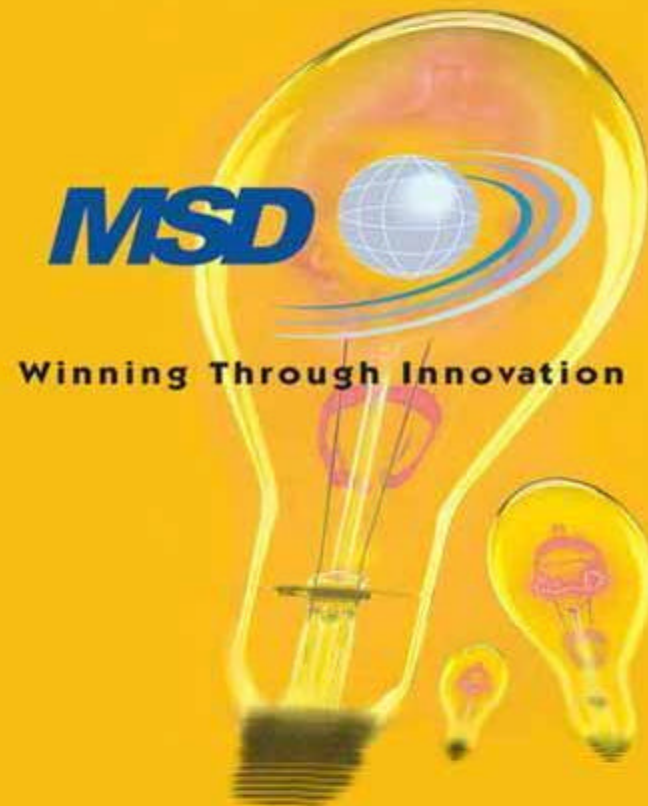


Enhanced Log Records to Assure Test Traceability

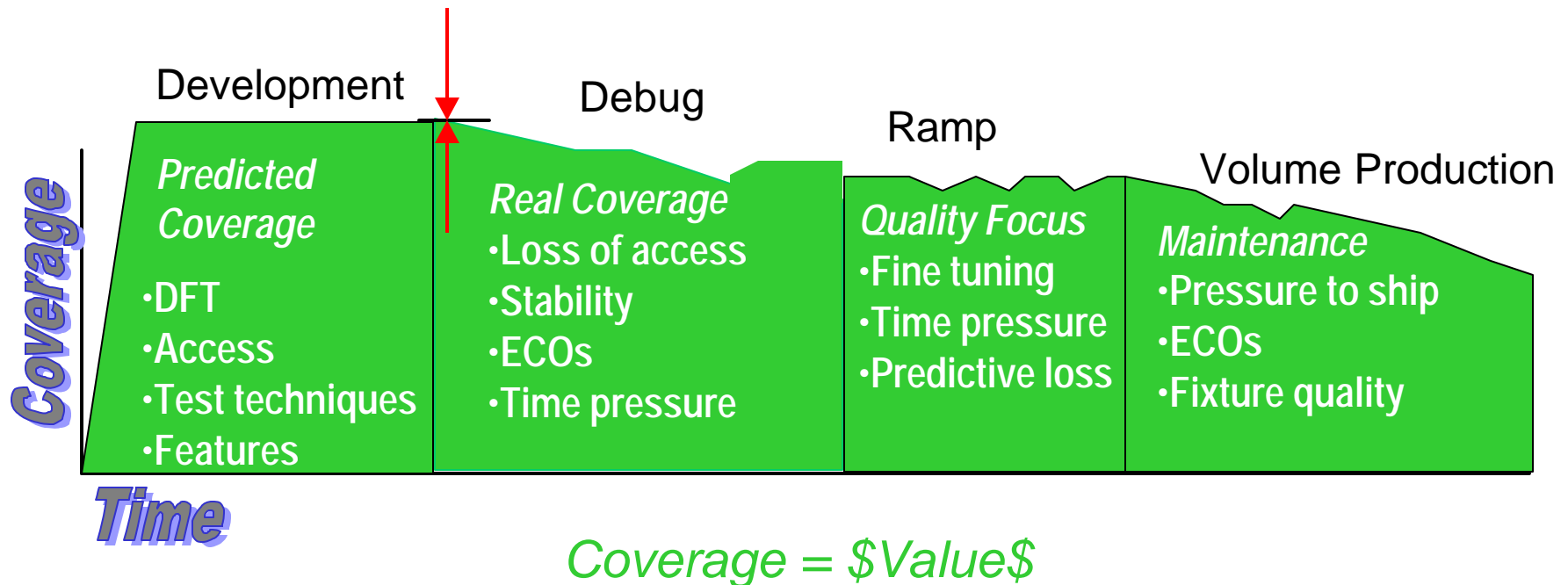
Dayton Norrgard
MSD-ICT, Loveland CO
dayton_norrgard@agilent.com

BTW 2007



Problem Statement

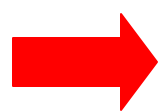
Production board test lacks the metrology to detect test program modifications and provide alerts when these events occur. This results in a loss of confidence in test quality over time.



Patient Monitoring System

A heartbeat monitor continuously sends telemetry to a nurse station based on a patient's heart condition. The data stream describes several health conditions:

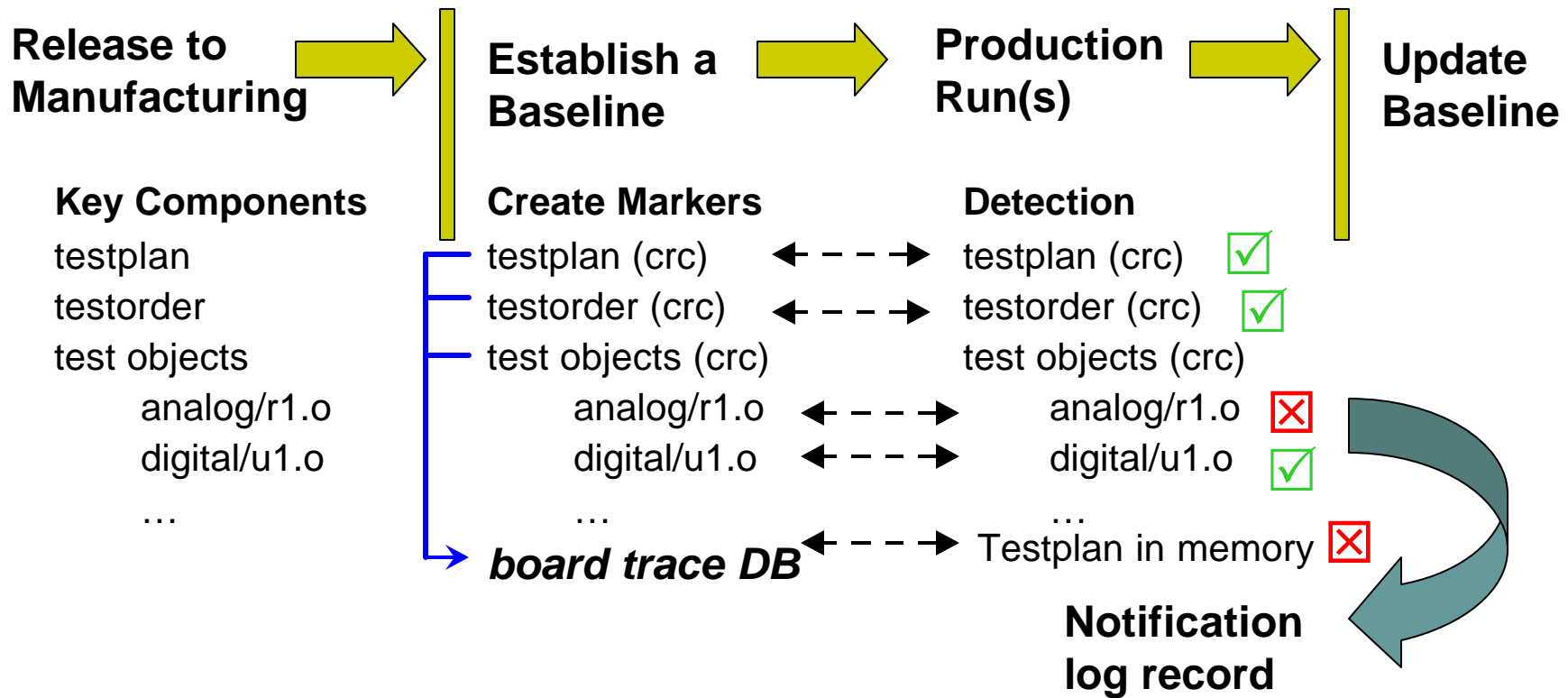
- Normal, regular heartbeats are good.
- Heartbeat fluctuations may indicate a potential life threatening condition, requiring immediate attention.
- Loss of telemetry indicates a break down in the patient monitoring system and no insight on the stability of the patient. Attention required.



Critical events must notify someone who can take immediate action. These triggers could dispatch alarms, emails, pages, or log status reports for periodic review.

Integrating Traceability into a Tester

Use Case Scenario



Baseline phase establishes a known good working state by storing aside important board information

Detection phase identifies when a change occurred between a component and its established baseline

Notification phase communicates the change in a signed and verified log record

Baseline Creation

A baseline gathers details about board directory files such as:

- File name (i.e., analog/1%r1)
- Timestamp
- CRC
- System details

Baselining is not file revision control

- Test objects and test sources are decoupled, that is a source file can be edited and not recompiled.
- Test objects are binary so “diffing” a binary object only effectively reports that it is different.
- There are dozens of ways to edit files: BT-Basic, notepad, vi, word, ...; a third-party revision control system is the best way to capture and manage file content modification; however, this alternative can be expensive.

Baselining is password protected

- Passwords associated with the database, not system accounts
- Modification rights follow the board

Baselining captures state information on all objects, independent of board versions or test program flow

Detection Phase

Test objects are loaded into BT-Basic memory from disk for execution at first run

- Any performance penalty is realized at first runtime
- For $nrun > 1$, objects are not re-loaded from disk unless the file changes (object checking on/off command controls timestamp checking)
- The detection system knows when the file is loaded and when to perform the crosscheck with the baseline
- Detection is stealth, does not halt or inhibit board test

Likewise the detection system knows when the testplan in memory has been edited. When the board test is run, this edited testplan is crosschecked against its baseline.

Notification Phase

Notification built upon an already excepted standard using 3070 log records

- 3070 data logging is an established, well known practice
- 3rd party vendors support 3070 log records in their quality management software

“log board end” builds the notification record

- Normal records contain telemetry that the process is working for a stated board serial number
- Questionable records flag problems that drill down when and where a file change occurred
- Data log integrity check ensures record contents are digitally signed from board test database to prevent and detect log record tampering

Sample Log Records

An example of a good “*heartbeat*” log record:

```
{@BTEST|SN0123456|01|070518191203|000000|0|all||n|n|070131110651||0  
{@BLINE|SN0123456|0|070518191203|1|mtddayton.agilent.com|0|  
C:/Agilent_ICT/boards/myBoard|testplan|0  
{@BSIGN|3127dbf1afc2f77e8fcbf01|a5fe4a76ce67da91be2c59d9d}}}
```

These you want to see!

- A heartbeat identifies the board, where it is tested, when it is tested, the testplan used, and its current baseline revision.
- A digital signature verifies the BLINE record content, which includes the board serial number, location, and its public production key.
- A separate application named logsentry parses the log record and validates its authenticity. You can be assured where it came from and that it is genuine!

Sample Log Records (cont)

This example shows that the testplan loaded into memory has been modified:

```
{@BTEST|SN0123456|01|070131110651|000000|0|all||n|n|070131110651||0  
{@BLINE|SN0123456|Status|070131110328|2|mtddayton.agilent.com|0|C:/Agilent3070/boards/myboard|testplan|1  
{@BFILE|testplan|1  
{@REV|1|0x8b27c1ba|070125114758|165517}  
{@MEM|1|0x14a4de15|070131111308|165517}  
}  
{@BSIGN|3127dbf1afc2f77e8fcbf01|ab807da2c829d4ef5d9864f46}}}
```

- *Additional details provided pin point where, what, and when*
- *with authenticity*

What's in the 7.1 Release

- **Baseline creation tools**
 - Simple tools for creating, updating, and monitoring modification activity.
 - Fast database engine.
- **Test validation detection and reporting**
 - Cross checking at runtime
 - First runtime—unnoticeable performance hit
 - Stealth operation, invisible to production activities
- **Testplan in memory edit detection and reporting**
- **Log notification and protection with digital signatures**
 - Digitally signed and assured by the tester
 - Log sentry application verifies log signatures
- **New log records supported by 3rd party vendors**
 - Derby Associates
 - SigmaQuest
- **User Documentation**