Product Development in TANDBERG

olve.maudal@tandberg.com

TANDBERG has never cared much about documentation, procedures, methodologies and risk reduction. However, we do care very much about our culture and our principles. This has enabled us to outperform all our competitors in the video conferencing and telepresence market during the last decade.

In retrospect, we realize that TANDBERG has for 10-15 years built a culture that is quite compatible with Agile and Lean ideas.

In this talk I will start by giving a glimpse into TANDBERG R&D at Lysaker. Then I will present the company and our products. After the break I will give an example of how we developed a particular product with emphasis on software development, before I dive into the principles that we follow.

A presentation for Oslo Lean Meetup
February 9, 2010

Disclaimer: This is an extremely subjective view of how we do product development in TANDBERG. Please do not assume that it is possible to generalize well over the examples given.

About Olve

geek - very proud of being a computer programmer.

Been programming nearly every day since I bought my first computer - the Commodore VIC-20.

Studying Software Engineering (BEng in Manchester 1992-1995), Artificial Intelligence (MSc in Edinburgh 1995-1996) and did some postgrad studies in Knowledge Discovery (NTNU 1996). Professionally I started developing systems for finding oil (Schlumberger 1996-2000), then I developed systems for moving money (BBS 2000-2004), and now I am developing systems for effective communication between people (TANDBERG since 2004).

Active member of the vibrant geek community in Oslo. Eg, JavaPils, Smidig, JavaZone, XP Meetup, Cantara, Lean Meetup, Rubberducks and Oslo C++ Users Group, and a lot of other things. Also an active member of ACCU.

Blogs regularly on http://olvemaudal.wordpress.com/ and Twitter @olvemaudal

As a software engineer joining TANDBERG...

at first you might get this impression...

at first you might get this impression...

- No documentation

at first you might get this impression...

- No documentation
- No routines

at first you might get this impression...

- No documentation
- No routines
- Fooling around

at first you might get this impression...

- No documentation
- No routines
- Fooling around
- Not following plans

at first you might get this impression...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed

at first you might get this impression...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides

at first you might get this impression...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management

at first you might get this impression...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization

at first you might get this impression...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision

at first you might get this impression...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness

at first you might get this impression...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers
- No integration period

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers
- No integration period
- Spectacular products

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers
- No integration period
- Spectacular products
- Fast deliveries

but then you start to notice...

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers
- No integration period
- Spectacular products
- Fast deliveries
- Sustainable pace

... and while you still see the "negative" stuff, you will start to appreciate the "positive" stuff more.

- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers
- No integration period
- Spectacular products
- Fast deliveries
- Sustainable pace

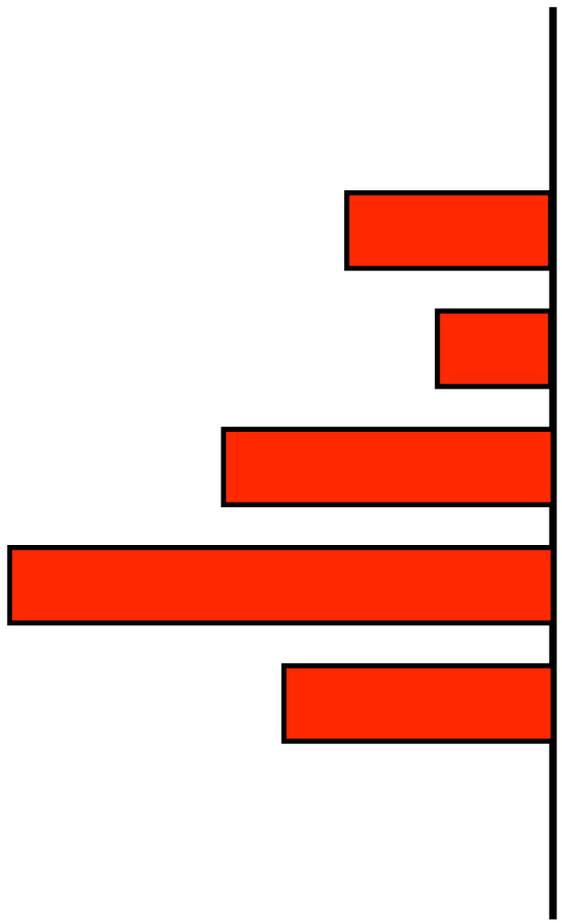
... and while you still see the "negative" stuff, you will start to appreciate the "positive" stuff more.

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers
- No integration period
- Spectacular products
- Fast deliveries
- Sustainable pace

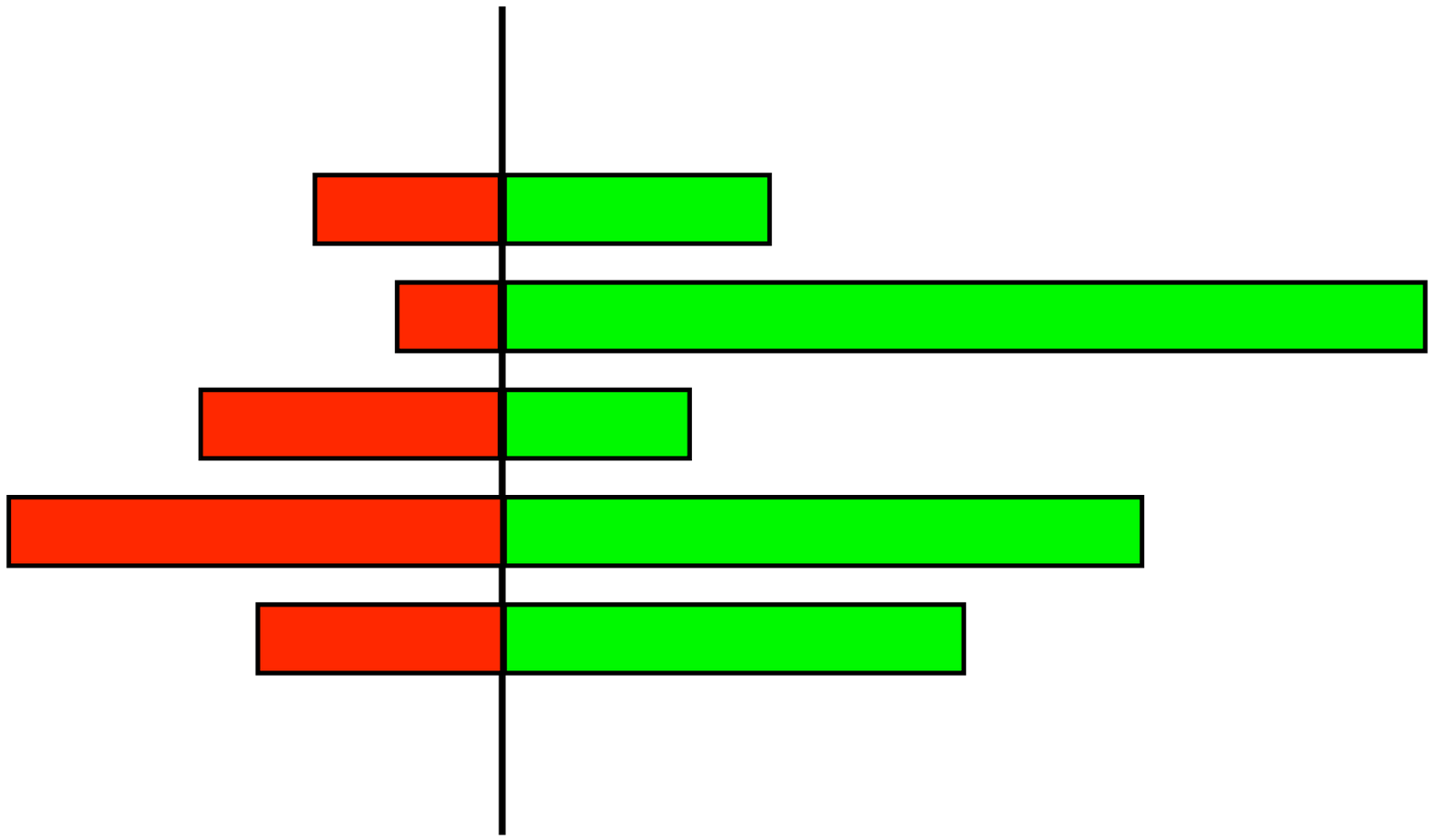
- No documentation
- No routines
- Fooling around
- Not following plans
- Decision are postponed
- Nobody decides
- Little respect for management
- Little modularization
- Lack of precision
- Sloppiness
- People are not working hard

Some thoughts about negative and positive components

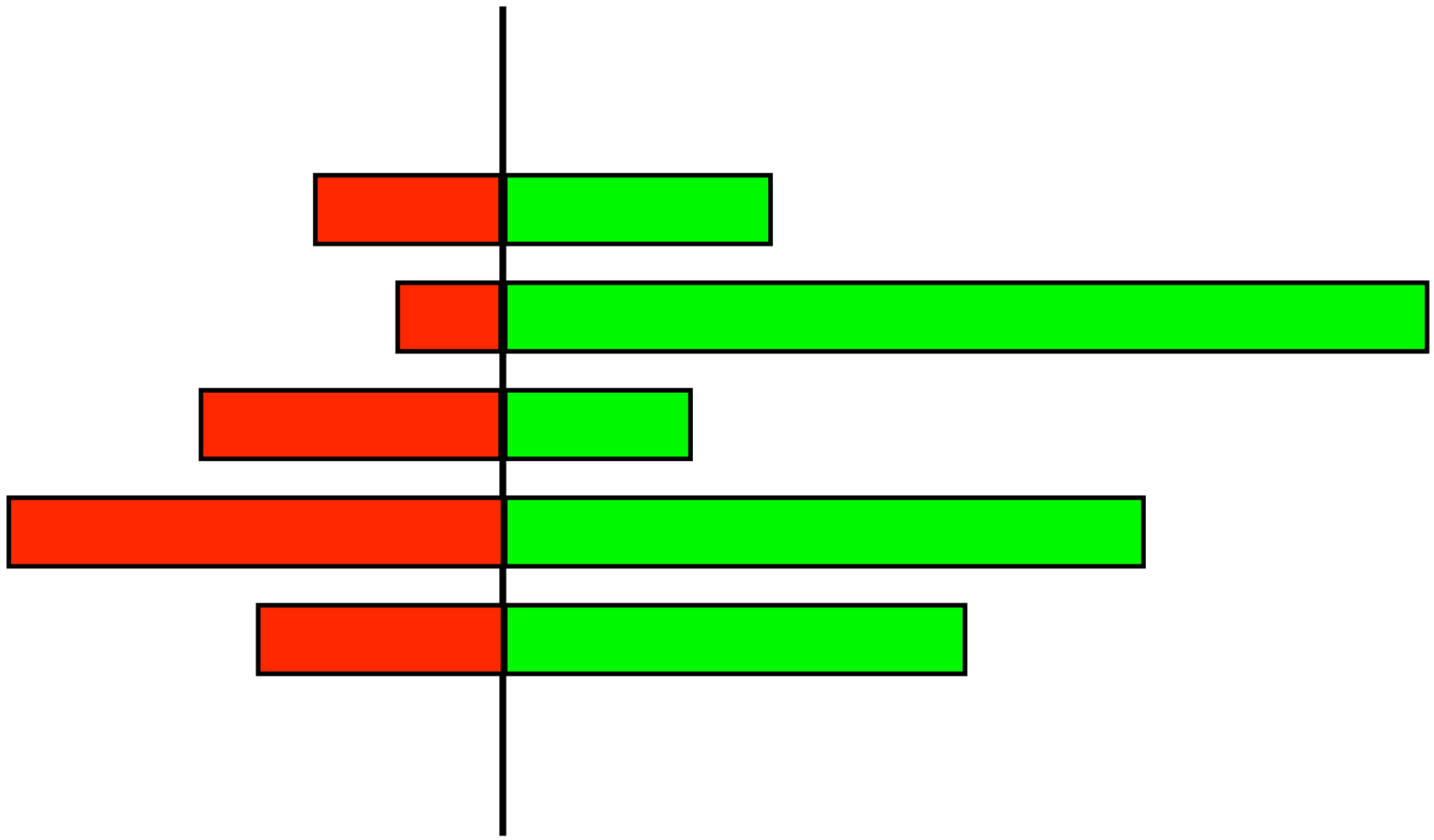
everything has a negative component ...



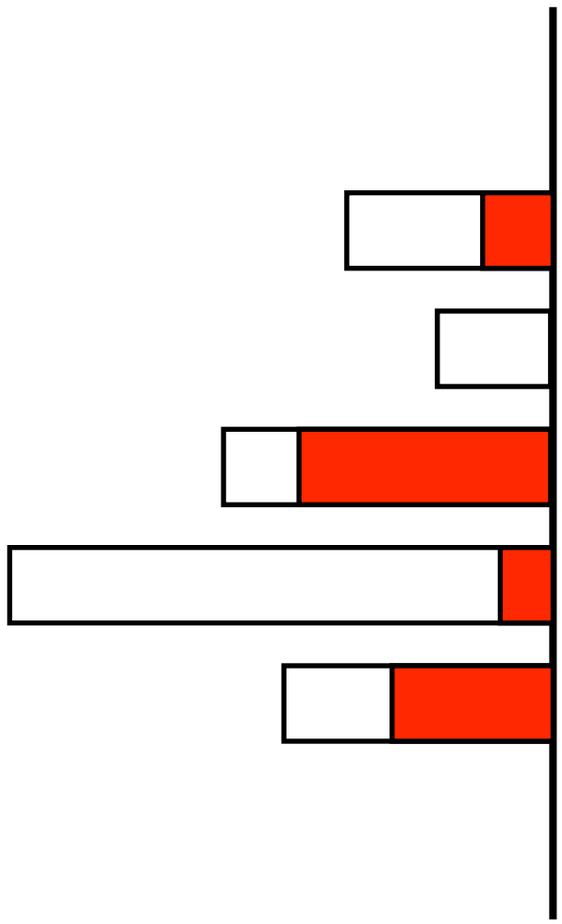
... as well as a positive component



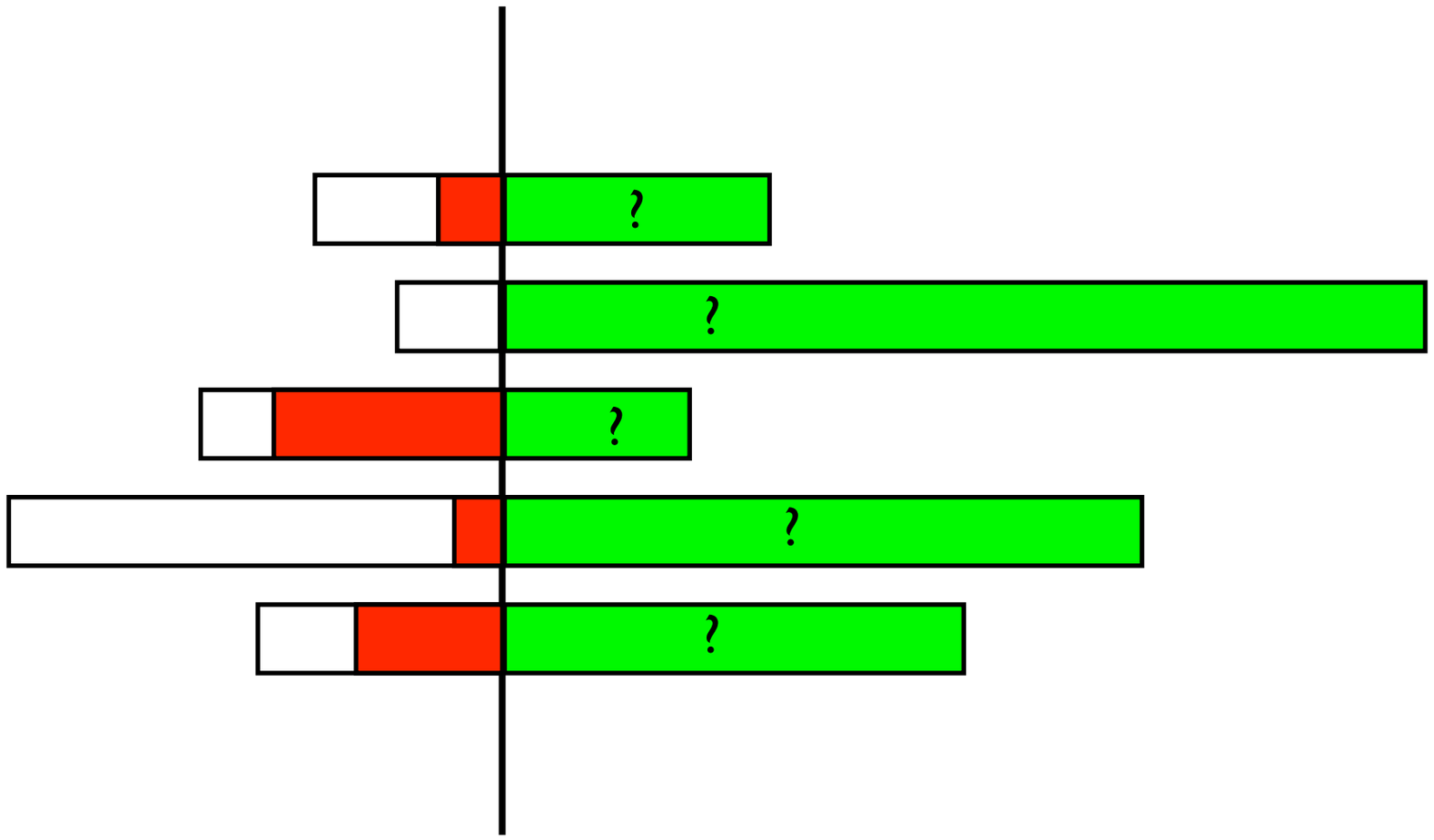
so if you want to improve something...



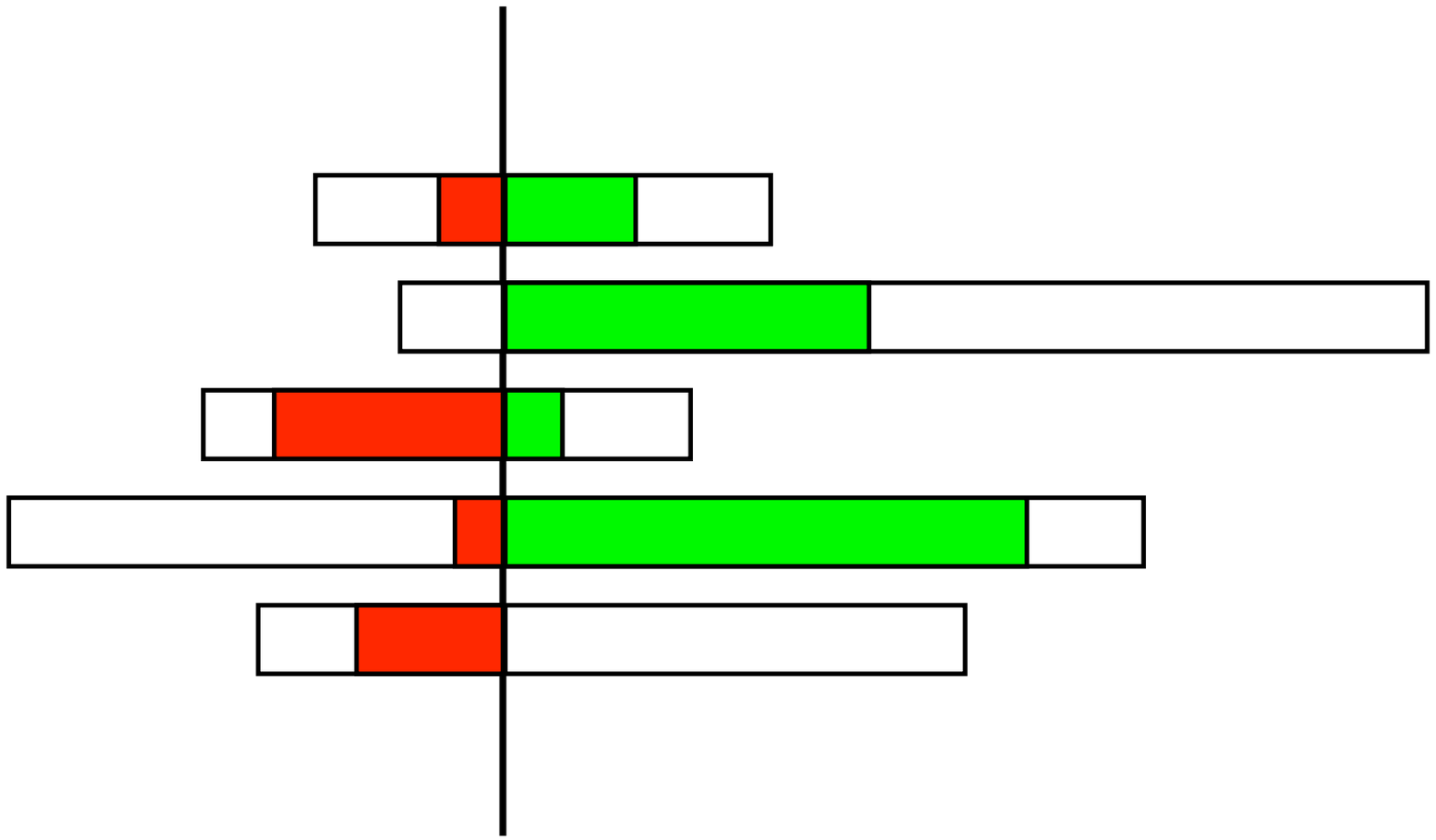
... do **not** try to fix the negative stuff ...



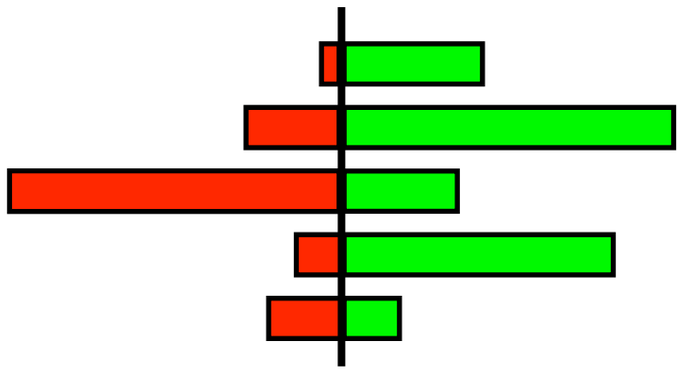
... before understanding how it will affect the positive component



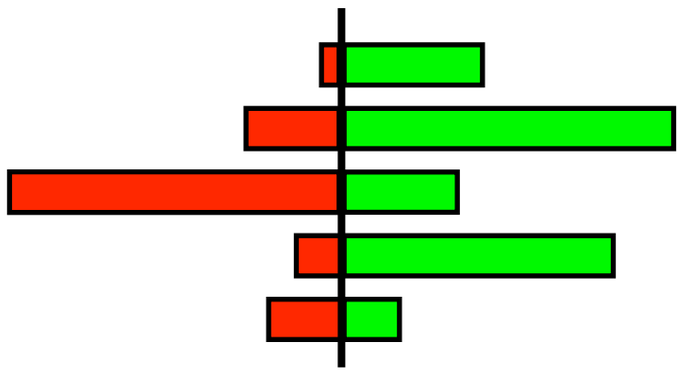
because you might end up by reducing the positive component by even more



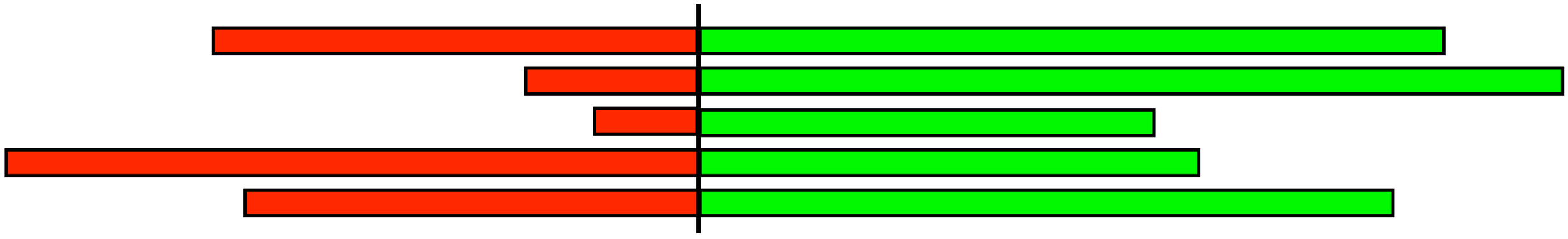
while some organizations might have a profile like this



while some organizations might have a profile like this

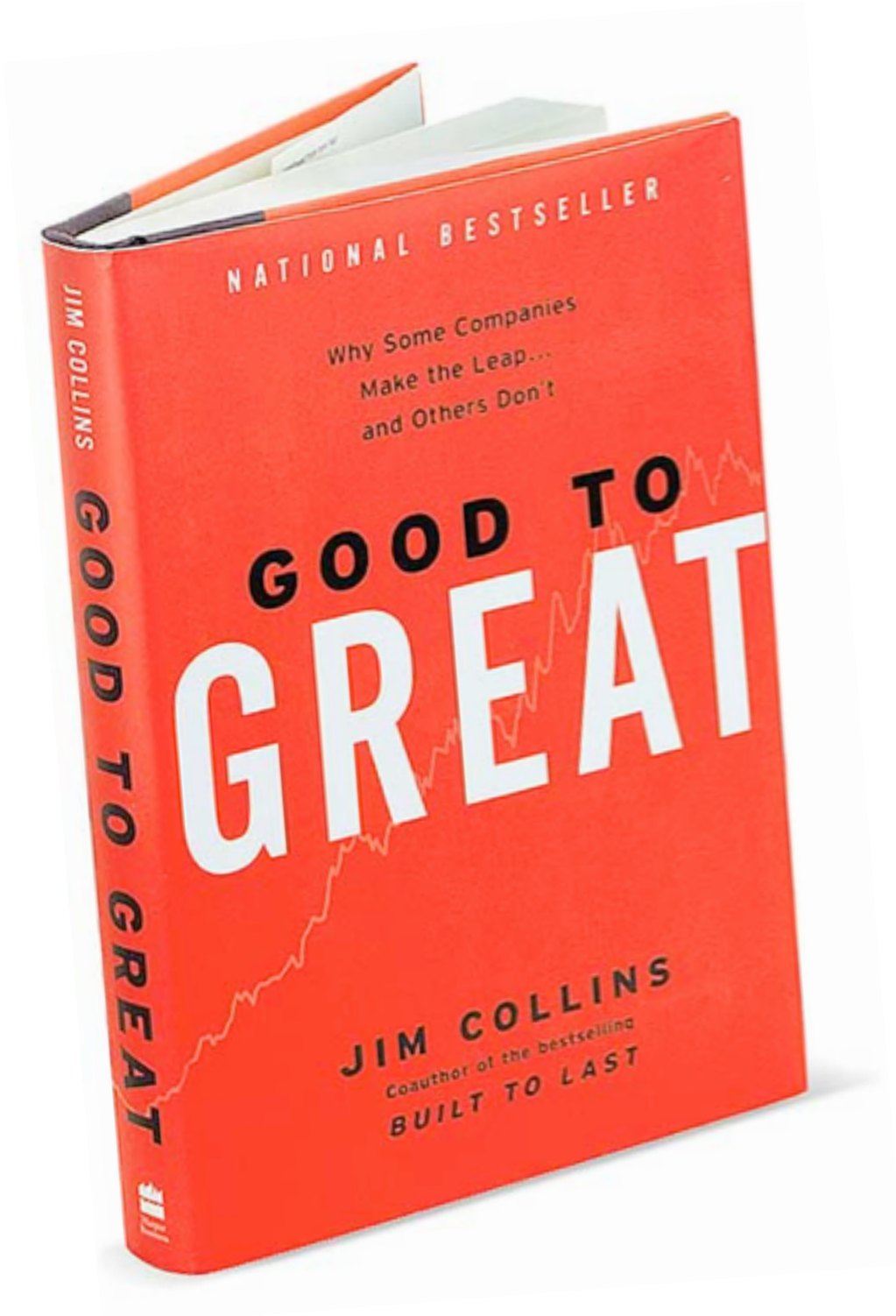


the TANDBERG profile looks more like this



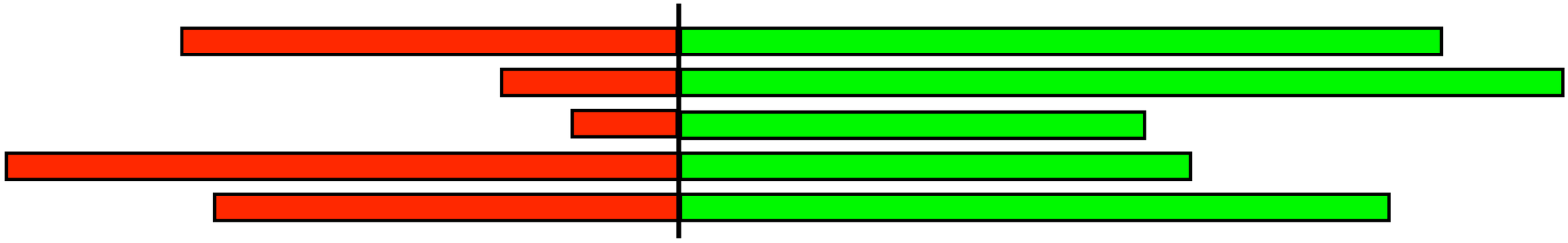
The TANDBERG culture has been extremely focused on improving things that we are already quite good at, and spend less time on worrying about things that "should" have been improved.


NILS ARNE EGGEN
GODFOTEN
SAMHANDLING
– VEIEN TIL SUKSESS
ASCHEHOUG
PSYKOLOGI
"... nyttig fagstoff, antagelig det beste som er produsert her til lands – fordi det er originalt, ekte og enestående."
OTTO ULSETH, ADRESSEAVISEN



Managing your problems can only make you good, whereas building your opportunities is the only way to become great. (Collins, 2001)

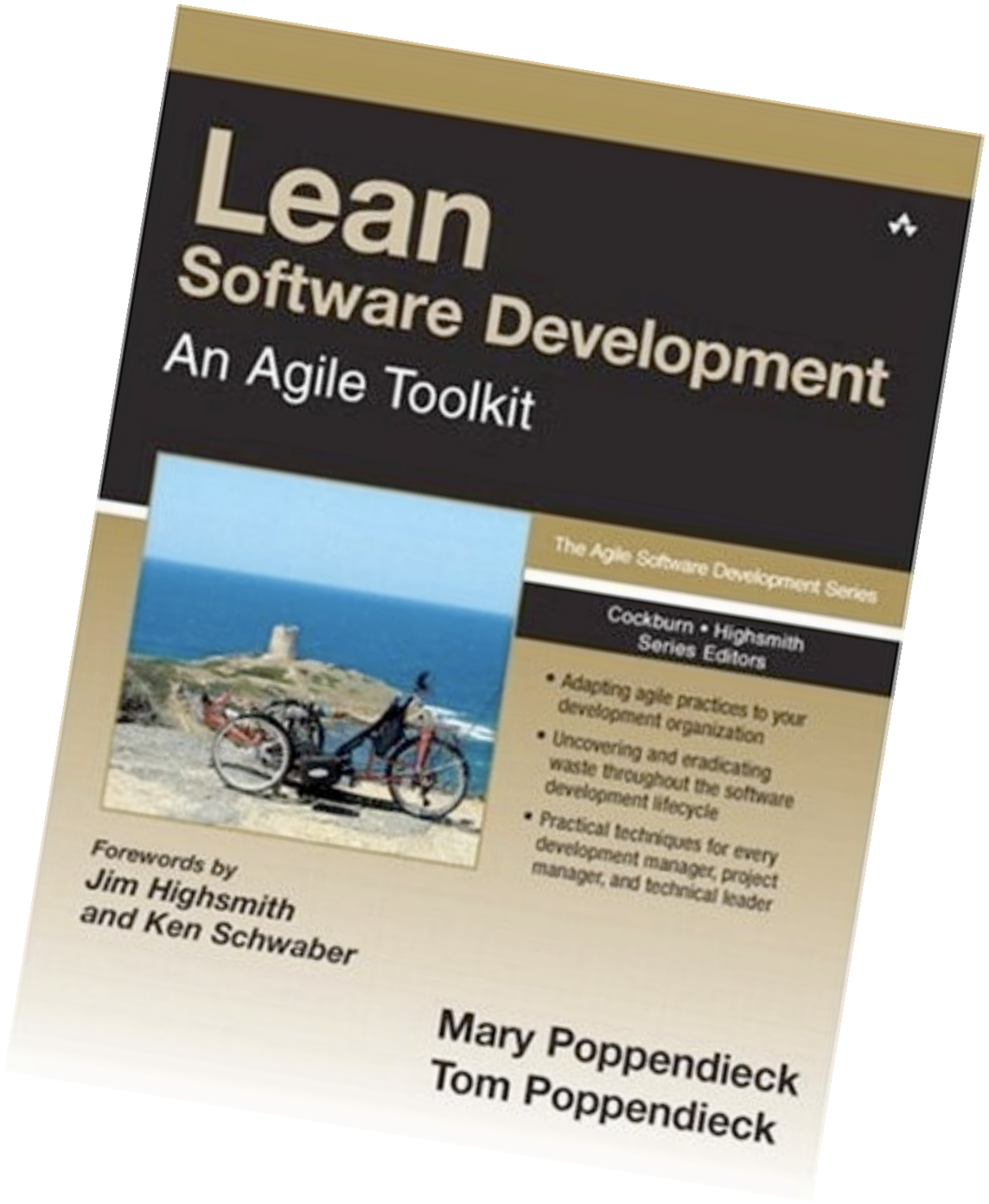
Indeed, this is the signature of a GREAT organization



and it explains why we are winning over and over again, even if it (to some) looks like the we are doing everything wrong.

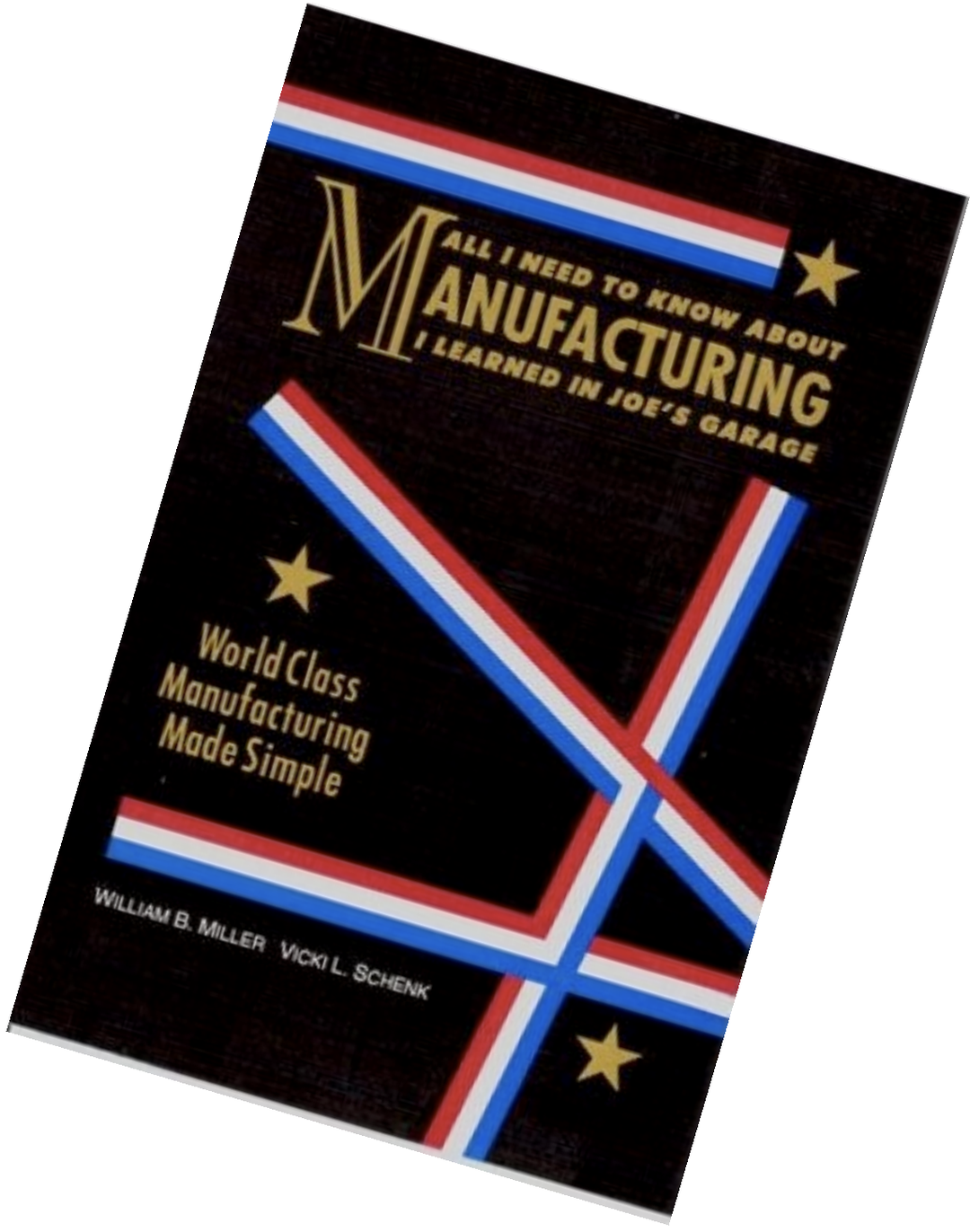
Observations from TANDBERG

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers
- No integration period
- Spectacular products
- Fast deliveries
- Sustainable pace

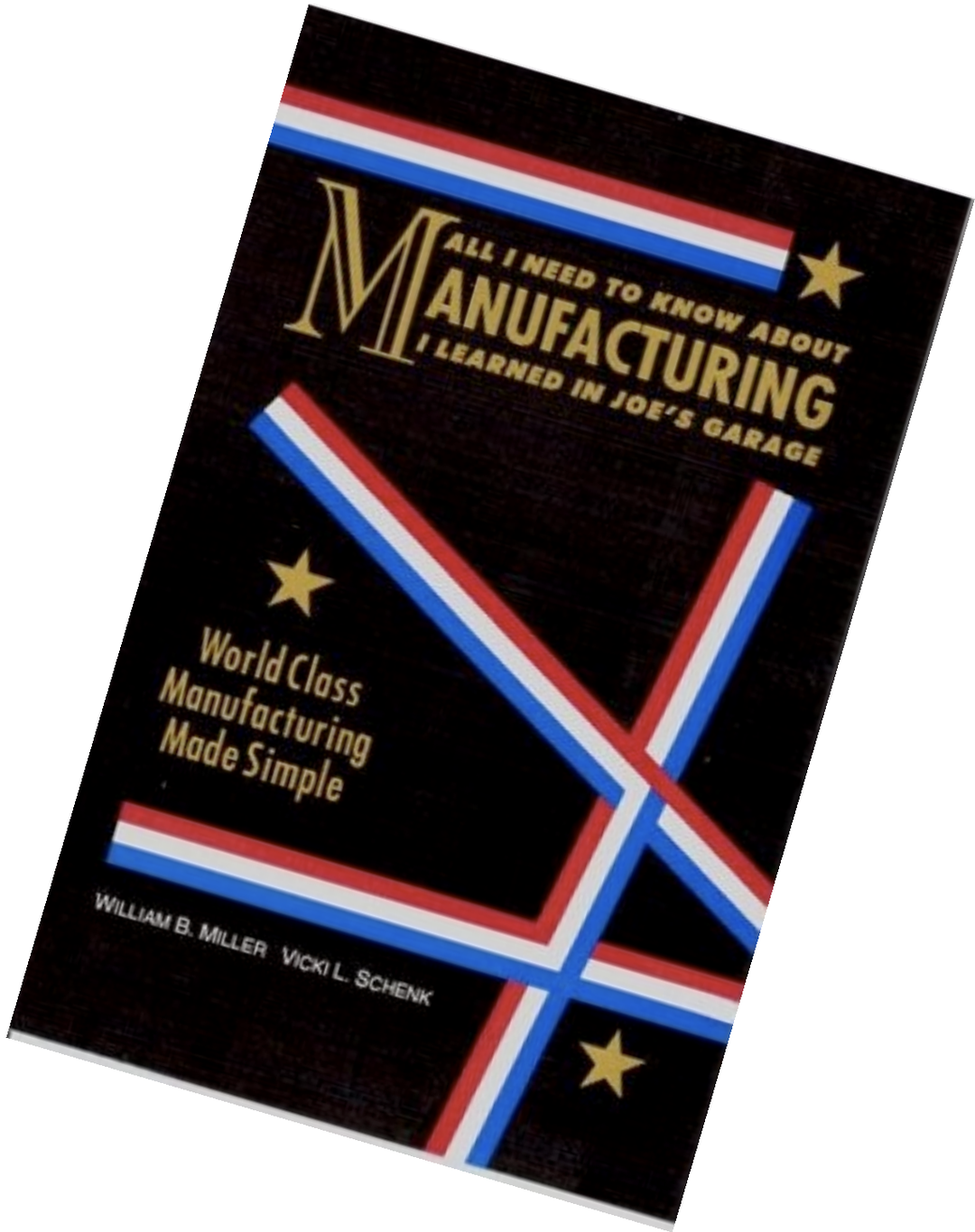

Lean
Software Development
An Agile Toolkit
The Agile Software Development Series
Cockburn • Highsmith
Series Editors
• Adapting agile practices to your development organization
• Uncovering and eradicating waste throughout the software development lifecycle
• Practical techniques for every development manager, project manager, and technical leader
Forewords by
Jim Highsmith
and Ken Schwaber
Mary Poppendieck
Tom Poppendieck


Lean
Software Development
An Agile Toolkit
The Agile Software Development Series
Cockburn • Highsmith
Series Editors
• Adapting agile practices to your development organization
• Uncovering and eradicating waste throughout the software development lifecycle
practical techniques for every
velopment manager, project
nager, and technical leader
oppendieck
oppendieck
NATIONAL BESTSELLER
"The best current book on the changes reshaping manufacturing and the most readable." —Business Week
THE MACHINE THAT CHANGED THE WORLD
THE STORY OF LEAN PRODUCTION
HOW JAPAN'S SECRET WEAPON IN THE GLOBAL AUTO WARS WILL REVOLUTIONIZE WESTERN INDUSTRY
THE MIT
INTERNATIONAL MOTOR VEHICLE PROGRAM
JAMES P. WOMACK, DANIEL T. JONES, AND DANIEL ROOS


Lean
Software Development
An Agile Toolkit
The Agile Software Development Series
Cockburn • Highsmith
Series Editors
• Adapting agile practices to your development organization
• Uncovering and eradicating waste throughout the software development lifecycle
practical techniques for every
velopment manager, project
nager, and technical leader
oppendieck
oppendieck
NATIONAL BESTSELLER
"The best current book on the changes reshaping manufacturing and the most readable." —Business Week
THE MACHINE
THAT CHANGED
THE WORLD
THE STORY OF
LEAN PRODUCTION
HOW JAPAN'S SECRET
WEAPON IN THE
GLOBAL AUTO WARS
WILL REVOLUTIONIZE
WESTERN INDUSTRY
THE
MIT
INTERNATIONAL
MOTOR VEHICLE
PROGRAM
JAMES P. WOMACK, DANIEL T. JONES, AND DANIEL ROOS


ALL I NEED TO KNOW ABOUT
MANUFACTURING
I LEARNED IN JOE'S GARAGE
World Class
Manufacturing
Made Simple
WILLIAM B. MILLER VICKI L. SCHENK


Lean
Software Development
An Agile Toolkit
NATIONAL BEST
"The best current book on the changes reshaping m
readable." —Business W
THE M
THAT C
TH
JAMES P. WOMACK, DANIEL T. JONES, AND
Taiichi
Ohno
TOYOTA PRODUCTION SYSTEM
Beyond Large-Scale Production


ALL I NEED TO KNOW ABOUT
MANUFACTURING
I LEARNED IN JOE'S GARAGE
World Class
Manufacturing
Made Simple
WILLIAM B. MILLER VICKI L. SCHENK

Lean
Software Development
An Agile Toolkit

NATIONAL BEST

"The best current book on the changes reshaping m
readable." —Business W

THE M
THAT C
TH

JAMES P. WOMACK, DANIEL T. JONES, AND

Taiichi
Ohno

TOYOTA PRODUCTION SYSTEM
Beyond Large-Scale Production

Now includes
Eli Goldratt's
Personal Story
"My Saga"

Eliyahu M. Goldratt and Jeff Cox

THE GOAL

A PROCESS OF ONGOING IMPROVEMENT

Eli Goldratt has been described by **Fortune** as a **"guru to industry"** and by **Business Week** as a **"genius"**. His book, ***The Goal***, is a gripping fast-paced business novel.

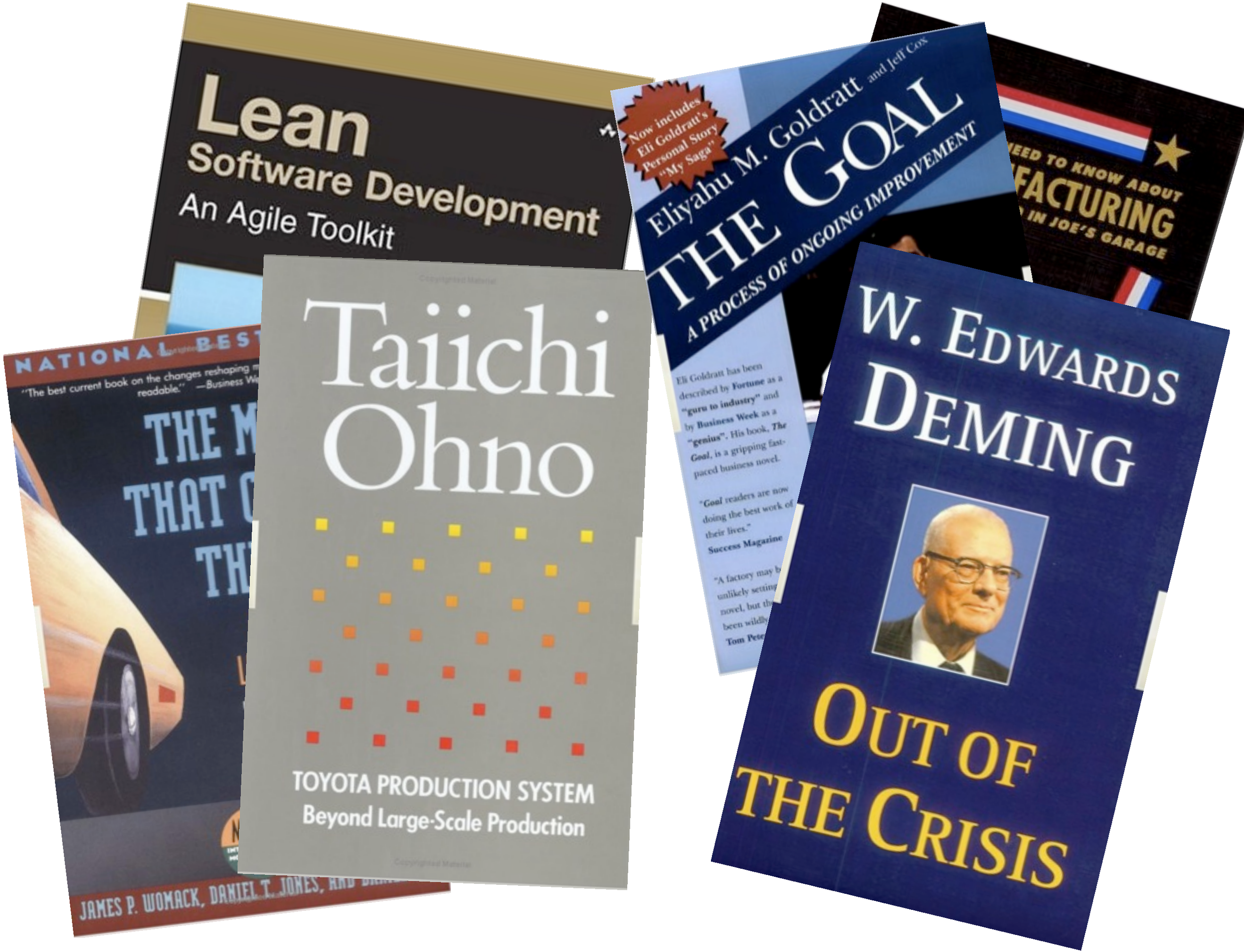
"***Goal*** readers are now doing the best work of their lives."
Success Magazine

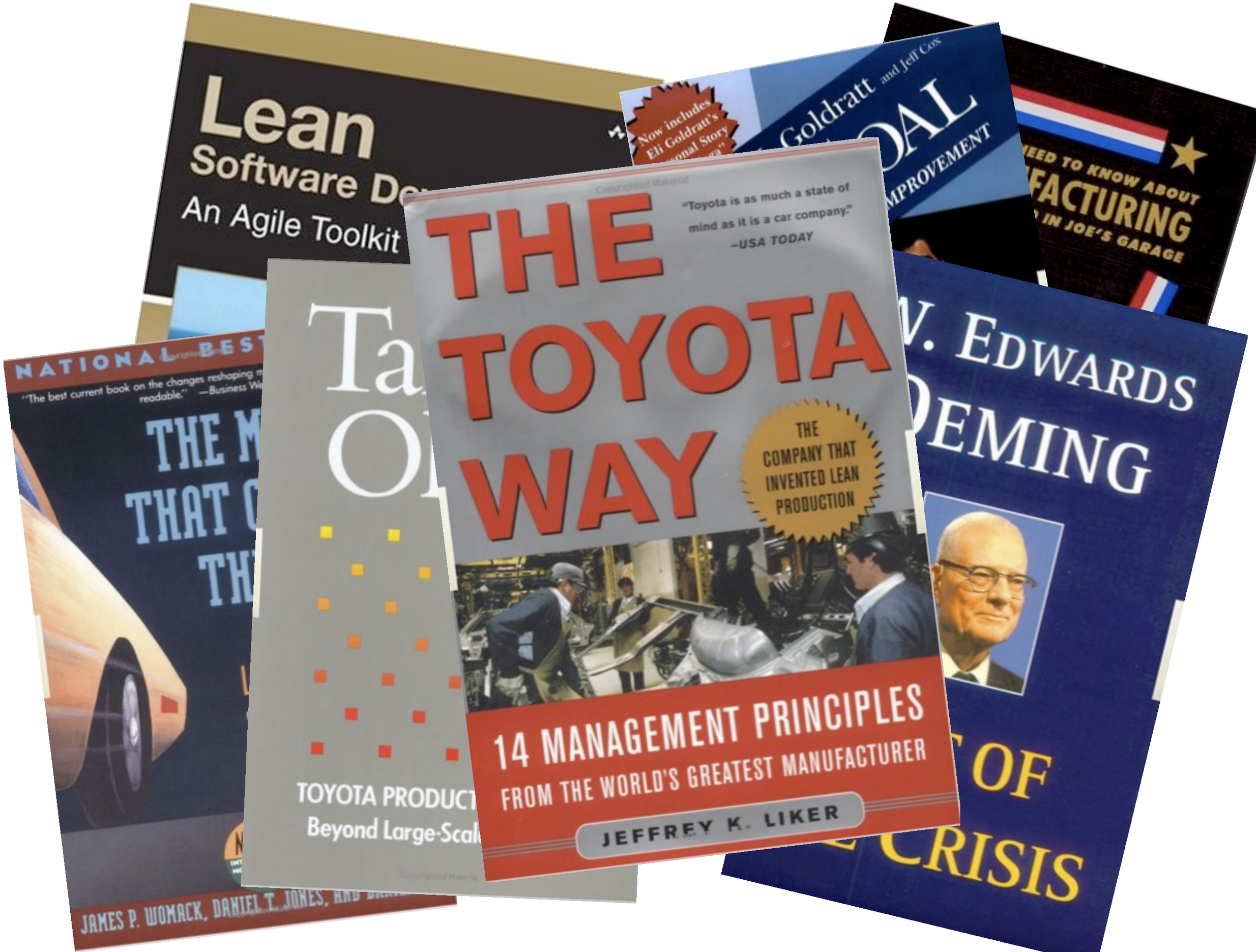
"A factory may be an unlikely setting for a novel, but the book has been wildly effective…"
Tom Peters

THE BEST-SELLING BUSINESS NOVEL THAT INTRODUCED THE
THEORY OF CONSTRAINTS
AND CHANGED HOW AMERICA DOES BUSINESS.

OVER 2 MILLION COPIES SOLD!
SECOND REVISED EDITION

EED TO KNOW ABOUT
FACTURING
IN JOE'S GARAGE


Lean
Software Development
An Agile Toolkit
Taiichi
Ohno
TOYOTA PRODUCTION SYSTEM
Beyond Large-Scale Production
JAMES P. WOMACK, DANIEL T. JONES, AND
Now includes
Eli Goldratt's
Personal Story
"My Saga"
Eliyahu M. Goldratt and Jeff Cox
THE GOAL
A PROCESS OF ONGOING IMPROVEMENT
Eli Goldratt has been described by Fortune as a "guru to industry" and by Business Week as a "genius". His book, The Goal, is a gripping fast-paced business novel.
"Goal readers are now doing the best work of their lives."
Success Magazine
NEED TO KNOW ABOUT
FACTURING
IN JOE'S GARAGE
W. EDWARDS
DEMING
OUT OF
THE CRISIS


Lean
Software De
An Agile Toolkit
Now includes
Eli Goldratt's
Goldratt and Jeff Cox
OAL
MPROVEMENT
EED TO KNOW ABOUT
FACTURING
IN JOE'S GARAGE
NATIONAL BEST
"The best current book on the changes reshaping
readable." —Business
THE M
THAT C
TH
Ta
O
THE
TOYOTA
WAY
"Toyota is as much a state of
mind as it is a car company."
–USA TODAY
THE
COMPANY THAT
INVENTED LEAN
PRODUCTION
14 MANAGEMENT PRINCIPLES
FROM THE WORLD'S GREATEST MANUFACTURER
JEFFREY K. LIKER
TOYOTA PRODUCT
Beyond Large-Scal
JAMES P. WOMACK, DANIEL T. JONES,
W. EDWARDS
EMING
T OF
CRISIS

THE TANDBERG WAY

SPEED AND PRECISION

Simplify - focus - act * Approximately right rather than accurately wrong * Think * Do it right the first time

INTEGRITY AND ENTHUSIASM

Sense of humour * Honesty * High ethical standards * Excitement * Trustworthiness * Loyalty

EXCEED EXPECTATIONS

Personal initiative * Fighting spirit * Go the last mile

FUN AND PROFIT

Maximize long term shareholder value * Pass på penga * One for all, all for one * Energy

TANDBERG FIRST

First in user benefits * Innovative * "Kreativ galskap" * Understanding customer needs

"We have to grow without sacrificing our culture and our values"

Jan Chr. Opsahl

Chairman

About TANDBERG

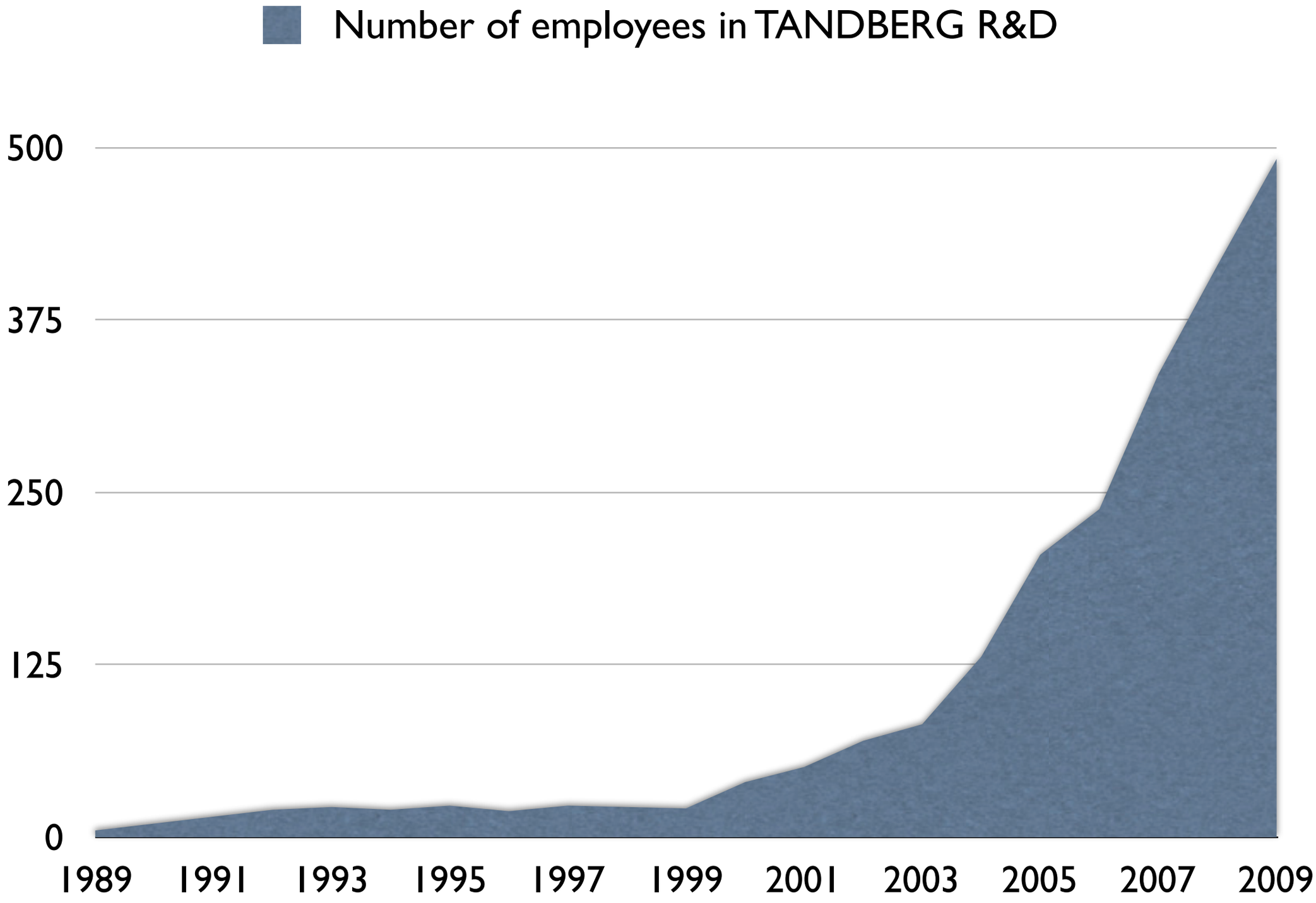
TANDBERG is the leading provider of telepresence, high-definition video conferencing and mobile video products and services. TANDBERG designs, develops and markets systems and software for video, voice and data. The company provides sales, support and value-added services in more than 90 countries worldwide.

TANDBERG shipped it's first product, a picture telephone for ISDN, in 1989. Since then TANDBERG has grown from a small startup based in Norway into an international company with ~1600 employees and a revenue of 800 MUSD in 2008. ~450 engineers works with product with research and development.

Dual headquarters in New York and Oslo. R&D centres at Lysaker (NO), Langley (UK), Ruscombe (UK), Bangalore (IN) and Hamilton (NZ).

www.tandberg.com

Breaking news (Dec 4, 2009): After a successful $3.4bn bid Cisco now controls more than 90% of shares in TANDBERG. The transaction is expected to close some time during 2010.



(the numbers are not exact)

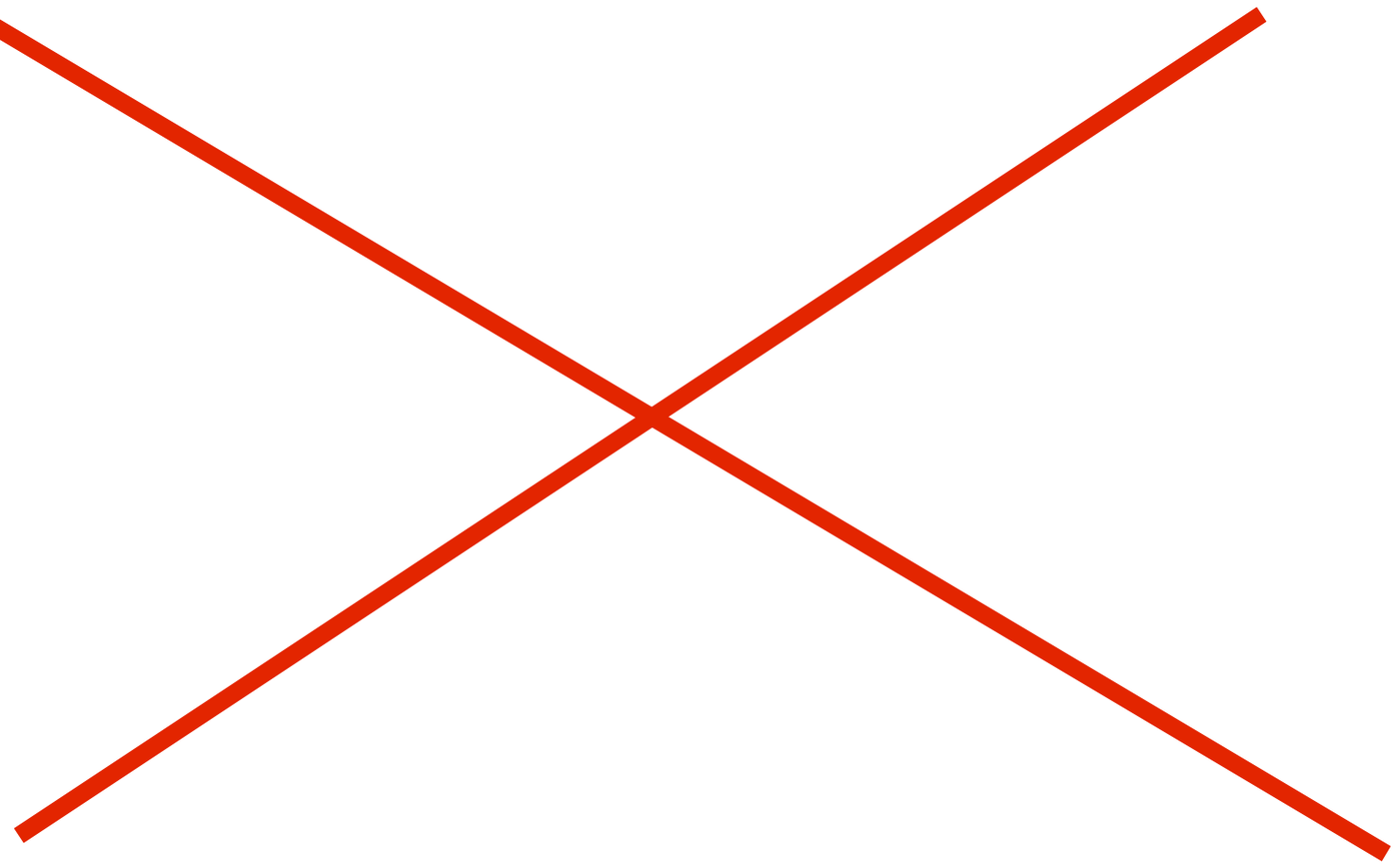

TANDBERG

TANDBERG

www.tandberg.com

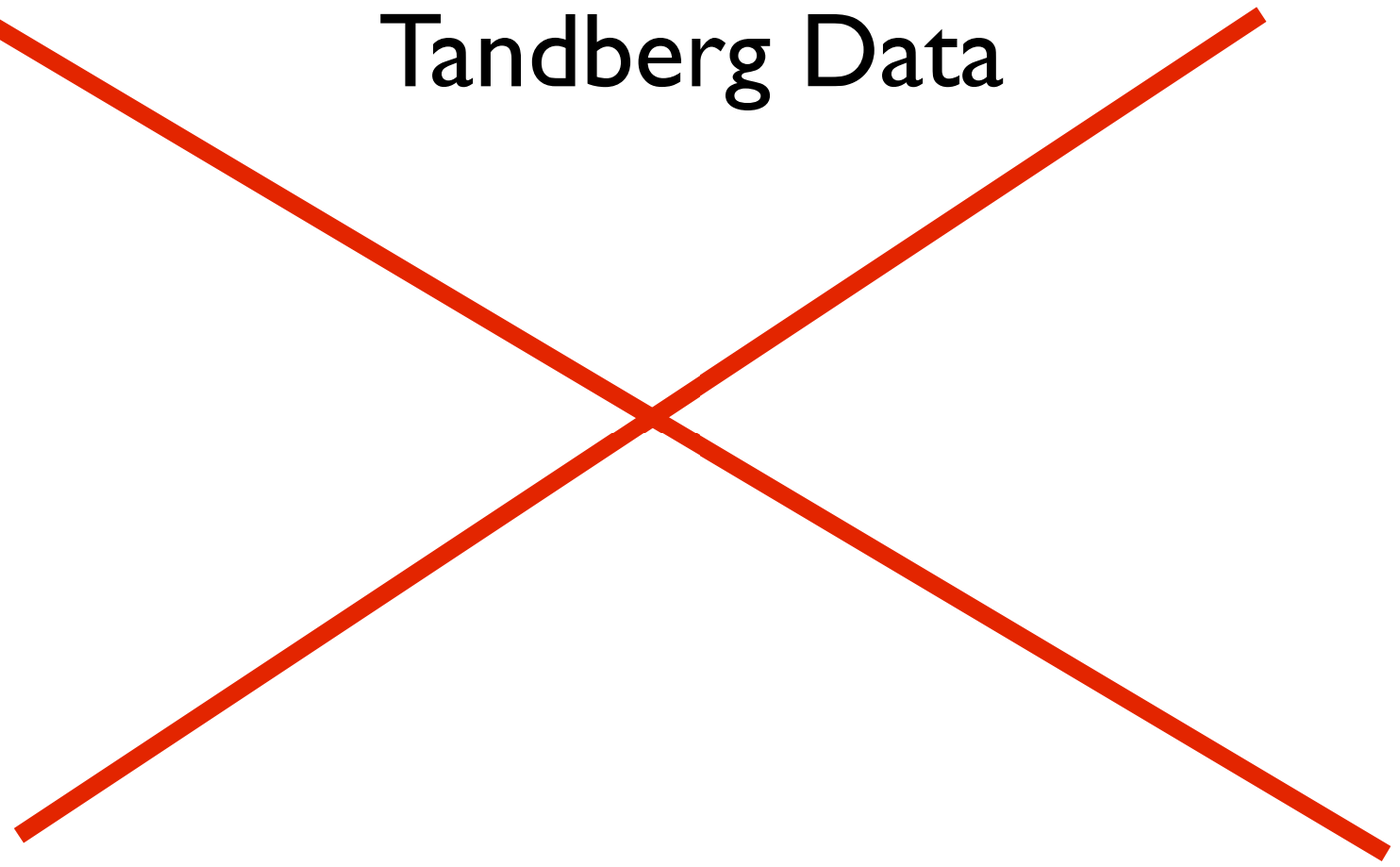
TANDBERG

www.tandberg.com



TANDBERG
www.tandberg.com

Tandberg Data



TANDBERG
www.tandberg.com

Tandberg Data
Tandberg Storage

TANDBERG

www.tandberg.com

~~Tandberg Data~~

~~Tandberg Storage~~

~~TANDBERG Television~~

TANDBERG
www.tandberg.com

~~Tandberg Data~~
~~Tandberg Storage~~
~~TANDBERG Television~~
~~TANDBERG Display~~

TANDBERG
www.tandberg.com

~~Tandberg Data~~
~~Tandberg Storage~~
~~TANDBERG Television~~
~~TANDBERG Display~~
~~Tandberg Eiendom~~

TANDBERG
www.tandberg.com

~~Tandberg Data~~
~~Tandberg Storage~~
~~TANDBERG Television~~
~~TANDBERG Display~~
~~Tandberg Eiendom~~
~~Robert Tandberg Møbler~~

TANDBERG
www.tandberg.com

Tandberg Data
Tandberg Storage
TANDBERG Television
TANDBERG Display
Tandberg Eiendom
Robert Tandberg Møbler
Tandberg's Bildeler, Larvik


TANDBERG

Video: The new way of working (2:00)

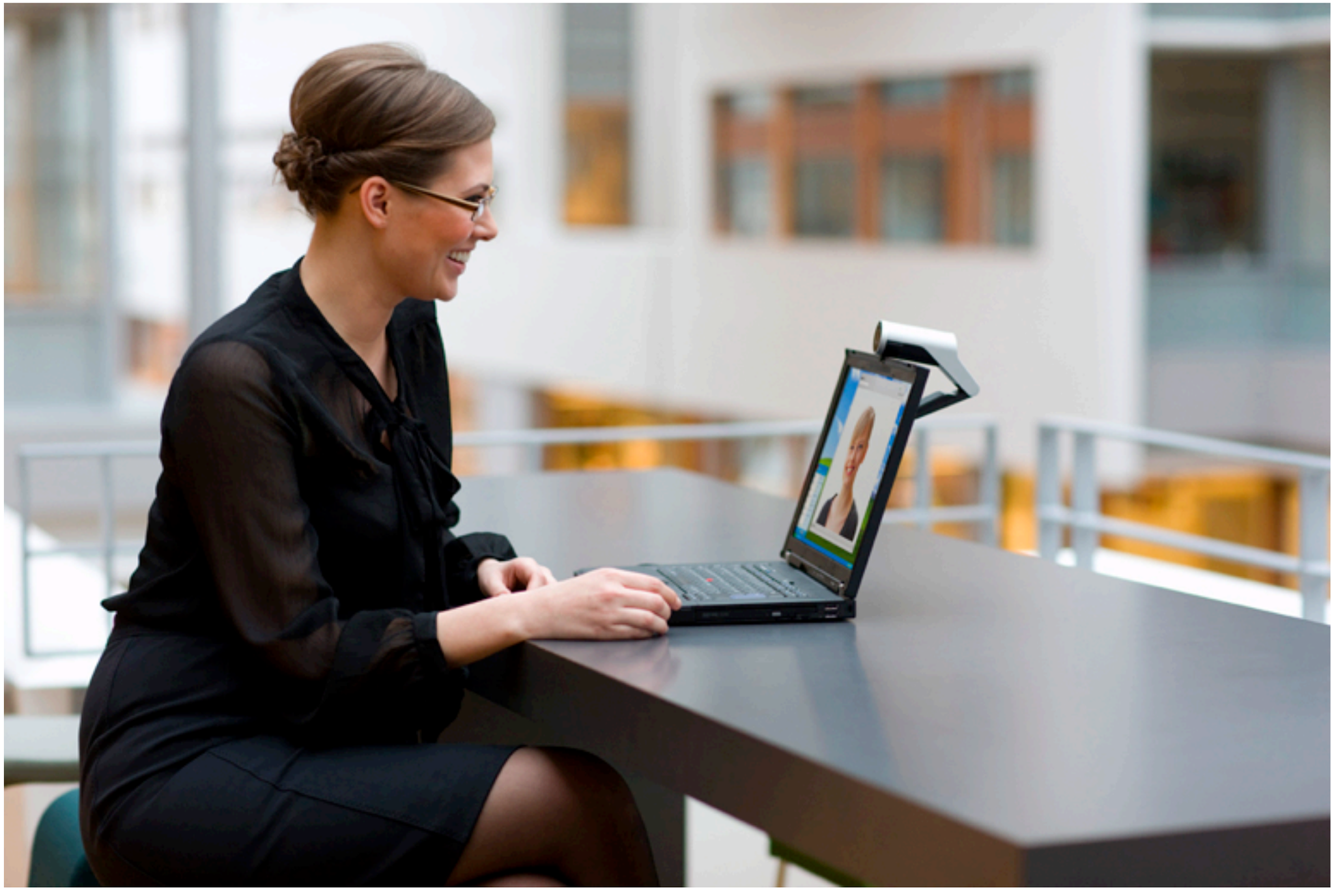
http://www.tandberg.com/media/index.jsp?id=1373

We develop and sell...

Meeting room systems

Telepresence systems

Personal systems



PC based solutions

Networking products

And a lot of other stuff

~1600 employees worldwide

~450 R&D engineers

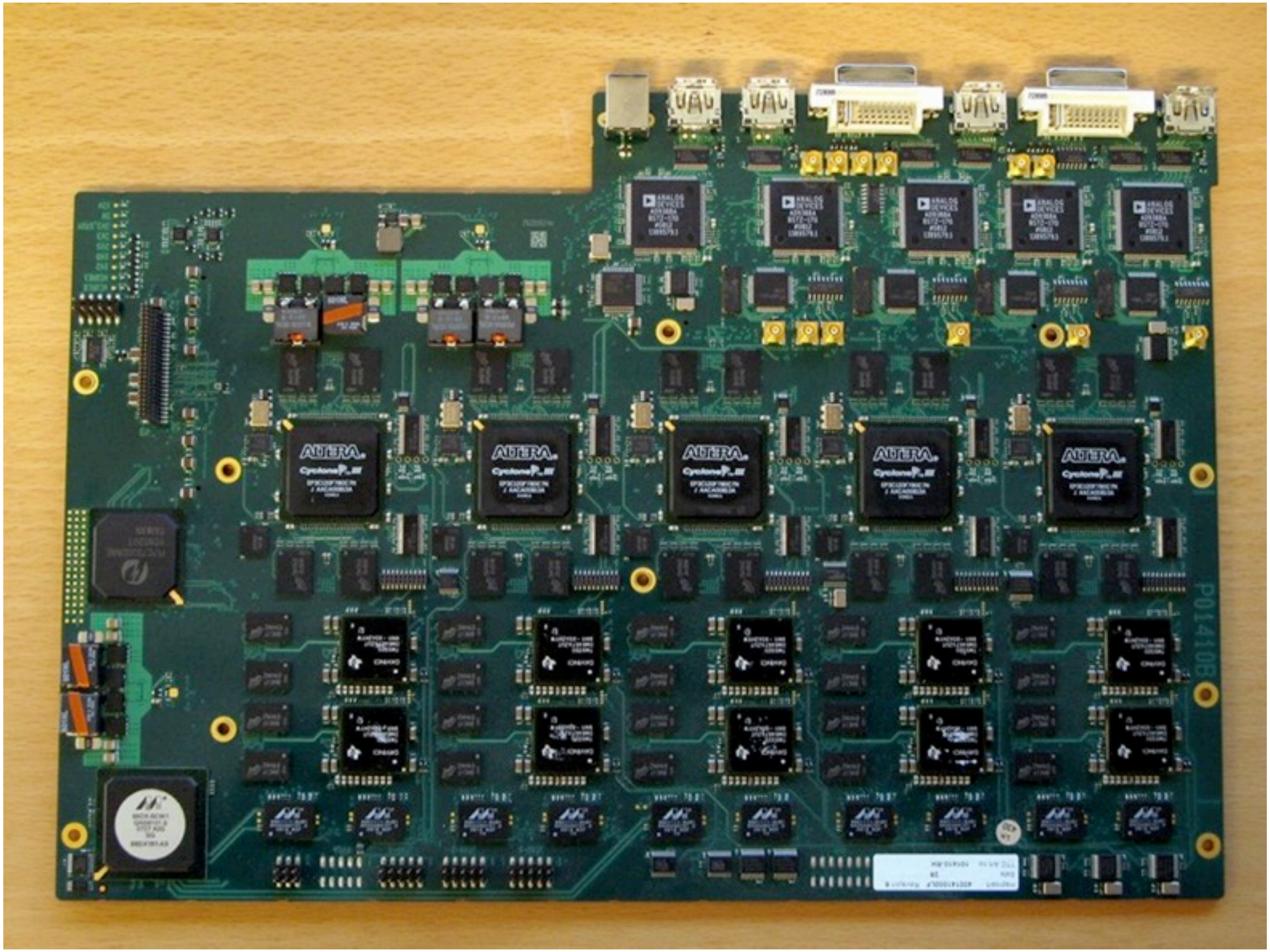
~250 write code every day

C, C++, Python, Java, C#, VHDL, Ruby, Scala

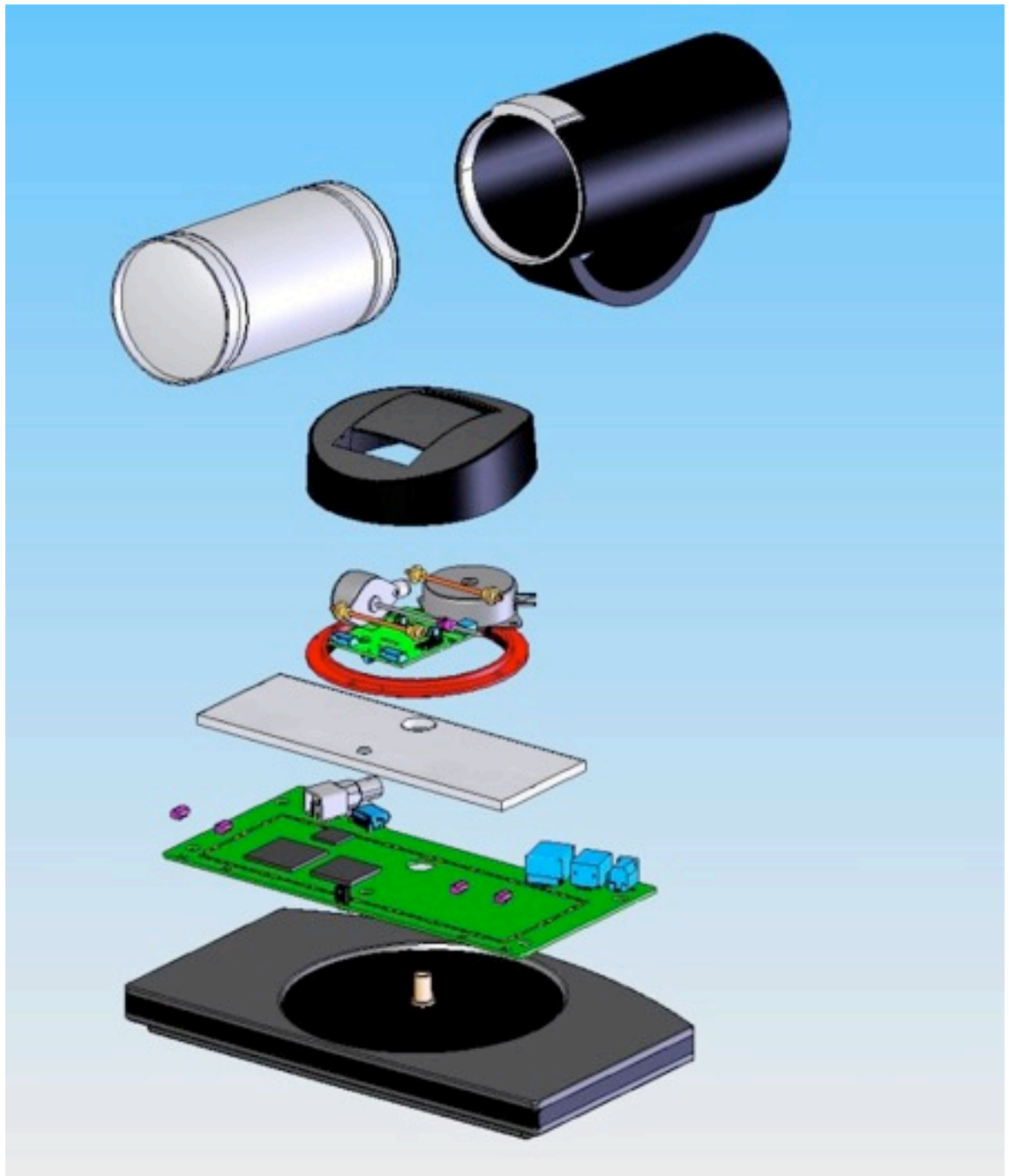
most of us work with software developement

but we also do...

Electronics / Hardware



Mechanics

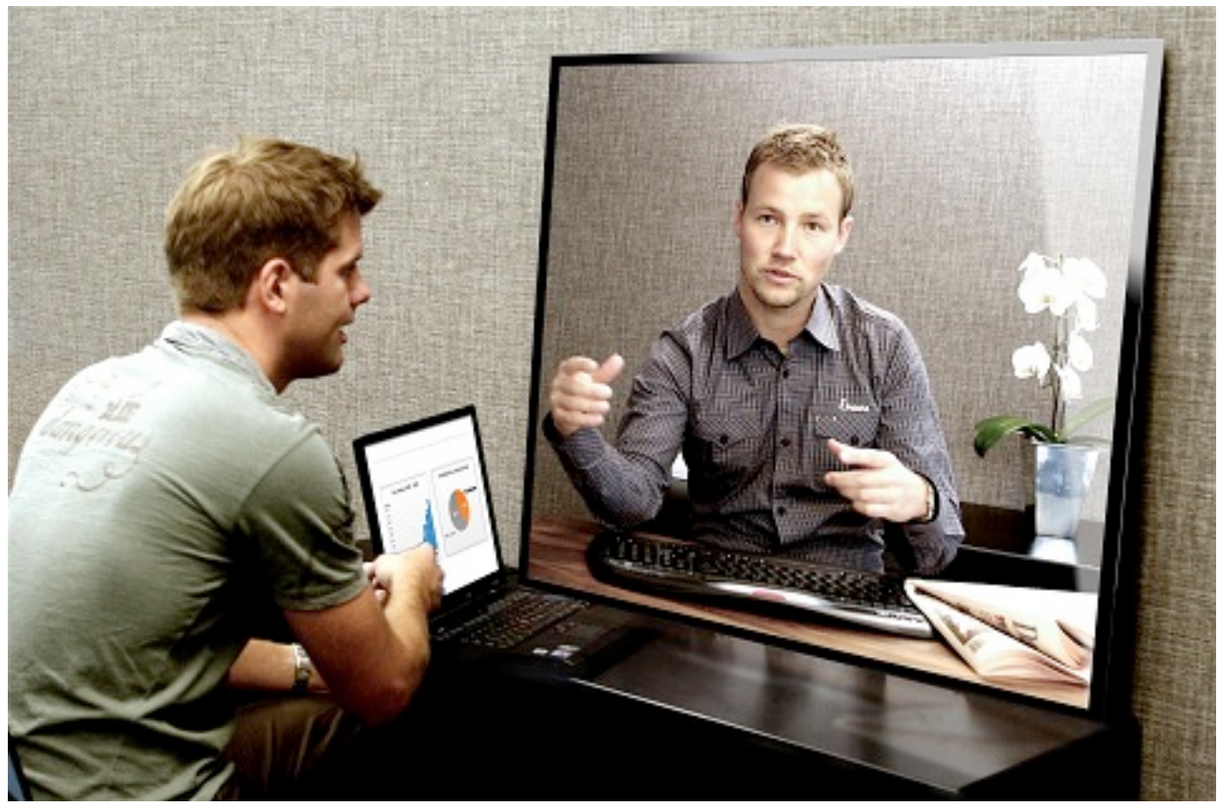


Industrial Design

1992

2007

Looking into



the future

A case study:

TANDBERG Codec C90 - “The Saturn Project”

How did we do it?

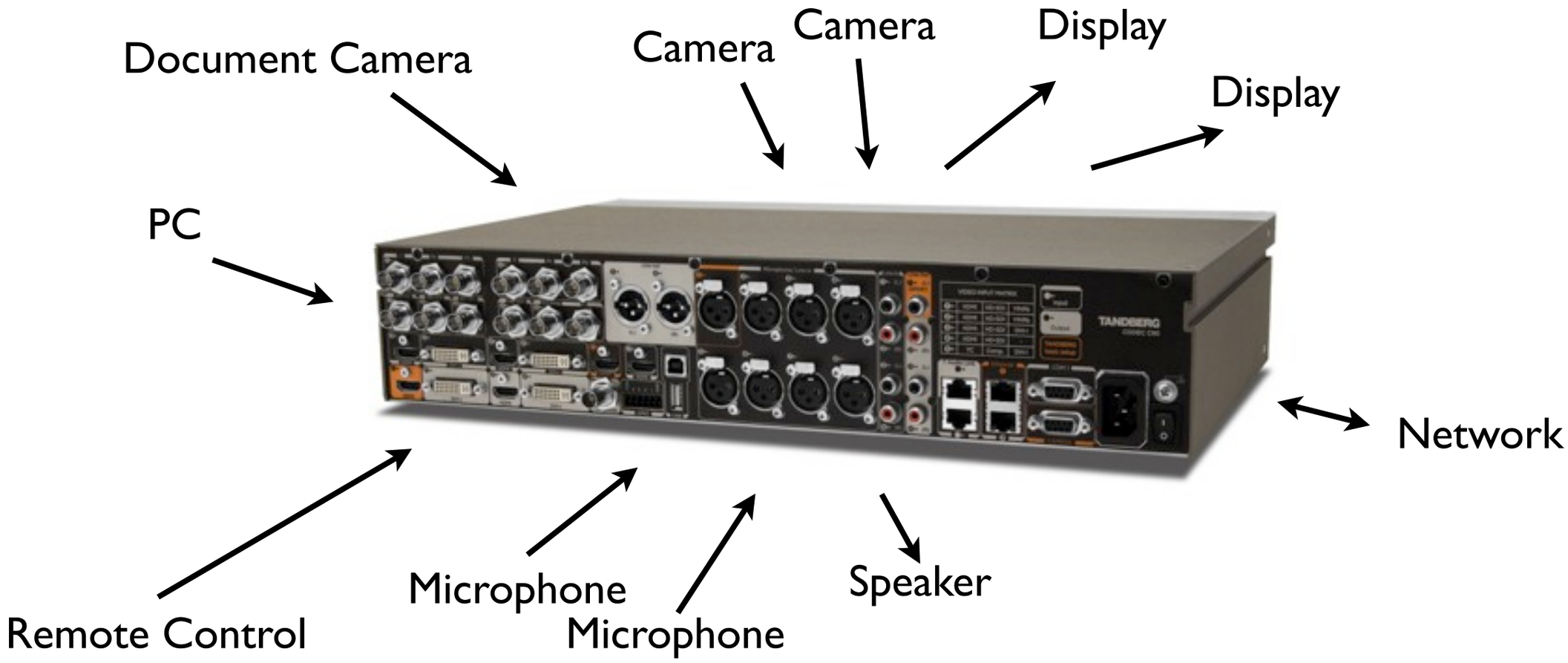
Disclaimer:

The following description does not show how projects in Tandberg are typically developed, it is just an example of how a particular project actually did it. We think about every project, product and team as something unique, thus it does not make sense to create a particular procedure to follow.

Indeed, when it comes to product development, TANDBERG is "allergic" to corporate procedures. It is "unthinkable" that anyone outside a project or a team should impose a certain way of doing things, so we can not say "This is the way we do it", but you may look at a particular project and say "This is the way we did it".

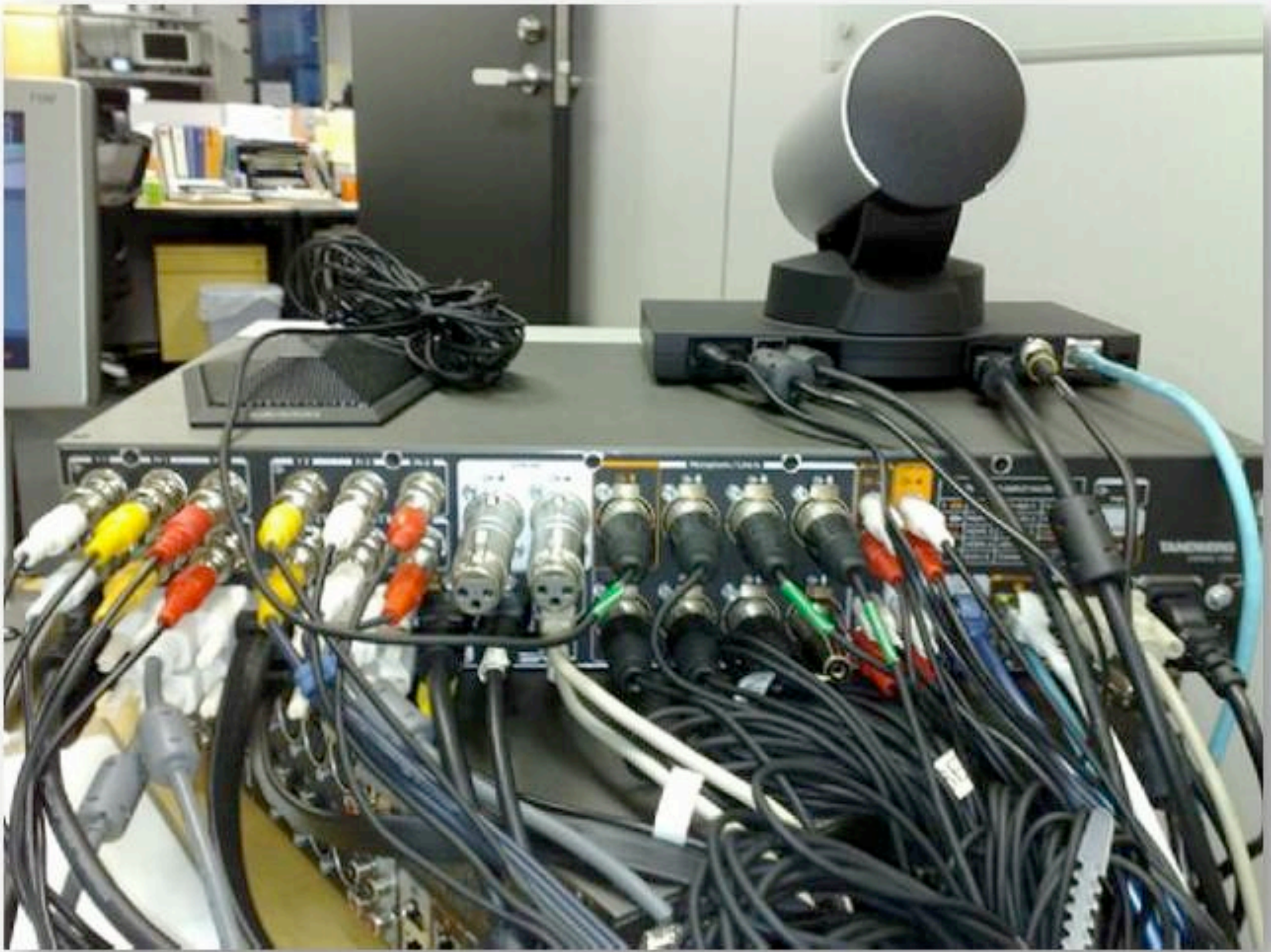
C90 video (1:19)

http://www.tandberg.com/media/index.jsp?id=1312



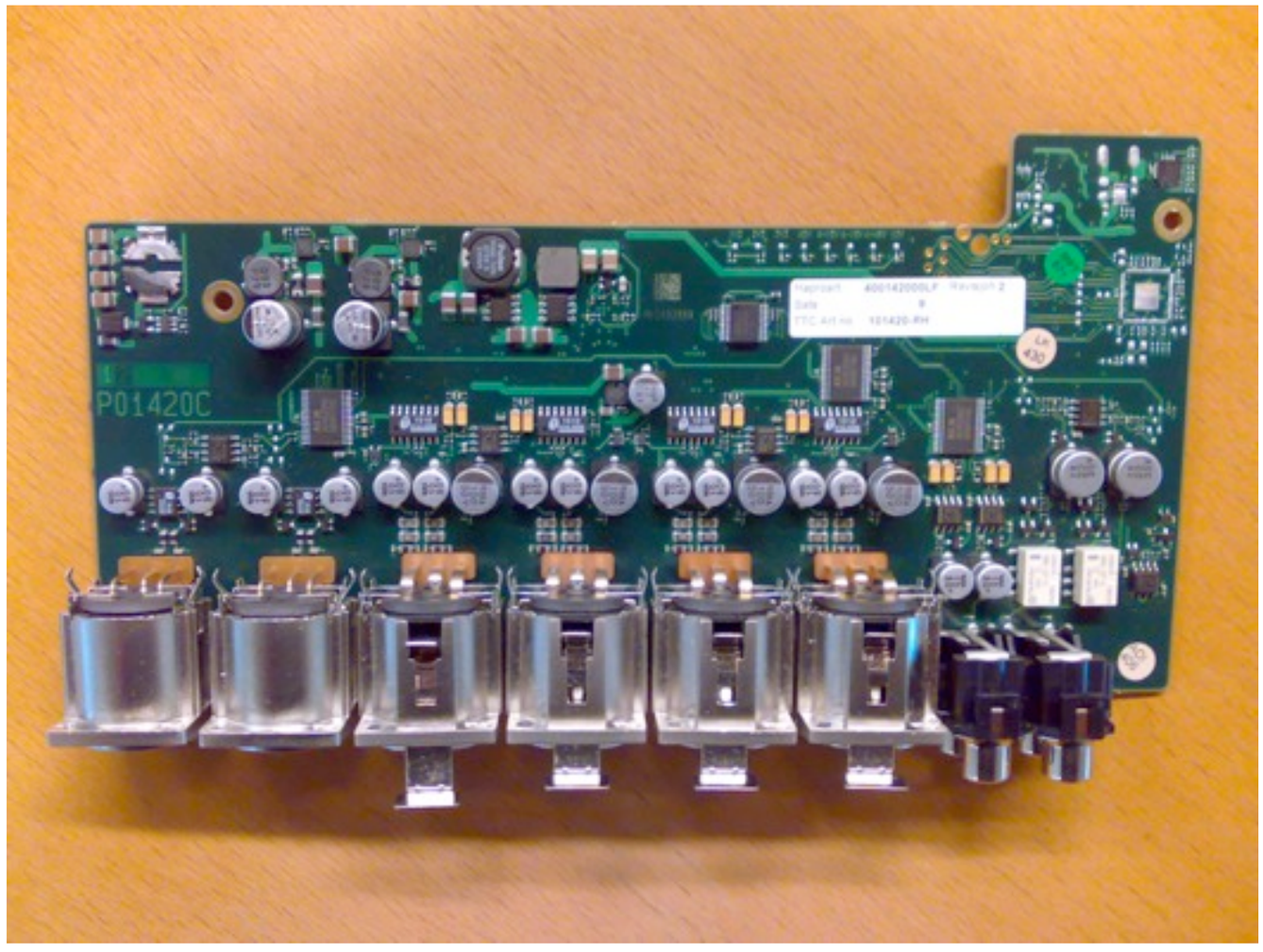
C90 Features:

- realtime H.264 encoding/decoding
- full HD 1080p30, (4+4) concurrent streams
- 12 high definition video sources
- 8 high quality audio sources
- support for many-to-many communication
- Interoperability through H323 and SIP
- API for integration and remote control



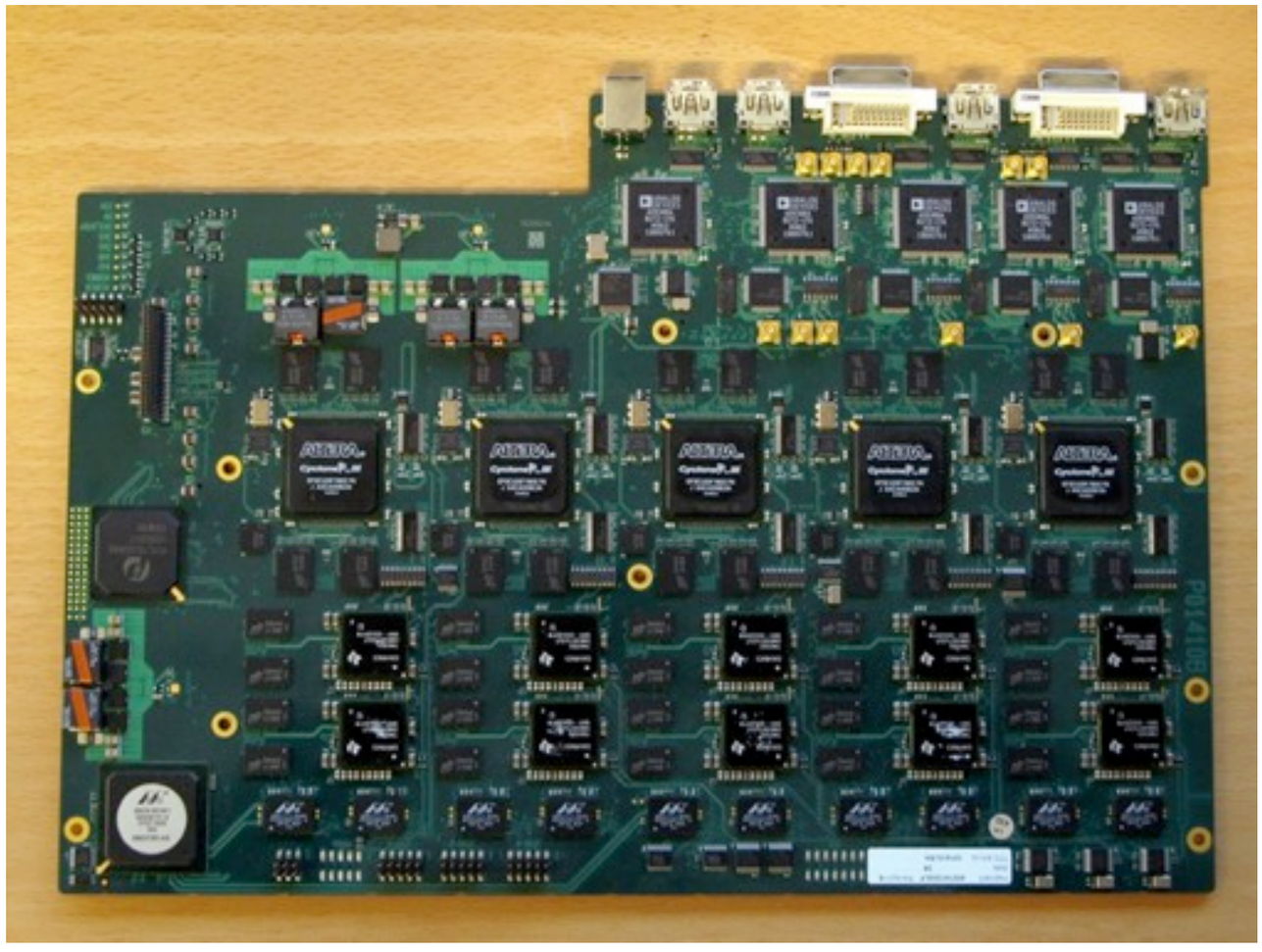
C90 AUDIO EXTENSION BOARD

- analog amplification
- high quality AD and DA converters
- pure electronics, no processor/SW
- 717 components
- 6 layers



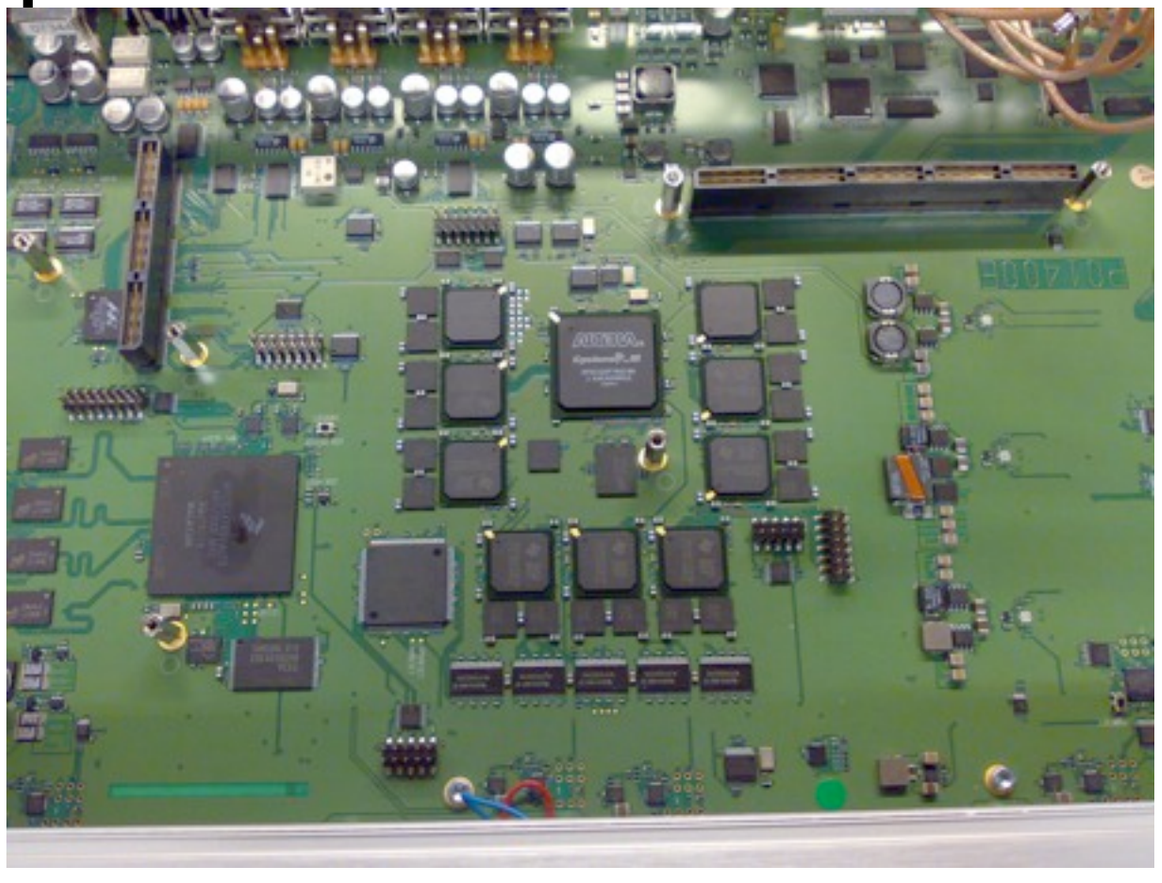
C90 VIDEO BOARD

- 10 Da Vinci DM6467 for video compression/ decompresion(1 ARM, 1 dsp, 2 coprocessors),
- 5 Altera Cyclone III 120 for video scaling & composing(Nios II softcore 50 MHz)
- 15 Gbps video backplane
- 3.8 GByte DDR2 RAM
- 128 mbit x5 SDRAM
- 6097 components
- 30520 pins
- 22 layers
- 6490 nets



C90 MAIN BOARD

- 1 Altera Cyclone III 120 for Audio switching (Nios II softcore 50 MHz)
- 9 TI 6727, audio dsp for echo control, compression, decompression, +++
- PowerPC 8347, main processor, application software, networking, user interface
- 3543 components / 15659 pins
- 16 layers
- 3264 nets



C90 - from a geek point of view

- 10000+ components
- 44 (6+22+16) layers
- 56 processor cores
- several million lines of code (C and C++)

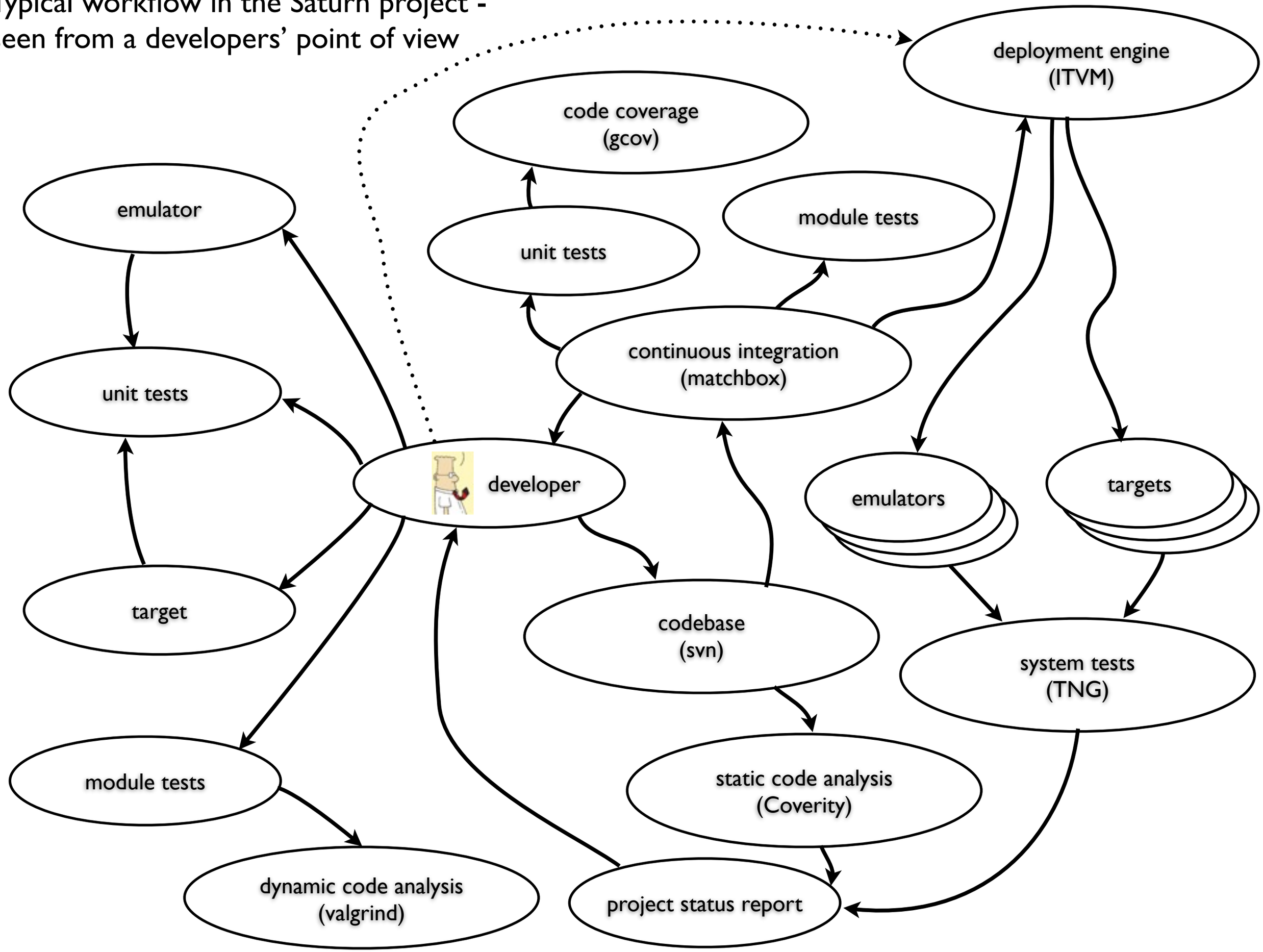
TANDBERG Codec C90

- Developed at Lysaker
- Started spring 2007
- First HW prototype arrived summer 2008
- Released late 2008 (~20 months of development)
- 2-3 people working with mechanics/design
- 4-5 people working with electronics/hardware
- 5-6 people working with FPGA development
- 40-50 people working with software development
- 4 people working with test developers
- 1 person working with approvals

Development Practices in the Saturn project

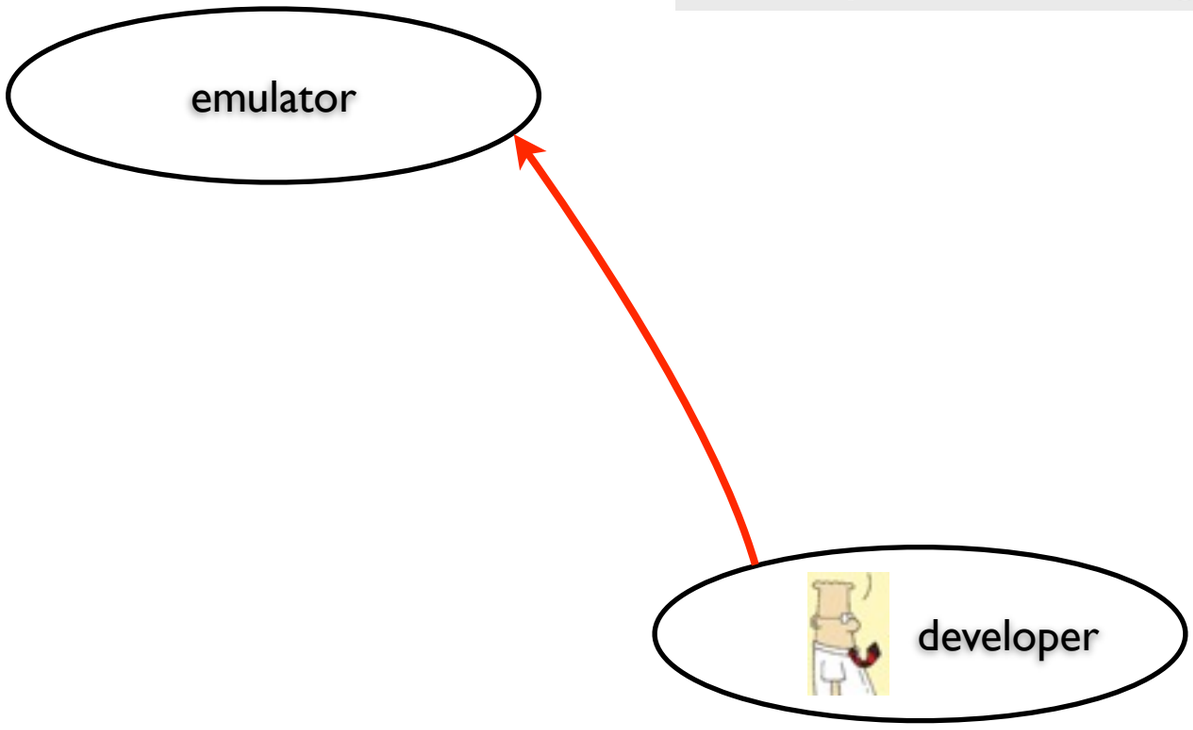
- Continuous planning
- Always attack high risks first
- Heavy focus on effective feedback mechanisms
- Visualization of actual status throughout project
- Parallel development
- Iterations and time-boxing
- Daily scrum of scrums
- Weekly rendezvous meetings
- Early and many prototypes

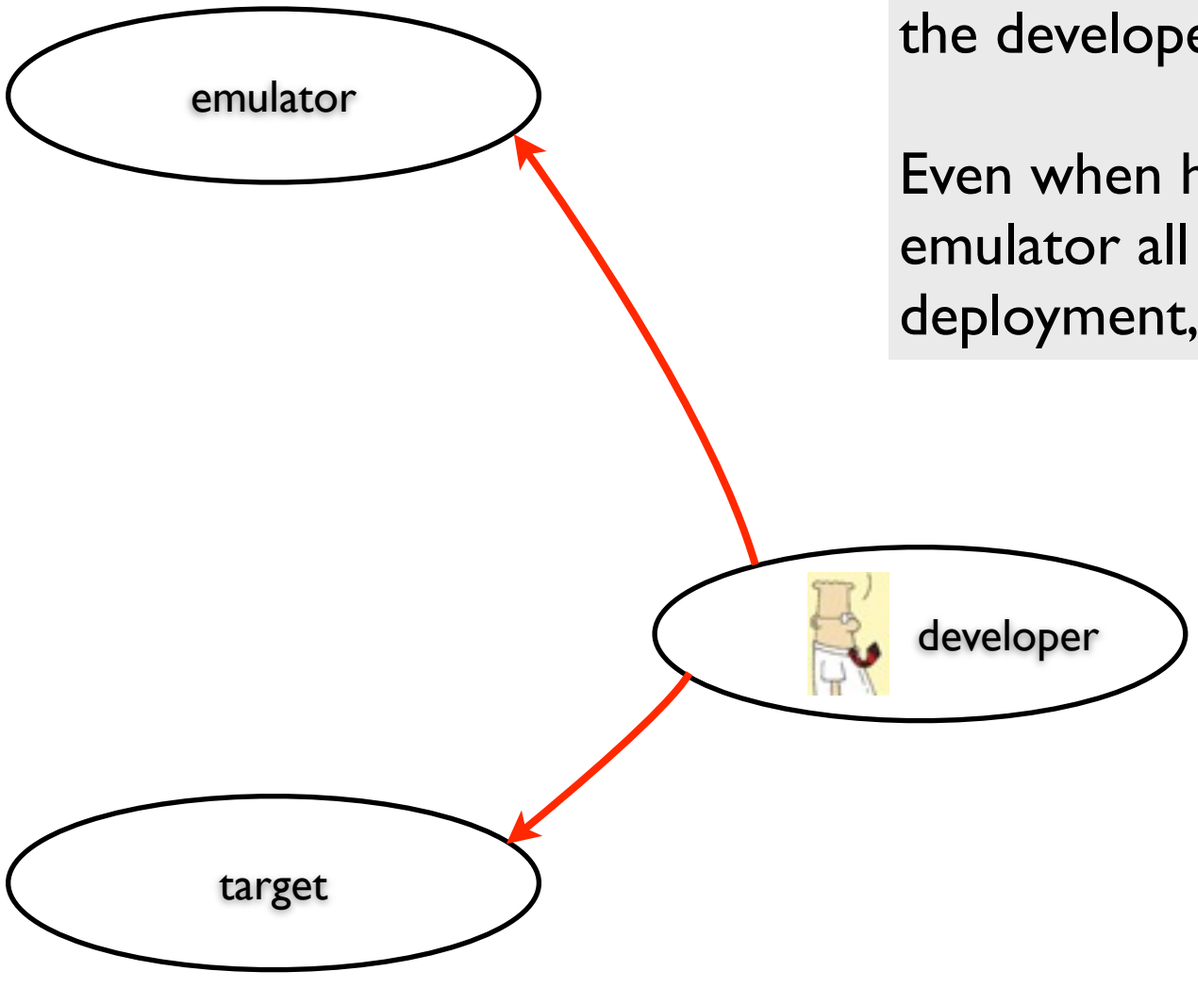
Typical workflow in the Saturn project -
seen from a developers' point of view



Software development in the Saturn project as seen from a developers point of view.

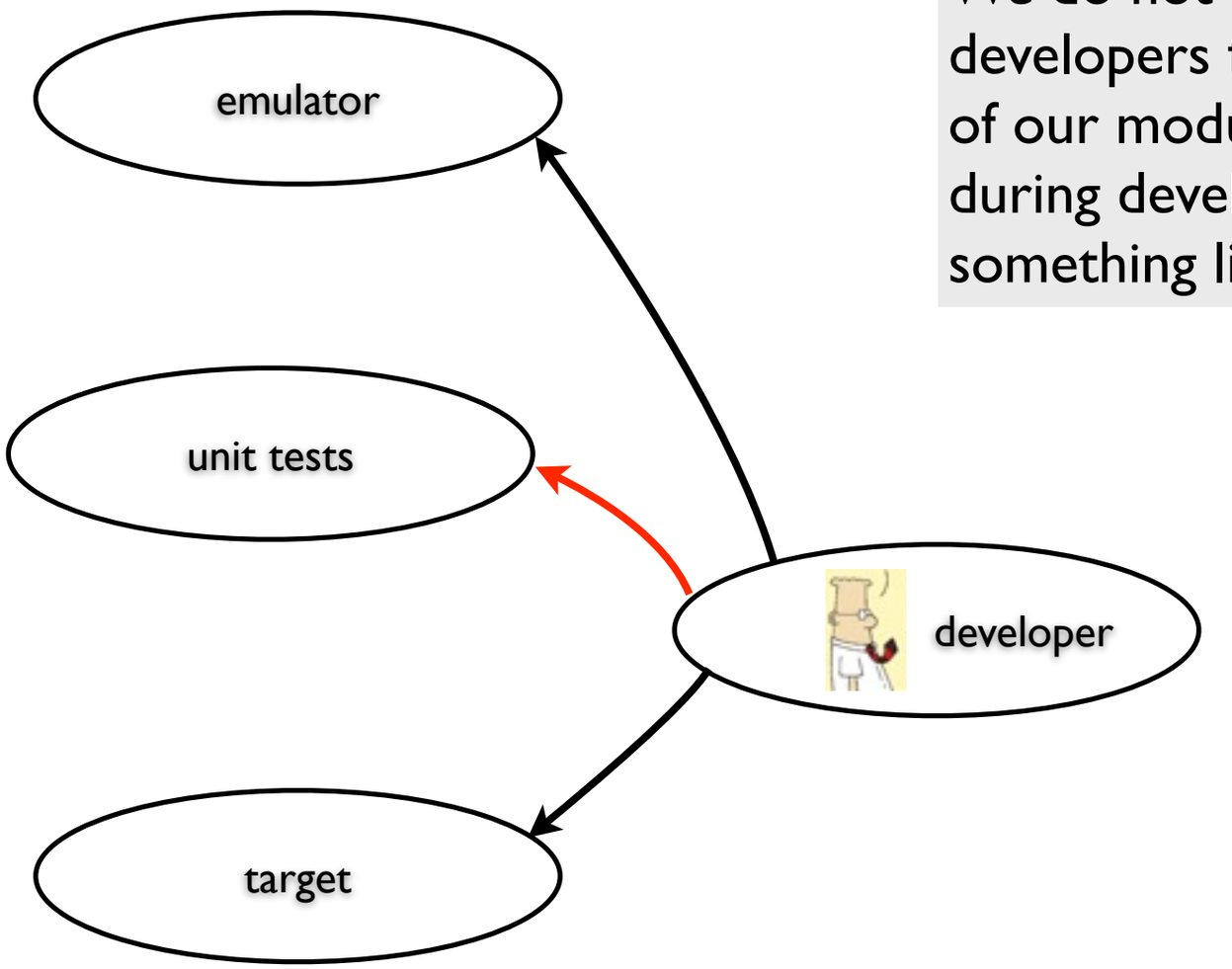
Software development started long before we had any hardware to play with. We spent a lot of energy to develop an emulating environment for this project.





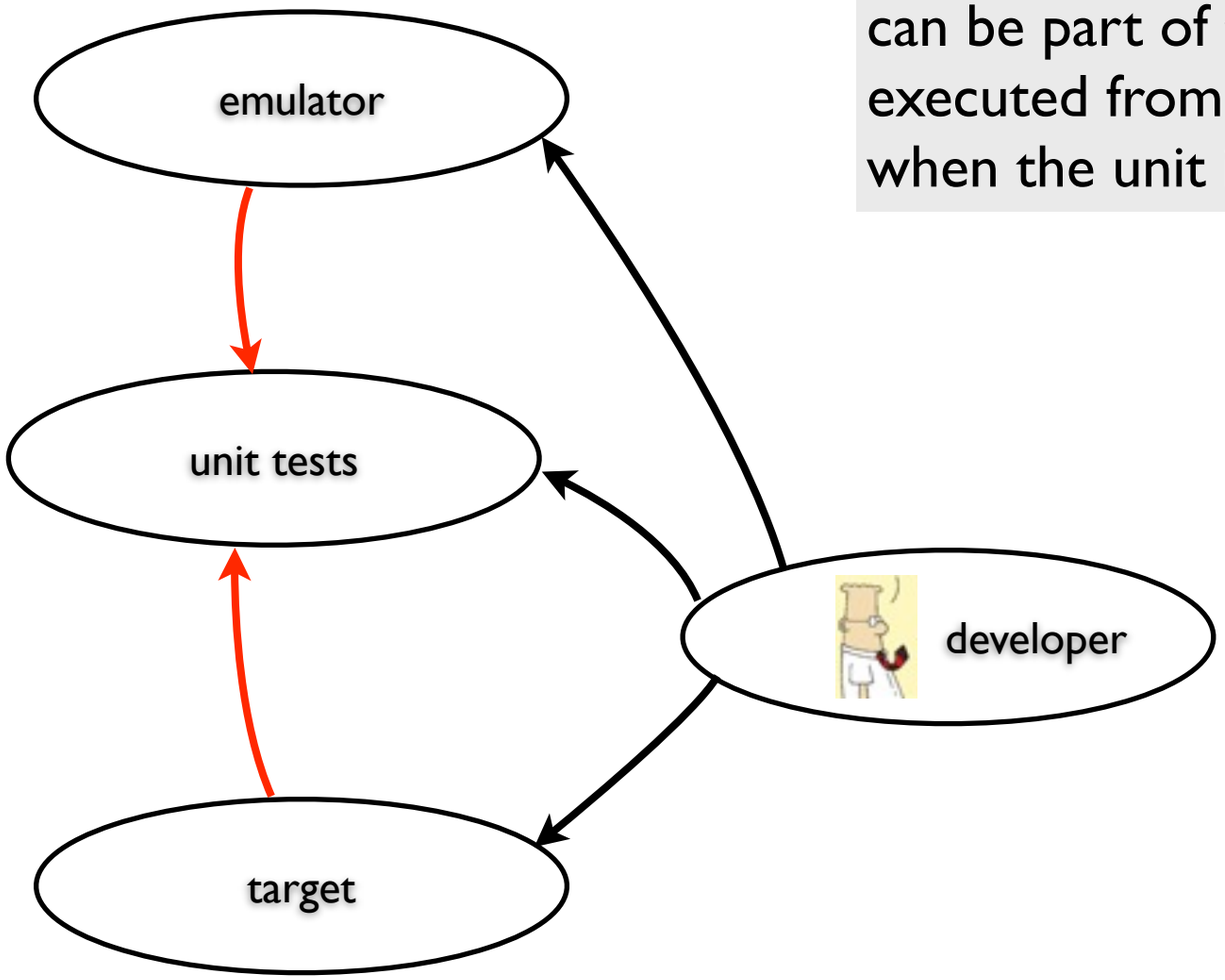
The emulator was/is nearly complete. Apart from some CPU intensive and HW specific parts of the application, most of the code can run quite smoothly in an emulator running on the developers machine.

Even when hardware became available developers still use the emulator all the time because of convenience (speed of deployment, proper debuggers and other development tools).

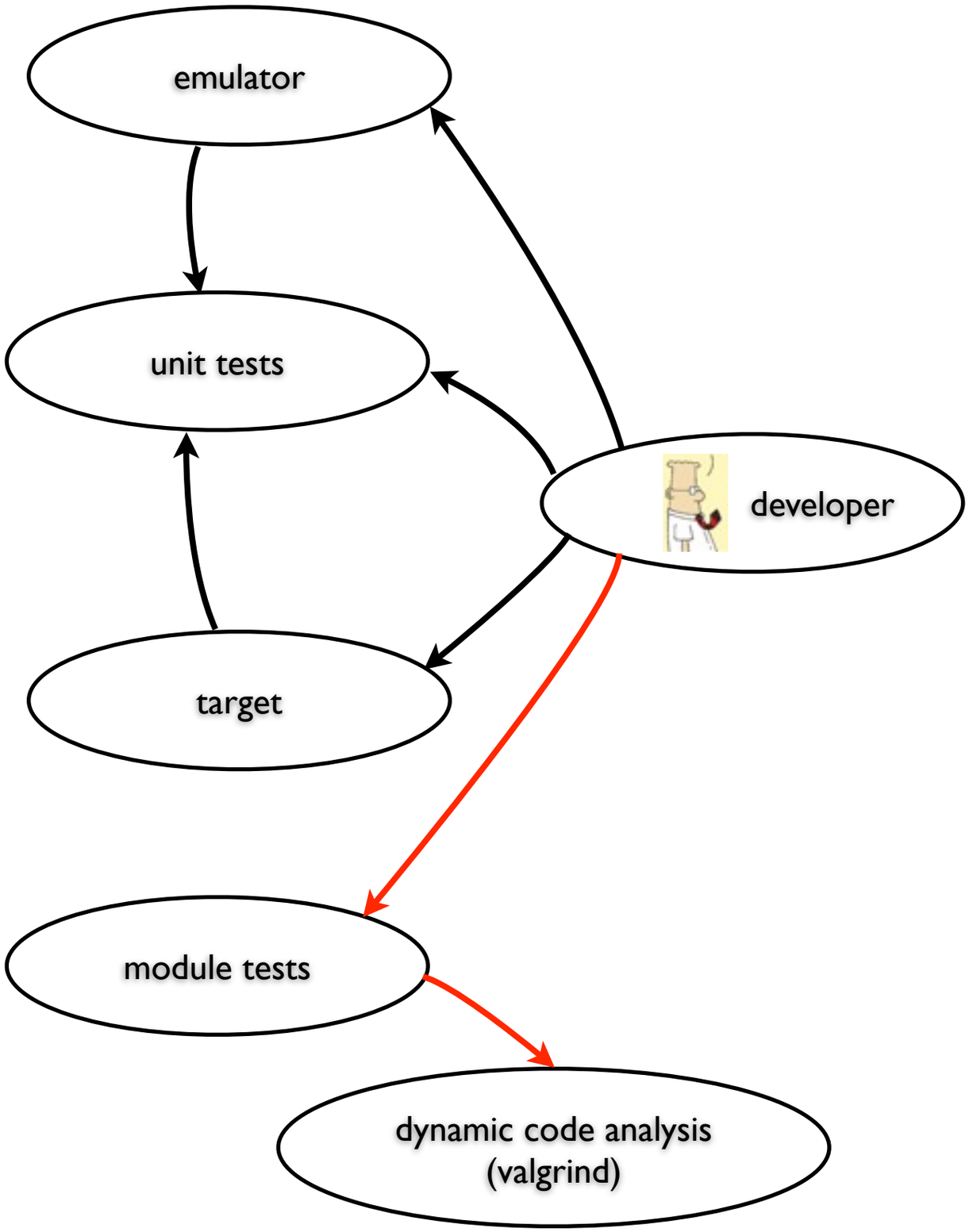


We do not enforce unit testing or TDD, but a lot of developers find it to be a productive practice so most of our modules have unit tests that can be executed during development. Currently the Saturn code has something like 20000+ unit tests.

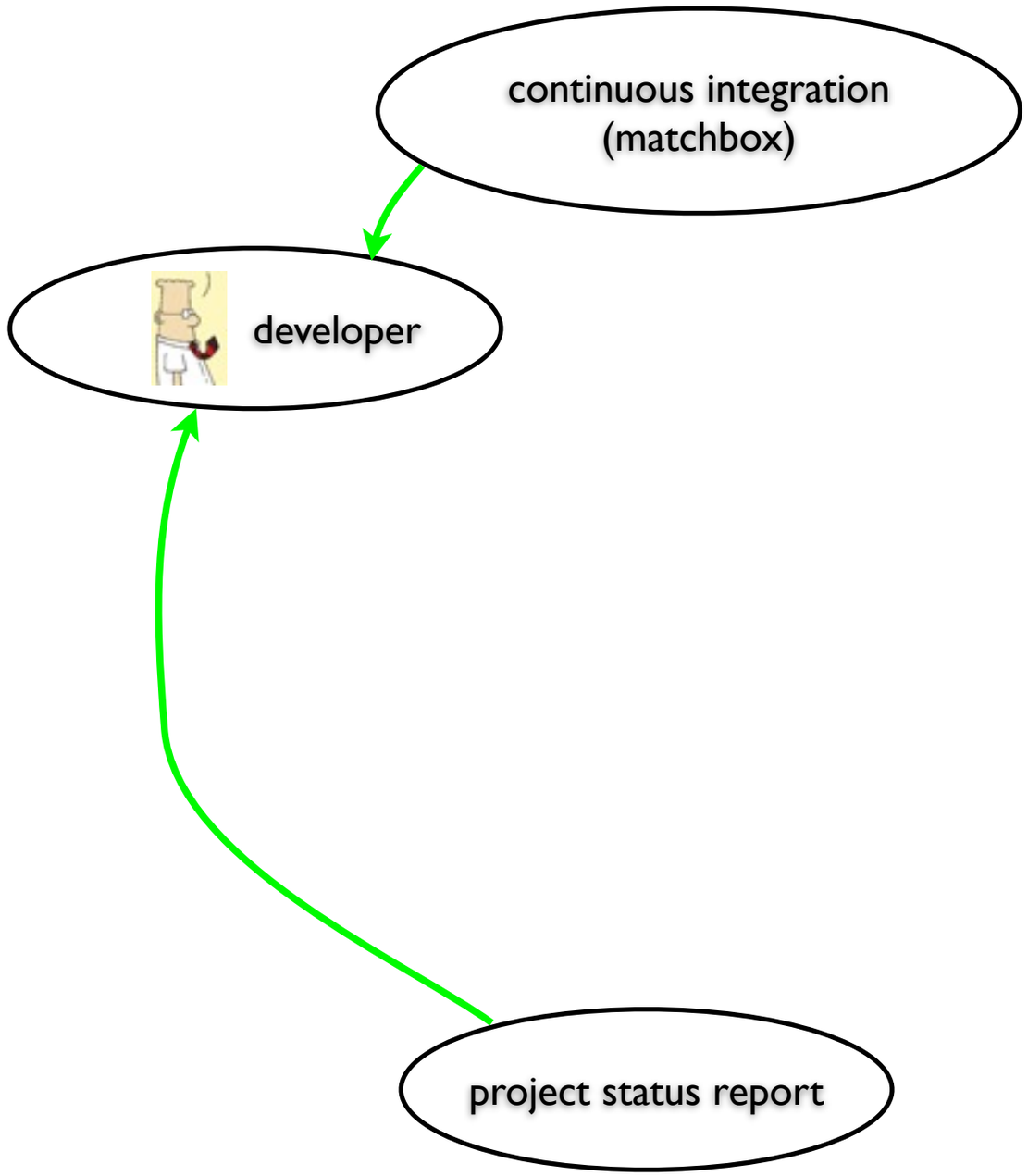
A very nice feature of our unit test framework is that it can also be executed on target and (of course) in the emulator. Actually, the unit tests can be part of the release build so they can be executed from an administrative console even when the unit is in the field.



We also have a module test framework. The module tests are also used to execute dynamic code analysis (valgrind).

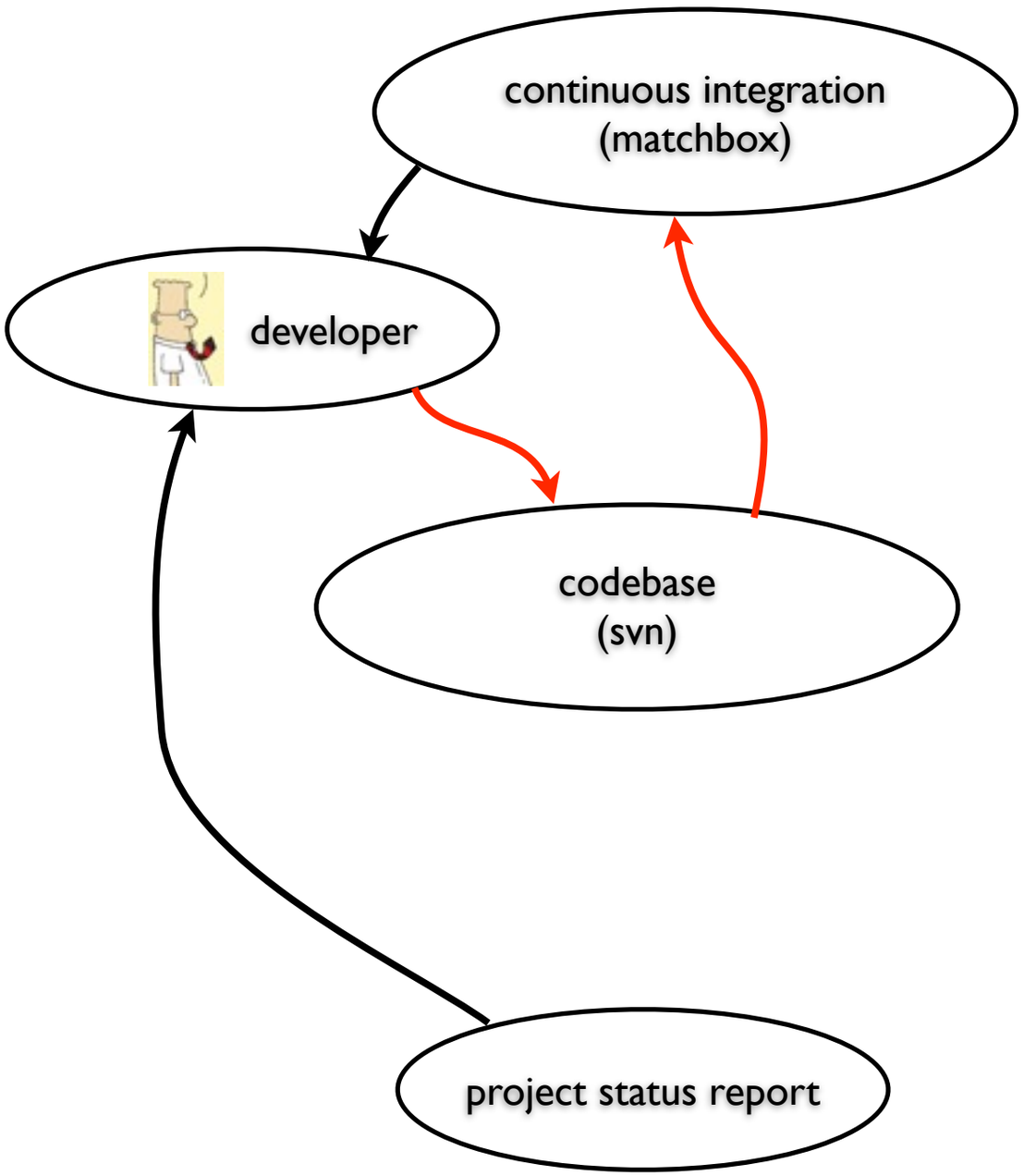


Before checking in changes, the developer checks the continuous integration server and the current project status report before checking into the source control system.



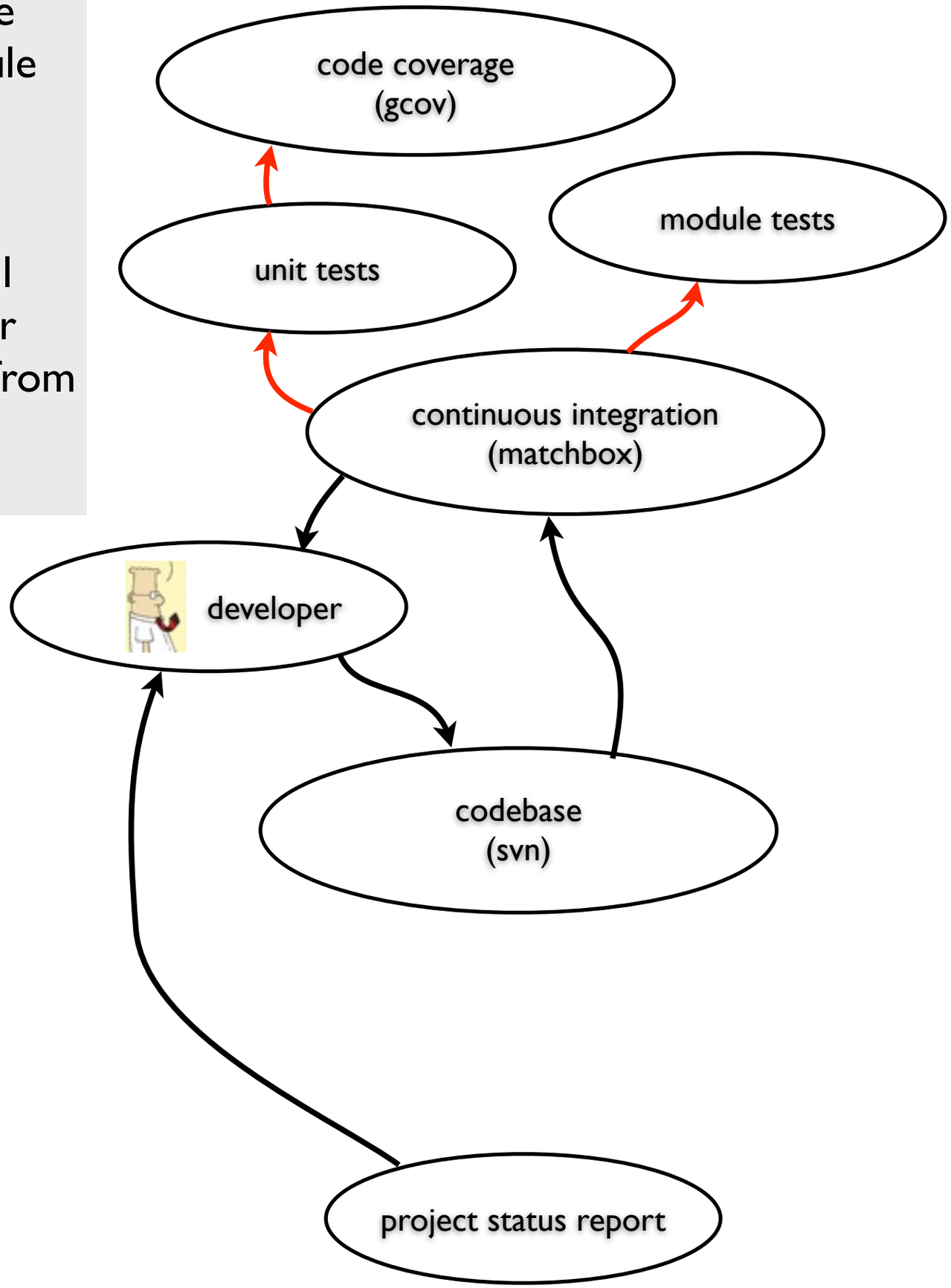
Any change applied to the shared codebase is immediately picked up by matchbox, our continuous integration system. Matchbox builds all products depending on that change, perhaps as many as ten products using different compilers targeting various processors.

All compilation is with high warning levels (-Wall, -Wextra) and warnings are treated as errors (-Werror)



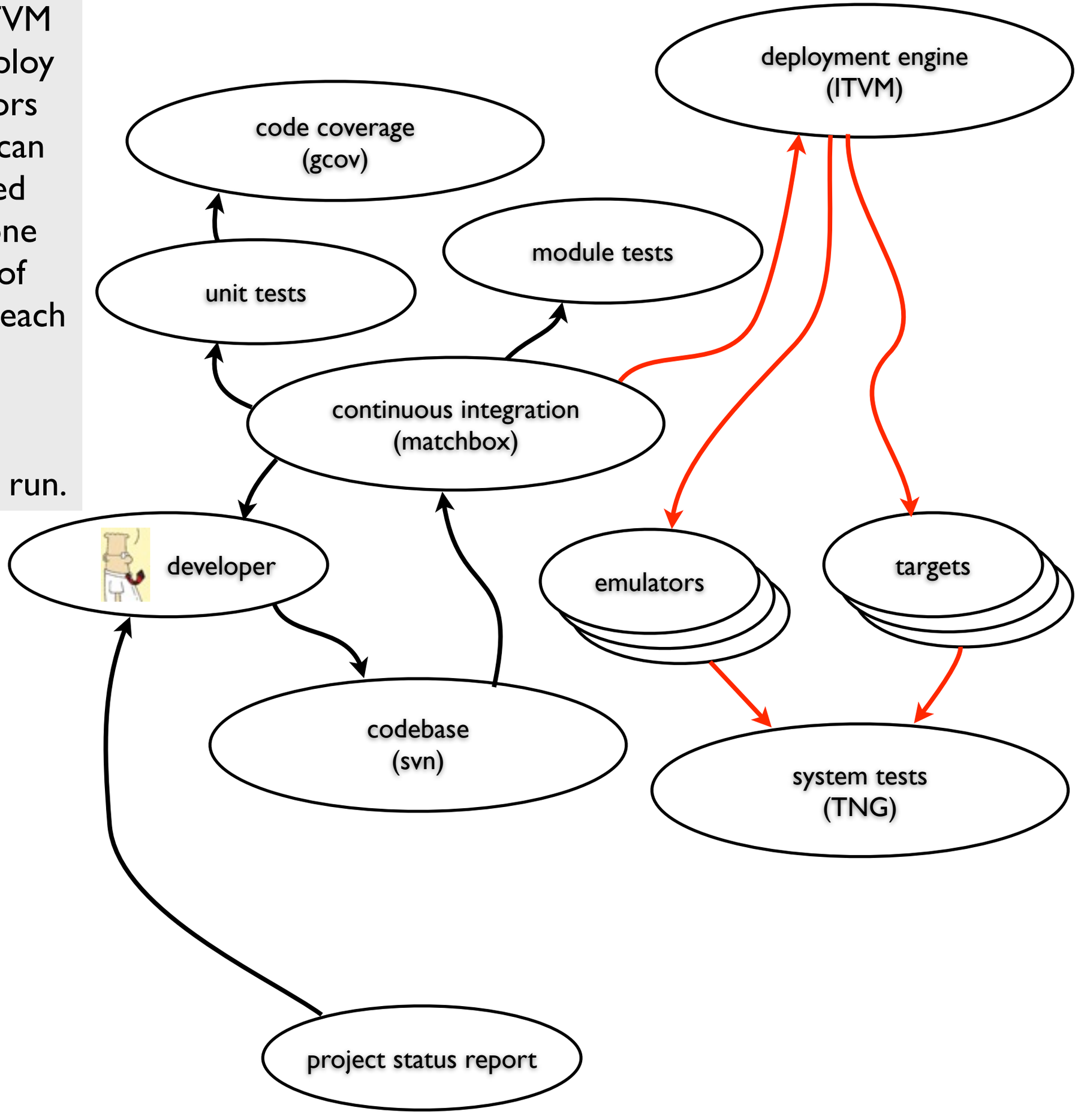
Matchbox will also run all unit tests, measure code coverage and run module tests for all products affected by the change.

We have many powerful build servers. Developer typically gets feedback from matchbox within a few minutes.



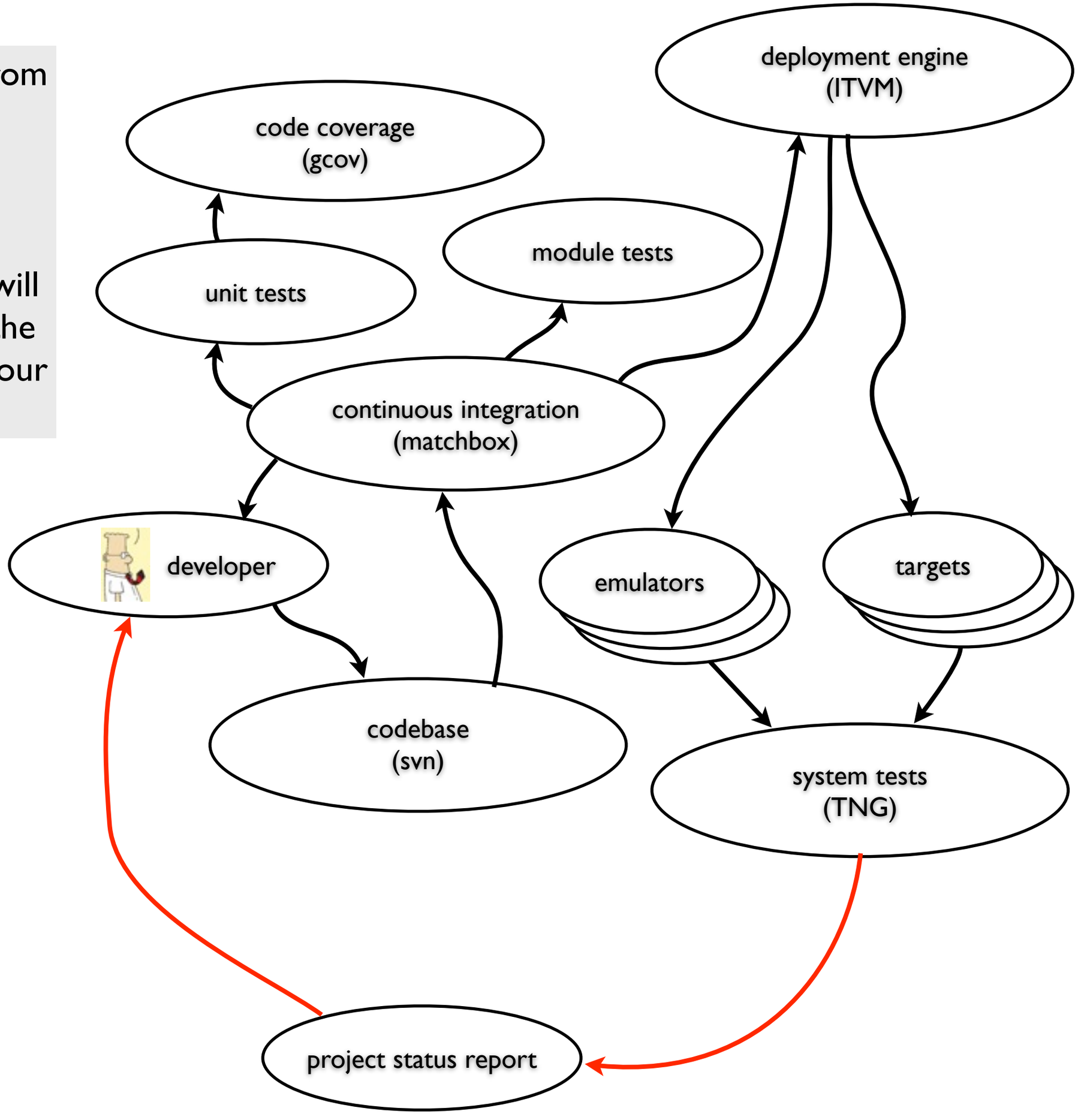
If matchbox is happy, then ITVM take the built images and deploy them onto a farm of emulators and/or targets so that TNG can start running loads of scripted test scenarios both testing one particular target or a group of targets communicating with each other.

Some of the tests runs fairly quickly other takes hours to run.

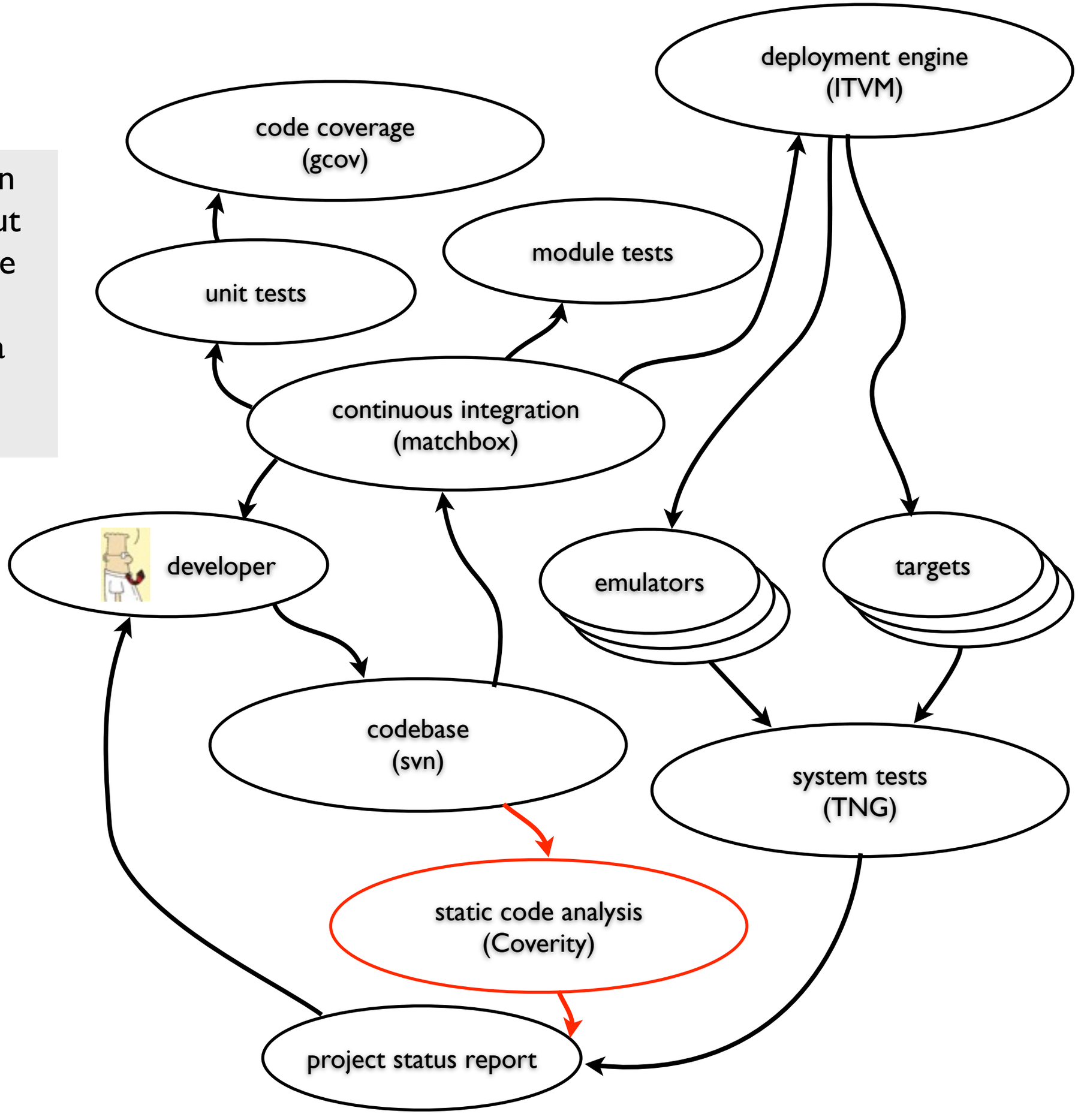


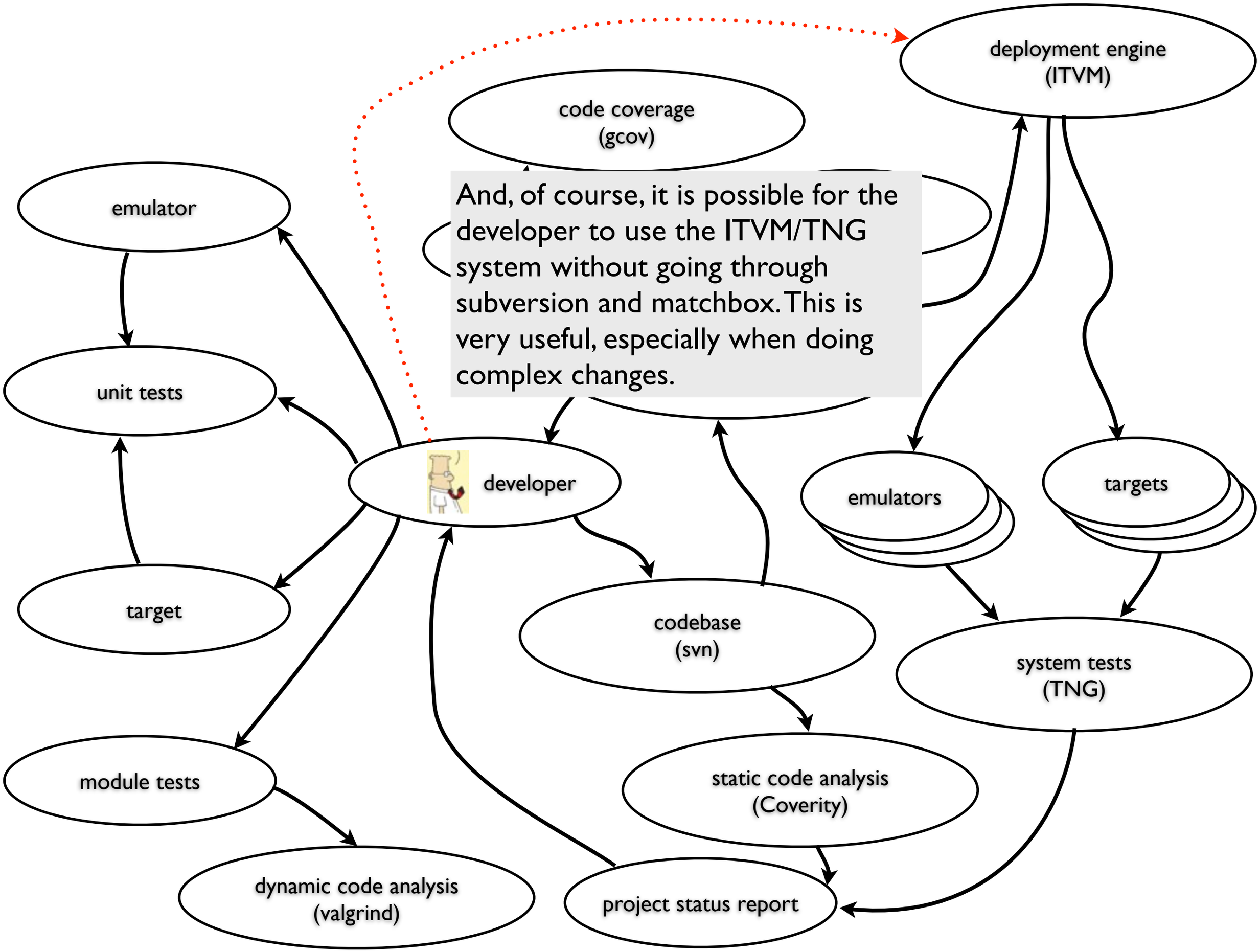
As soon as the results from various system tests are ready, the project status report gets updated.

Typically, the developer will see the results from all the system tests within an hour or two.



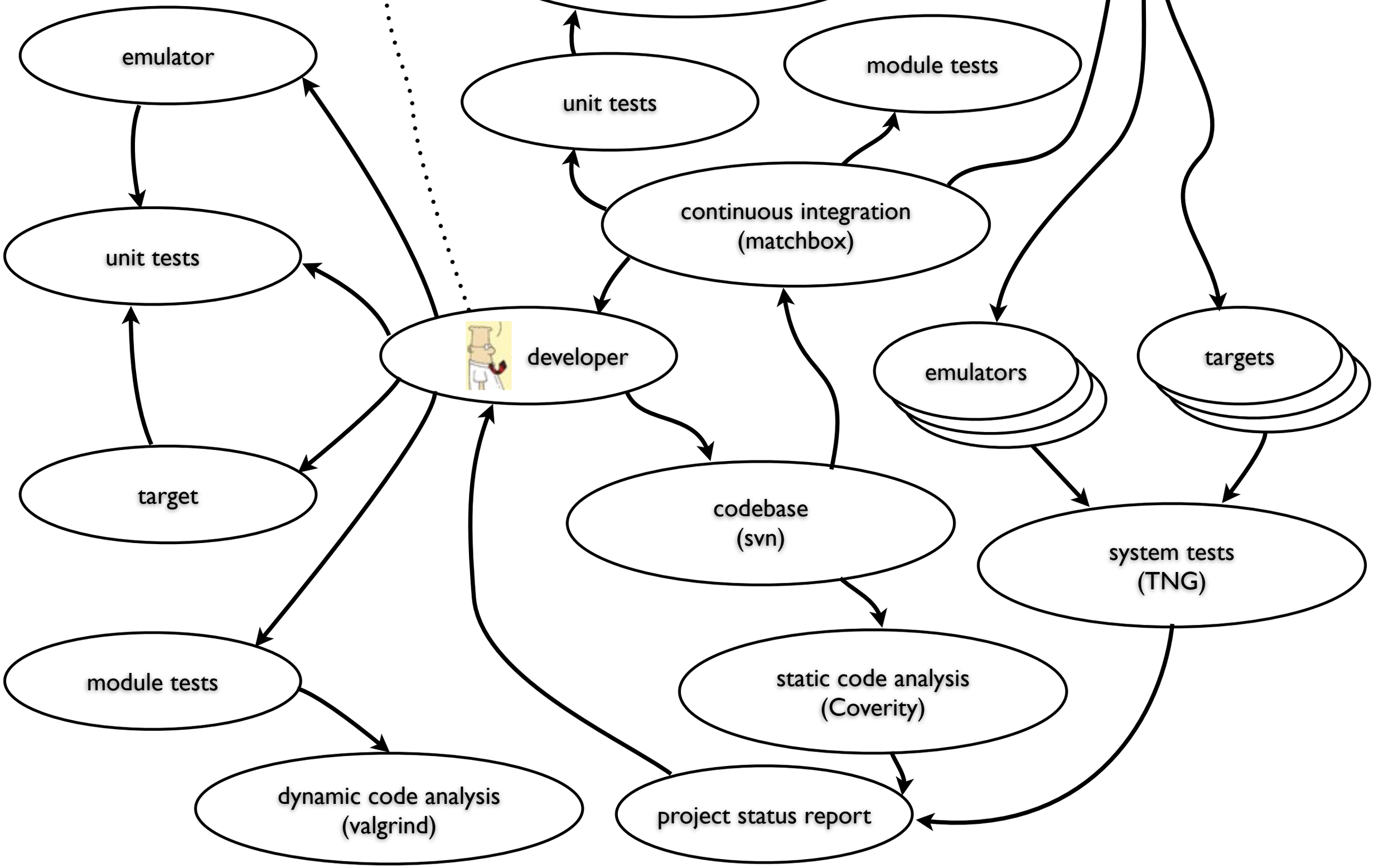
Coverity is also working on the codebase. It takes about 12 hours to execute, so the project status report is updated about two times a day with the result of the static code analysis.





And, of course, it is possible for the developer to use the ITVM/TNG system without going through subversion and matchbox. This is very useful, especially when doing complex changes.

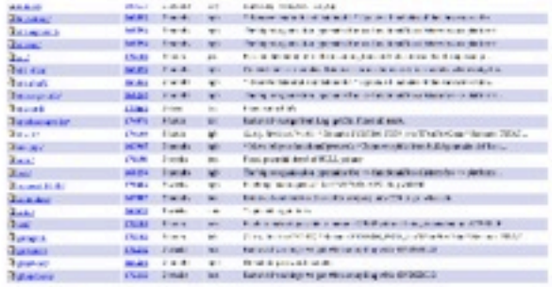
This workflow represents the feedback mechanism that developers rely on when applying changes to the codebase. Bad changes are reported back within minutes and hours - not days and weeks.



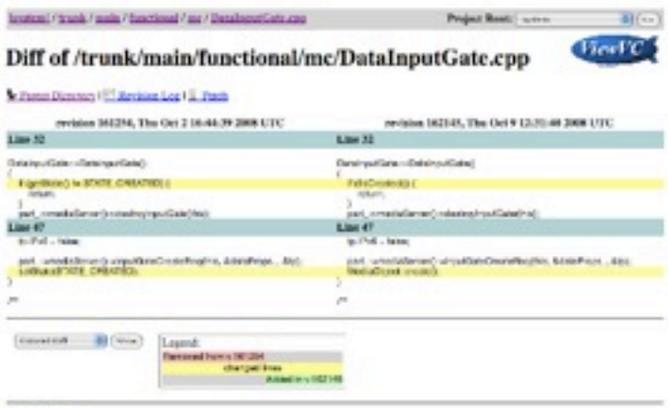
Other aspects of the Saturn project:

- IRC channels
- free choice of development platform
- software repository (svn)
- warning free compilation!
- proper training (C++, C, Testing, Professionalism, ...)
- lot of energy spent on software emulator of actual hardware
- project leader is also the configuration manager / build master
- Teams: GUI, App, Protocol, Video, Audio, FPGA, Platform, QA, Support
- static code analysis (Coverity)
- dynamic code analysis (valgrind)
- build system (genmake2, inhouse python)
- automatic deployment engine (ITVM, inhouse C#)
- automatic system testing (TNG, inhouse python)
- unit test framework (unittest, inhouse C and C++)
- module test framework (inhouse C++)
- code coverage (gcov)
- continuous integration system (matchbox, inhouse python)

Example of visual feedback (HTML pages used by all/most developers)

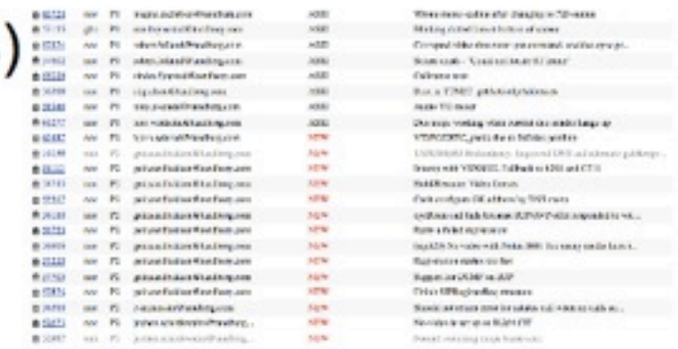


viewvc



diff from viewvc

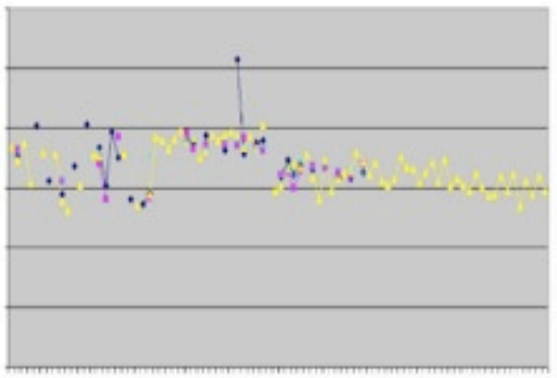
audio delay trend



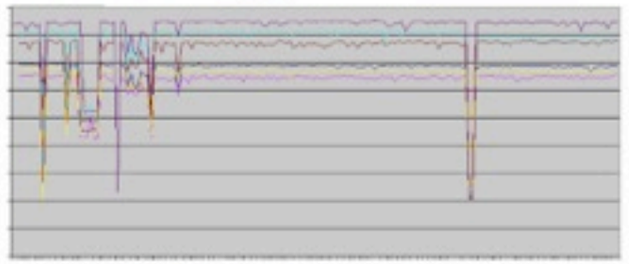
bugzilla

irc channel

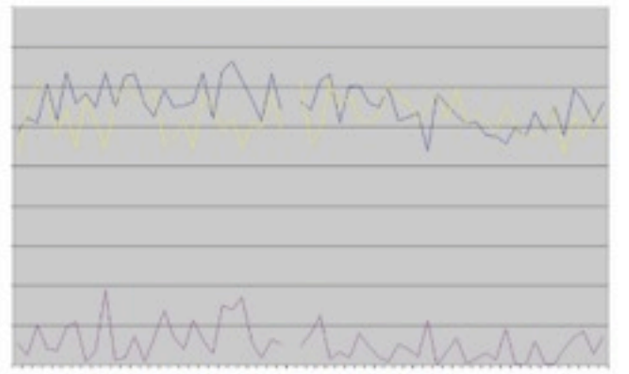
continuous integration



H.264 delay trend

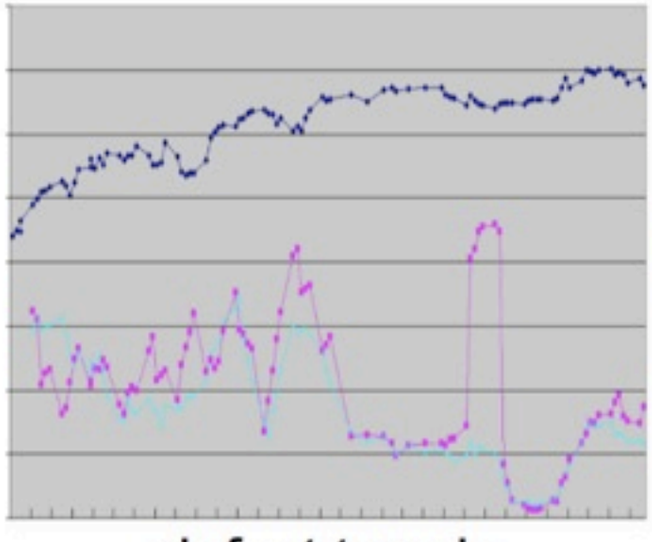


PESQ trend

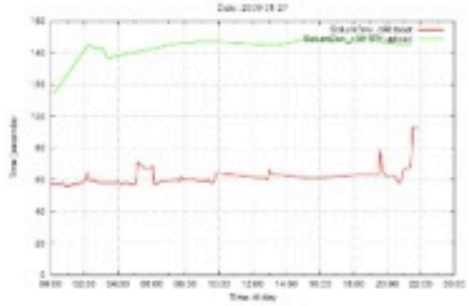


lipsync trend

system tests



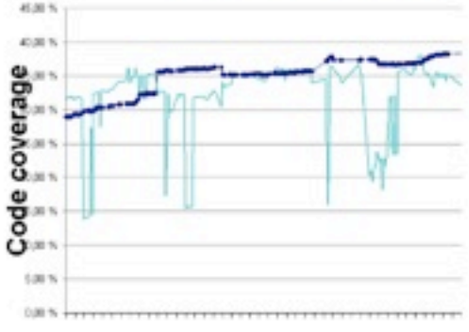
defect trends



endpoint timing

coverity

QA Status



Software development is a continuous learning process and a cooperative game of communication between professionals. Software development is about repeating cycles of preparing, changing, observing, reflecting, and learning.

Software development is a continuous learning process and a cooperative game of communication between professionals. Software development is about repeating cycles of preparing, changing, observing, reflecting, and learning.

Software development is a continuous learning process and a cooperative game of communication between professionals. Software development is about repeating cycles of preparing, changing, observing, reflecting, and learning.

Software development is a continuous learning process and a cooperative game of communication between professionals. Software development is about repeating cycles of preparing, changing, observing, reflecting, and learning.

Software development is a continuous learning process and a cooperative game of communication between professionals. Software development is about repeating cycles of preparing, changing, observing, reflecting, and learning.

Software development is a continuous learning process and a cooperative game of communication between professionals. Software development is about repeating cycles of preparing, changing, observing, reflecting, and learning.

Software development is a continuous learning process and a cooperative game of communication between professionals. Software development is about repeating cycles of preparing, changing, observing, reflecting, and learning.

So... for this project...

When we saw an opportunity to improve feedback mechanisms within the project we did. When we found things that reduced communication within the project we tried to get rid of it.

Especially in crunch mode, we focused on making sure that the feedback mechanisms was working. Automated feedback is about giving developers confidence when making changes - when confidence is lost, everything is lost...

A professional is defined by the way they act under pressure. Do you rush? Do you drop your disciplines? Or do you stay calm under fire? (Uncle Bob, 2009)

Product Development in TANDBERG

Product Development in TANDBERG

- No corporate standards or procedures

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and "skunk work" projects appreciated

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and "skunk work" projects appreciated
- Spend very little energy on things that are not essential

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg
- Autonomous organization

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg
- Autonomous organization
- We hire and keep exceptional people

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg
- Autonomous organization
- We hire and keep exceptional people
- Communication is a key skill for all our engineers

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg
- Autonomous organization
- We hire and keep exceptional people
- Communication is a key skill for all our engineers
- We are fast and “sloppy”

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg
- Autonomous organization
- We hire and keep exceptional people
- Communication is a key skill for all our engineers
- We are fast and “sloppy”
- We release early and we release often

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg
- Autonomous organization
- We hire and keep exceptional people
- Communication is a key skill for all our engineers
- We are fast and “sloppy”
- We release early and we release often
- Fun gives profit (not: profit, then fun)

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg
- Autonomous organization
- We hire and keep exceptional people
- Communication is a key skill for all our engineers
- We are fast and “sloppy”
- We release early and we release often
- Fun gives profit (not: profit, then fun)
- The company builds on trust

Product Development in TANDBERG

- No corporate standards or procedures
- Little documentation gives effective communication
- Treat engineers as professionals, not as resources
- Slack is embedded, and “skunk work” projects appreciated
- Spend very little energy on things that are not essential
- "Plans are nothing, planning is everything"
- Do not write hours, do not measure project cost
- Delay decisions as much as possible
- To fail is OK, therefore we deliver spectacular stuff
- Doers are very much respected in Tandberg
- Autonomous organization
- We hire and keep exceptional people
- Communication is a key skill for all our engineers
- We are fast and “sloppy”
- We release early and we release often
- Fun gives profit (not: profit, then fun)
- The company builds on trust

We follow principles, not processes!

SPEED AND PRECISION

Simplify - focus - act * Approximately right rather than accurately wrong * Think * Do it right the first time

INTEGRITY AND ENTHUSIASM

Sense of humour * Honesty * High ethical standards * Excitement * Trustworthiness * Loyalty

EXCEED EXPECTATIONS

Personal initiative * Fighting spirit * Go the last mile

FUN AND PROFIT

Maximize long term shareholder value * Pass på penga * One for all, all for one * Energy

TANDBERG FIRST

First in user benefits * Innovative * "Kreativ galskap" * Understanding customer needs

SPEED AND PRECISION

Simplify - focus - act * Approximately right rather than accurately wrong * Think * ~~Do it right the first time~~

INTEGRITY AND ENTHUSIASM

Sense of humour * Honesty * High ethical standards * Excitement * Trustworthiness * Loyalty

EXCEED EXPECTATIONS

Personal initiative * Fighting spirit * Go the last mile

FUN AND PROFIT

Maximize long term shareholder value * Pass på penga * One for all, all for one * Energy

TANDBERG FIRST

First in user benefits * Innovative * "Kreativ galskap" * Understanding customer needs

- People communicate
- Focus on important stuff
- Embedded slack
- Continuous planning
- Effective decisions
- Autonomous organisation
- Respect for the doers
- No integration period
- Spectacular products
- Fast deliveries
- Sustainable pace

The 7 Lean Software Development Principles

- Eliminate Waste
- Create Knowledge
- Build Quality In
- Defer Commitment
- Deliver Fast
- Respect People
- Improve the System

(Poppendieck)

Seven Principles of Lean Software Development

Eliminate Waste

- Provide market and technical leadership - your company can be successful by producing innovative and technologically advanced products but you must understand what your customers value and you know what technology you're using can deliver
- Create nothing but value - you have to be careful with all the processes you follow i.e. be sure that all of them are required and they are focused on creating value
- Write less code - the more code you have the more tests you need thus it requires more work and if you're writing tests for features that are not needed you are simply wasting time

Create Knowledge

- Create design-build teams - leader of the development team has to listen to his/her members and ask smart questions encouraging them to look for the answers and to get back with encountered problems or invented solutions as soon as possible
- Maintain a culture of constant improvement - create environment in which people will be constantly improving what they are working on - they should know that they are not and should not be perfect - they always have a field to improve and they should do it
- Teach problem-solving methods - development team should behave like small research institute, they should establish hypotheses and conduct many rapid experiments in order to verify them

Build Quality In

- Synchronize - in order to achieve high quality in your software you should start worrying about it before you write single line of working code - don't wait with synchronization because it will hurt
- Automate - automate testing, building, installations, anything that is routine, but do it smartly, do it in a way people can improve the process and change anything they want without worrying that after the change is done the software will stop working
- Refactor - eliminate code duplication to ZERO - every time it shows up refactor the code, the tests, and the documentation to minimize the complexity

Defer Commitment

- Schedule Irreversible Decisions at the Last Responsible Moment - you should know where you want to go but you don't know the road very well, you will be discovering it day after day - the most important thing is to keep the right direction
- Break Dependencies - components should be coupled as loosely as possible to enable implementation in any order
- Maintain Options - develop multiple solutions for all critical decisions and see which one works best

Optimize the Whole

- Focus on the Entire Value Stream - focus on winning the whole race which is the software - don't optimize local inefficiencies, see the whole and optimize the whole organization
- Deliver a Complete Product - teams need to have great leaders as well as great engineers, sales, marketing specialists, secretaries, etc. - they together can deliver great final products to their customers

Deliver Fast

- Work in small batches - reduce projects size, shorten release cycles, stabilize work environment (listen to what your velocity tells you), repeat what's good and eradicate practices that creates obstacles
- Limit work to capacity - limit tasks queue to minimum (one or two iterations ahead is enough), don't be afraid of removing items from the queue - reject any work until you have an empty slot in your queue
- Focus on cycle time, not utilization - put in your queue small tasks that cannot clog the process for a long time - reduce cycle time and have fewer things to process in your queue

Respect People

- Train team leaders/supervisors - give team leaders the training, the guidance and some free space to implement lean thinking in their environment
- Move responsibility and decision making to the lowest possible level - let your people think and decide on their own - they know better how to implement difficult algorithms and apply state-of-the-art software frameworks
- Foster pride in workmanship - encourage passionate involvement of your team members to what and how they do

Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

Principles behind the Agile Manifesto

We follow these principles:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity--the art of maximizing the amount of work not done--is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Appendix

It takes a lot to stay ahead in the game...

Inhouse training

C Foundation course
C++ Foundation course
C++ Advanced course
UML Course
Pattern-Based Software Development Course
Studygroups in OOAD
Studygroups in Java
Studygroups in C++
Agile Planning, Estimation and Retrospectives
Certified ScrumMaster
Agile Modeling Design Workshop
Software Architecture
Pattern Language of Tandberg
Python course
Scala course
... and much more

Hackers' Corner - Internal seminars

Inhouse conferences

TechZone Lillehammer 2007
~ 190 engineers, 32 talks, 4 tracks

TechZone Lysaker 2007
~ 260 engineers, 40 talks, 5 tracks

TechZone Barcelona 2008
~ 380 engineers, 43 talks, 5 tracks

TechZone Storefjell 2010
~ 500 engineers, 45 talks, 5 tracks, TechFair

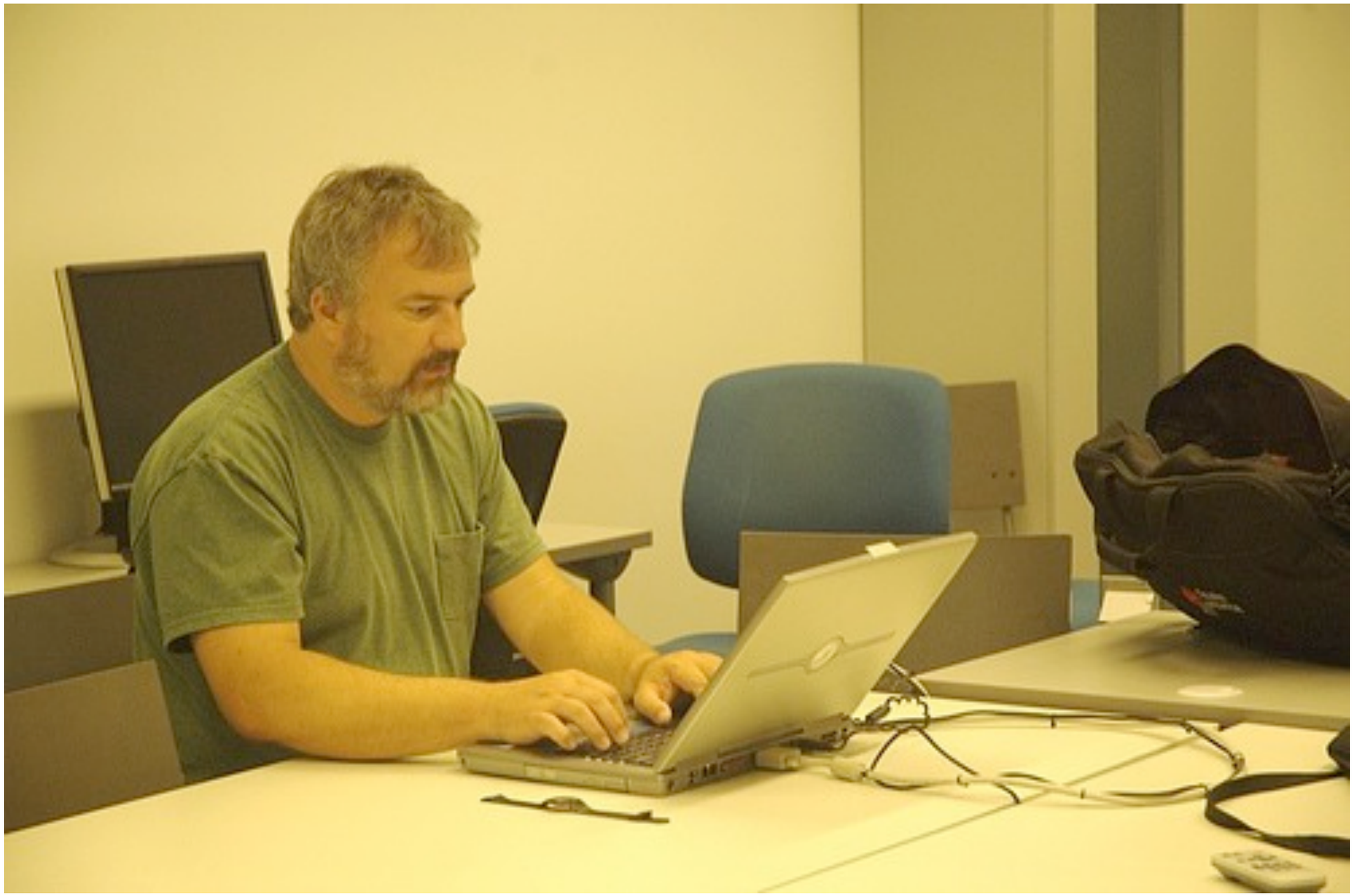
Engage industry experts...

Dana Bredemeyer - Software Architecture

Kevlin Henney

C++, C, UML, Pattern-Based Software Development, Pattern Language of Tandberg

Michael Feathers
Pair-Programming, Testing, Professionalism



Jon Jagger - C, C++, UML, Patterns, Agile, TDD

Jutta Eckstein
Project Management, Agile Planning, Retrospectives

Robert C. Martin “Uncle Bob”
TDD, Pair Programming, Professionalism, Agile Design Course (SOLID)

Tom and Mary Poppendieck
Project Management, Lean Software Development

Craig Larman
Agile Modeling Design Workshop

David Beazley - Python training

Geir Amsjø - Certified ScrumMaster Course

Bruce Perens - Free and Open Source issues

Jonas Boner - Scala training

Bjarne Stroustrup - About past and future of C++

RMS

"I have been working with software development groups all around the world, and you are way ahead of most."

(a consultant visiting our R&D department)