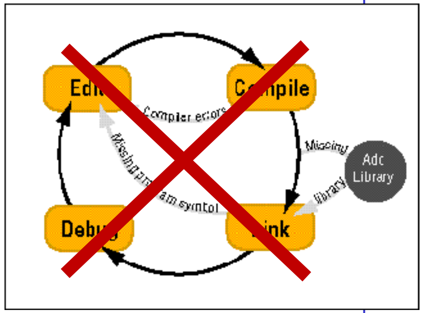
Repeat some of the above

Decision → implementation spec

Design Log

James:

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

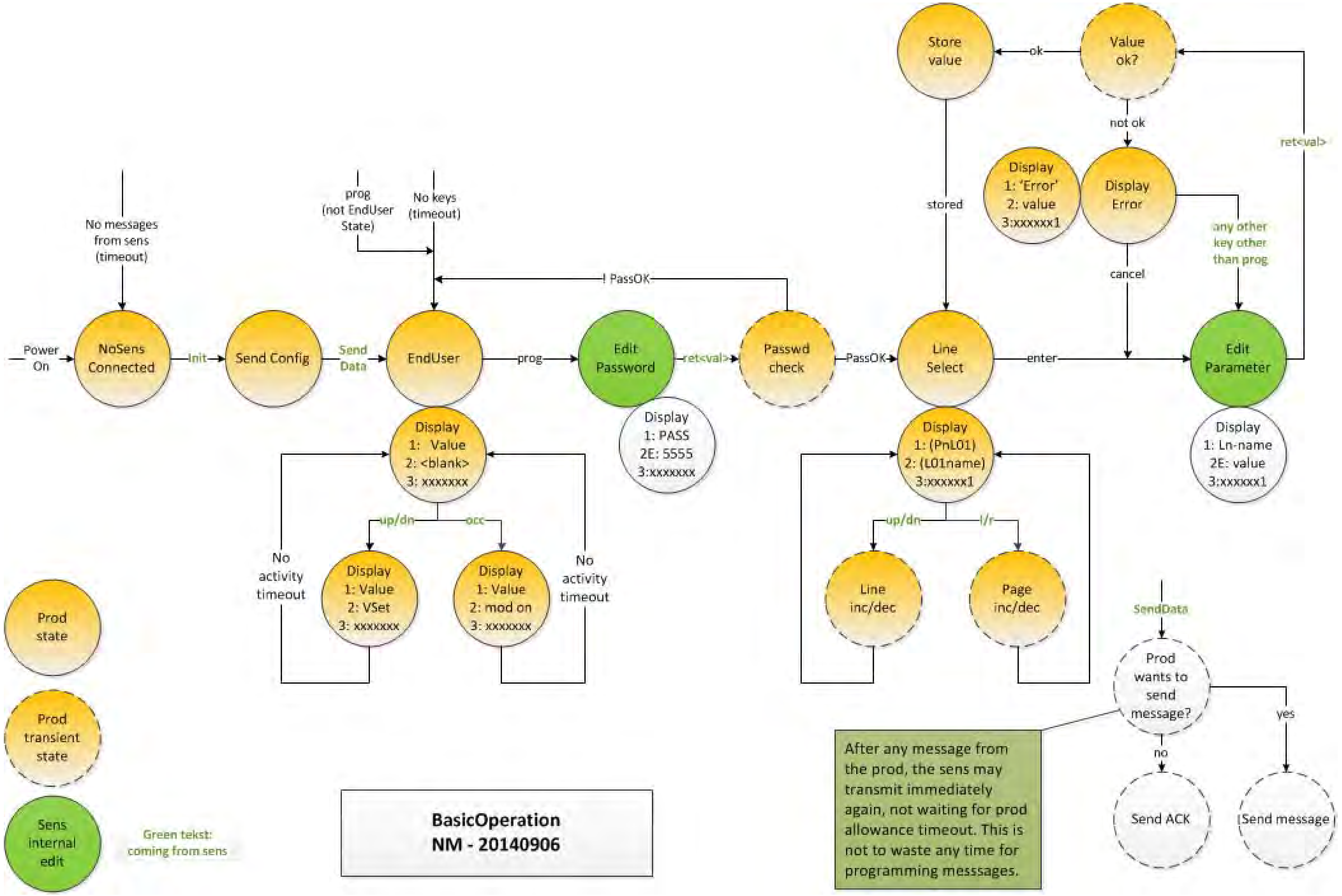


Susan:

- *We had only 1.5 week 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!*

Design example

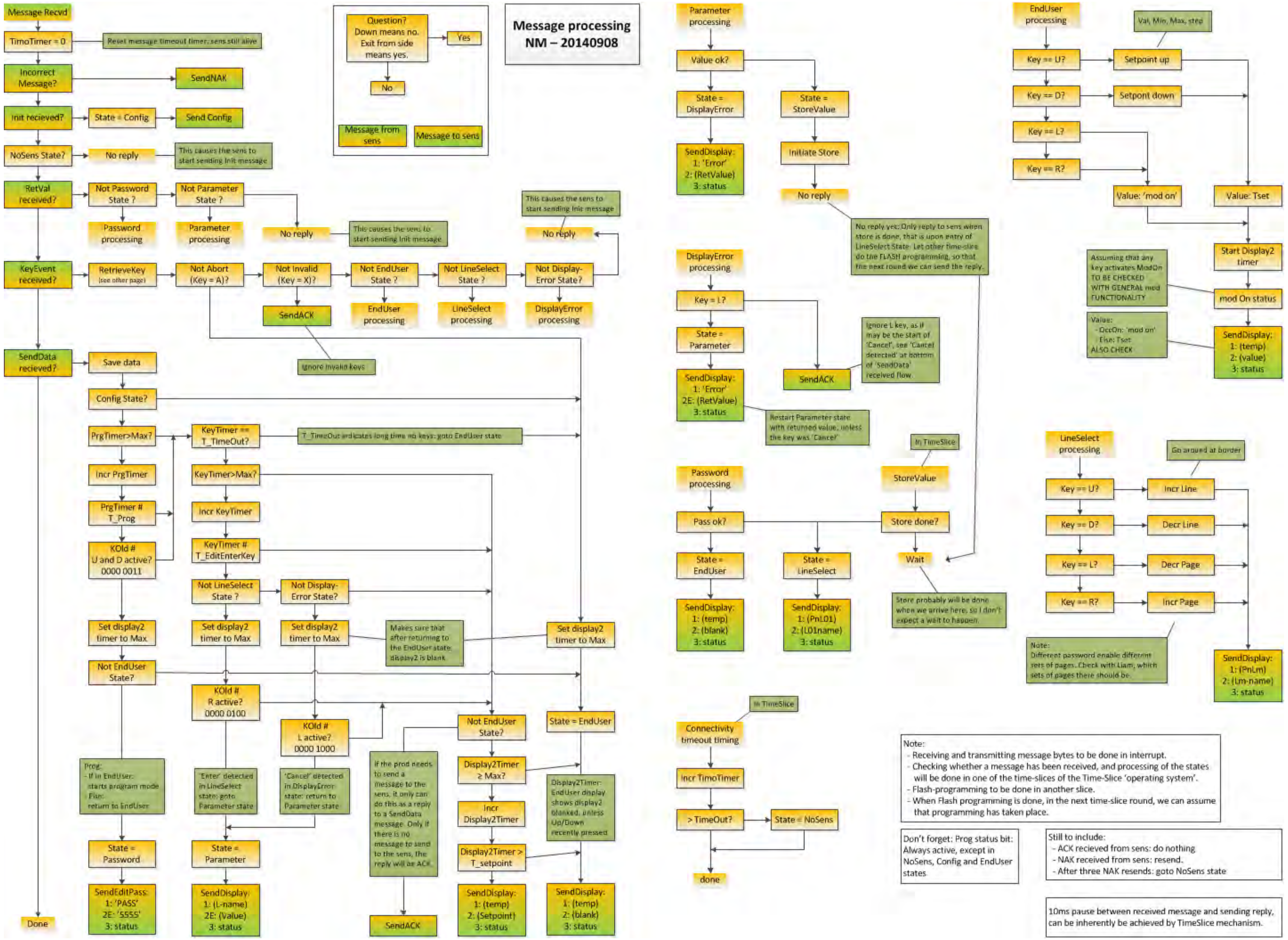
47 pages documentation condensed into one page



BasicOperation
NM - 20140906

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.

Design example



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 ?
Password processing
Not Parameter State ?
Parameter processing
No reply
This causes the sens to start sending Init message
KeyEvent received?
RetrieveKey (see other page)
Not Abort (Key = A)?
Not Invalid (Key = X)?
SendACK
Ignore invalid keys
Not EndUser State ?
EndUser processing
Not LineSelect State ?
LineSelect processing
Not Display-Error State?
DisplayError processing
This causes the sens to start sending Init message
No reply
SendData recieved?
Save data
Config State?
PrgTimer>Max?
Incr PrgTimer
PrgTimer # T_Prog
KOld # U and D active? 0000 0011
KeyTimer == T_TimeOut?
T_TimeOut indicates long time no keys: goto EndUser state
KeyTimer>Max?
Incr KeyTimer
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
Display2Timer ≥ Max?
Incr Display2Timer
Display2Timer > T_setpoint
Display2Timer: EndUser display shows display2 blanked, unless Up/Down recently pressed
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.
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
SendDisplay: 1: 'Error' 2: (RetValue) 3: status
State = StoreValue
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
SendDisplay: 1: 'Error' 2E: (RetValue) 3: status
Restart Parameter state with returned value, unless the key was 'Cancel'
SendACK
Ignore L key, as it may be the start of 'Cancel', see 'Cancel detected' at bottom of 'SendData' received flow.

Password processing
Pass ok?
State = EndUser
SendDisplay: 1: (temp) 2: (blank) 3: status
State = LineSelect
SendDisplay: 1: (PnL01) 2: (L01name) 3: status
In TimeSlice
StoreValue
Store done?
Wait
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
Key == U?
Setpoint up
Val, Min, Max, step
Key == D?
Setpont down
Key == L?
Key == R?
Value: 'mod on'
Value: Tset
Start Display2 timer
mod On status
Assuming that any key activates ModOn TO BE CHECKED WITH GENERAL mod FUNCTIONALITY
Value:
- OccOn: 'mod on'
Else: Tset
ALSO CHECK
SendDisplay: 1: (temp) 2: (value) 3: status

LineSelect processing
Key == U?
Incr Line
Go around at border
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.

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 need *improved performance* of functionality

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

- They actually succeeded !

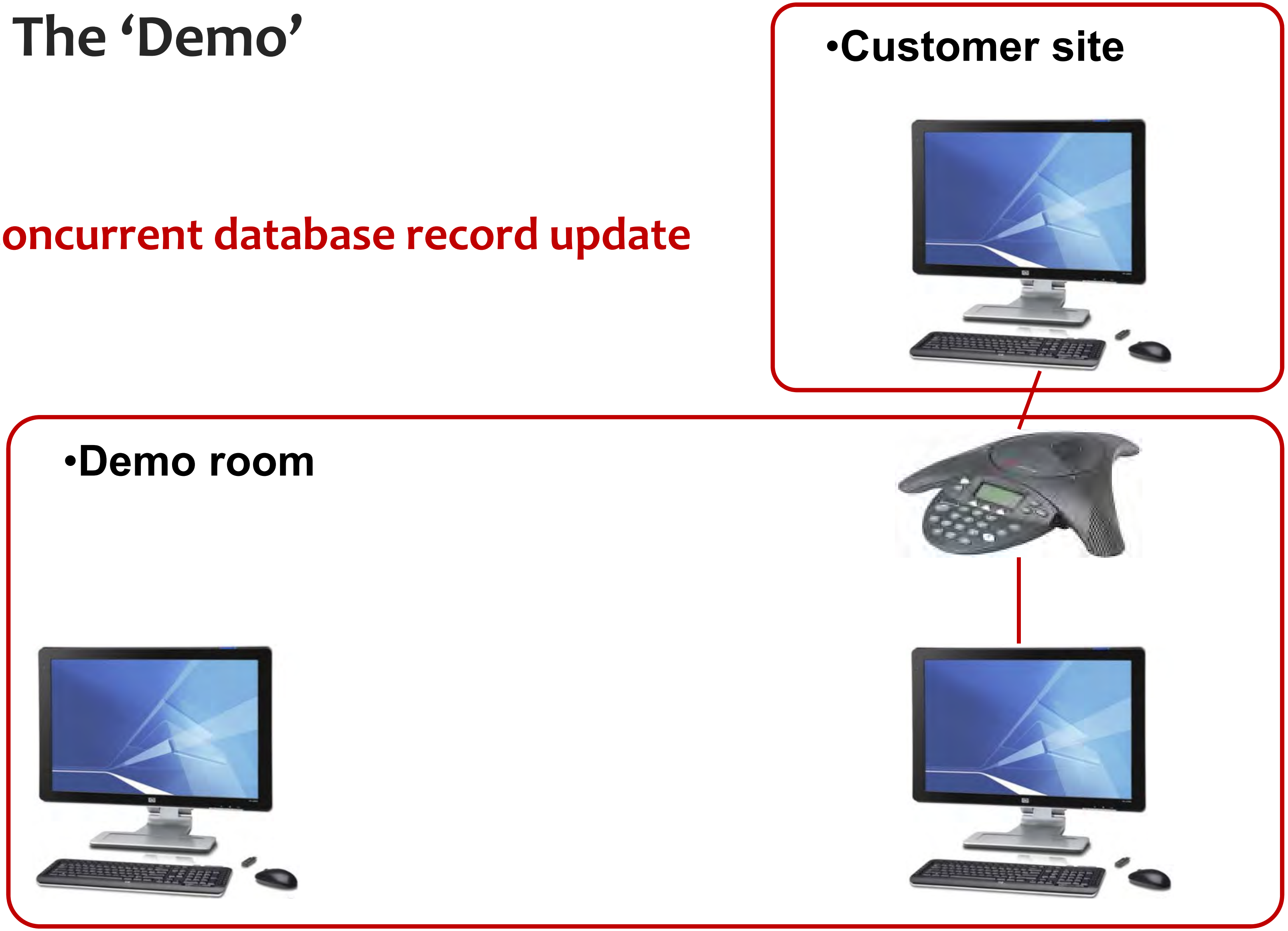
If we deliver

- Give the delivery to the stakeholders
- Keep your hands handcuffed on your back
- Keep your mouth shut
- and o-b-s-e-r-v-e what happens
- Seeing what the stakeholders actually do provides so much better feedback
- Then we can ‘talk business’ with the stakeholders

- Is this what you do ?

The ‘Demo’

Concurrent database record update



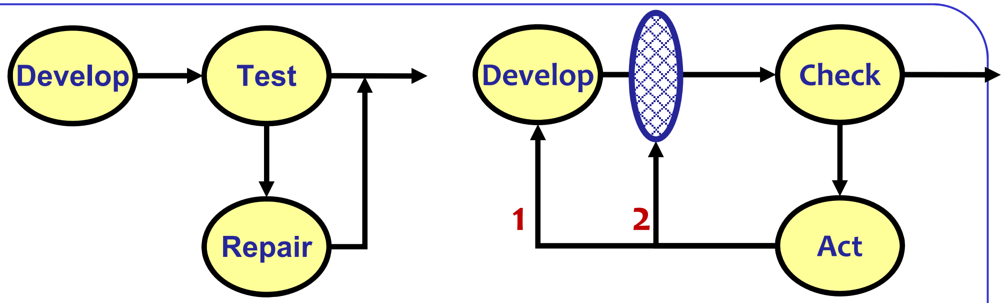
- Customer site

- Demo room

Large distributed system

- Busy with System Test 3, planned 4 and 5, probably 6
- PONQ of the organization wasn't very helpful for developers
- Translation: Zero Defects means just one System Test
- Requirement: No Questions - No Issues
- You find out how to do that
- Creating pain: No pain - no gain

Zero Defects at test



What we often see

What we should expect

- **What is delivered simply works**
  If it doesn't work, nothing was delivered
- **At the formal test: "No questions, no issues"**
- **Before that, in daily integrations:**
  work together, ask anything, learn anything, improve anything
- **Design Log - Review - Code - Review**
  iterate as needed *before* test (don't waste testers' time)
- **Daily integration** ("That's impossible !" Of course it was possible!)

→ The shorter the feedback-loop the better prevention improves

How much 'initial legwork' are you doing ?

- 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? (i.e. quality - versus expectation)*
- *Adrian: It's exactly as expected, which is absolutely unprecedented for a first delivery;*
  the initial legwork has really paid off

Short-Circuiting

- Firmware (in a controller) and Software in a PC on a network
- 9 months issues with communication - trial and error, no design

- Put them both in front of the whiteboard
- Let them draw the communication flow
- Quickly found the bottleneck
- Decided how to solve it
- Solved within a week

- I call this 'Short-Circuiting'

Some ‘laws’

- **When test is the principal defect removal method during development, corrective maintenance will account for the majority of the maintenance spent**
- **The number of defects found in production use will be inversely proportional to the number of defects removed prior to integration, system, and acceptance testing**
- **The number of defects found in production use will be directly proportional to the number of defects removed during integration, system, and acceptance testing**
- **We have a global reputation … for consistently delivering nearly defect free software on predictable cost and schedule. We offer firm *fixed price contracting* with *performance guarantees* including *lifetime warranty* on software defects.**

Girish Seshagiri

Some techniques shown

- Design
- Drawings
- DesignLog
- Review - Inspection
- Short-Circuiting

- Zero Defects attitude makes an immediate difference

More

www.malotaux.nl/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

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

www.malotaux.nl/inspections

Inspection pages

Approaching Zero Defects Is Absolutely Possible

If in doubt, let's talk about it

Niels Malotaux

N R Malotaux
Consultancy

niels@malotaux.nl

www.malotaux.nl