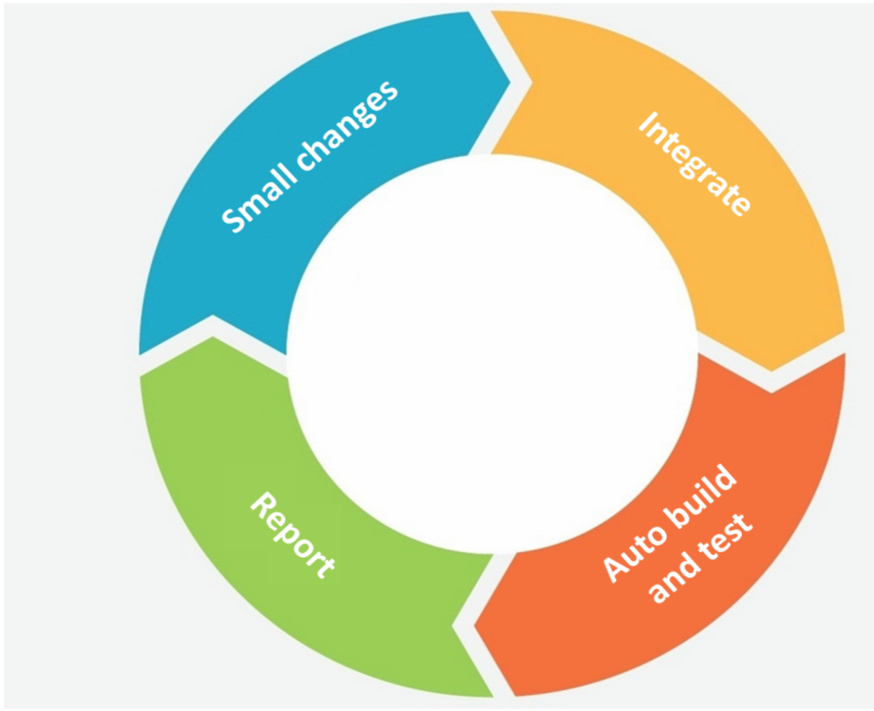


CONTINUOUS INTEGRATION

Elke Matthijs and Maïkel Vandorpe
elke.matthijs@mediagenix.tv
maikel.vandorpe@mediagenix.tv

CONTENTS

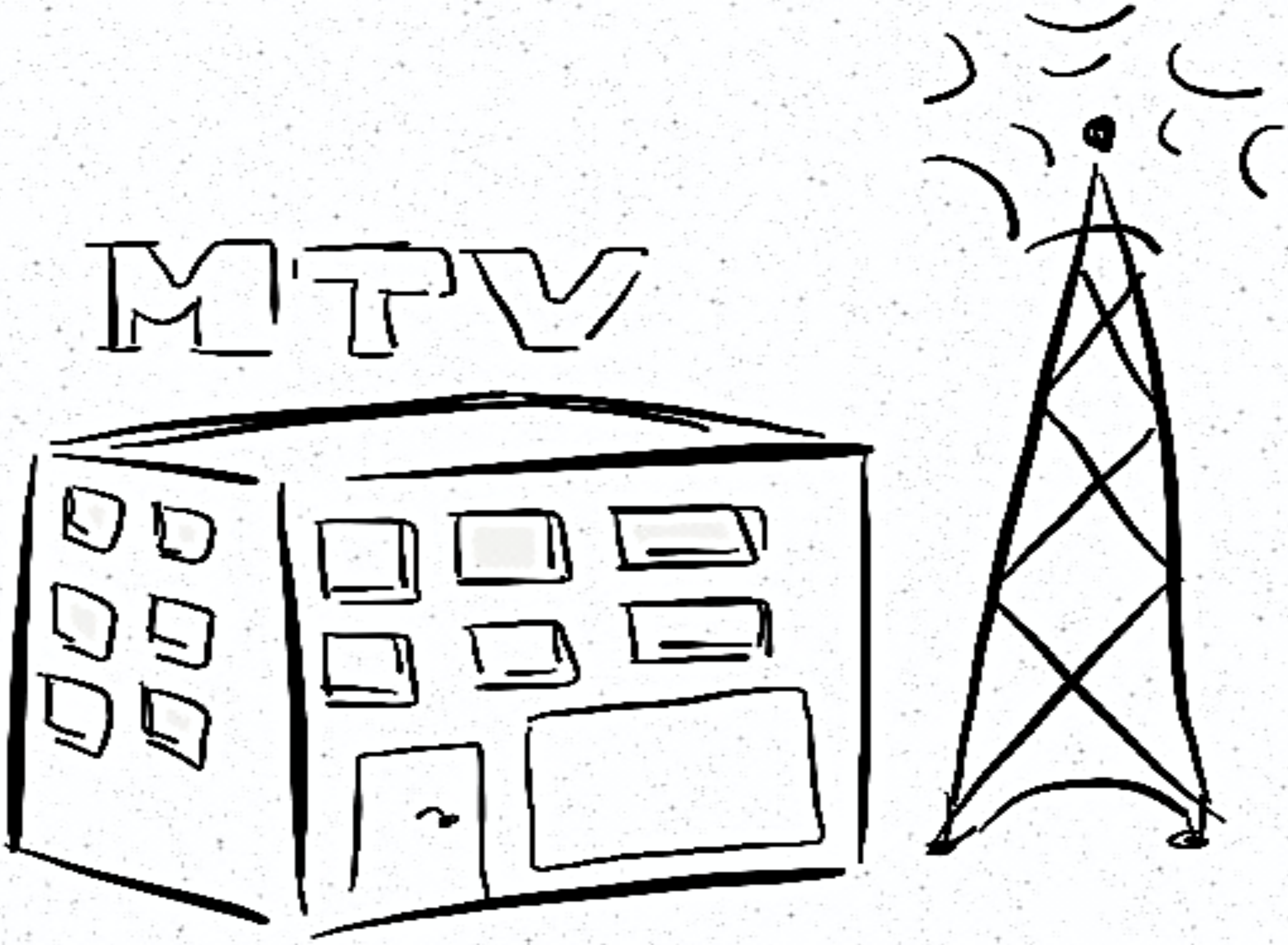
Our Product

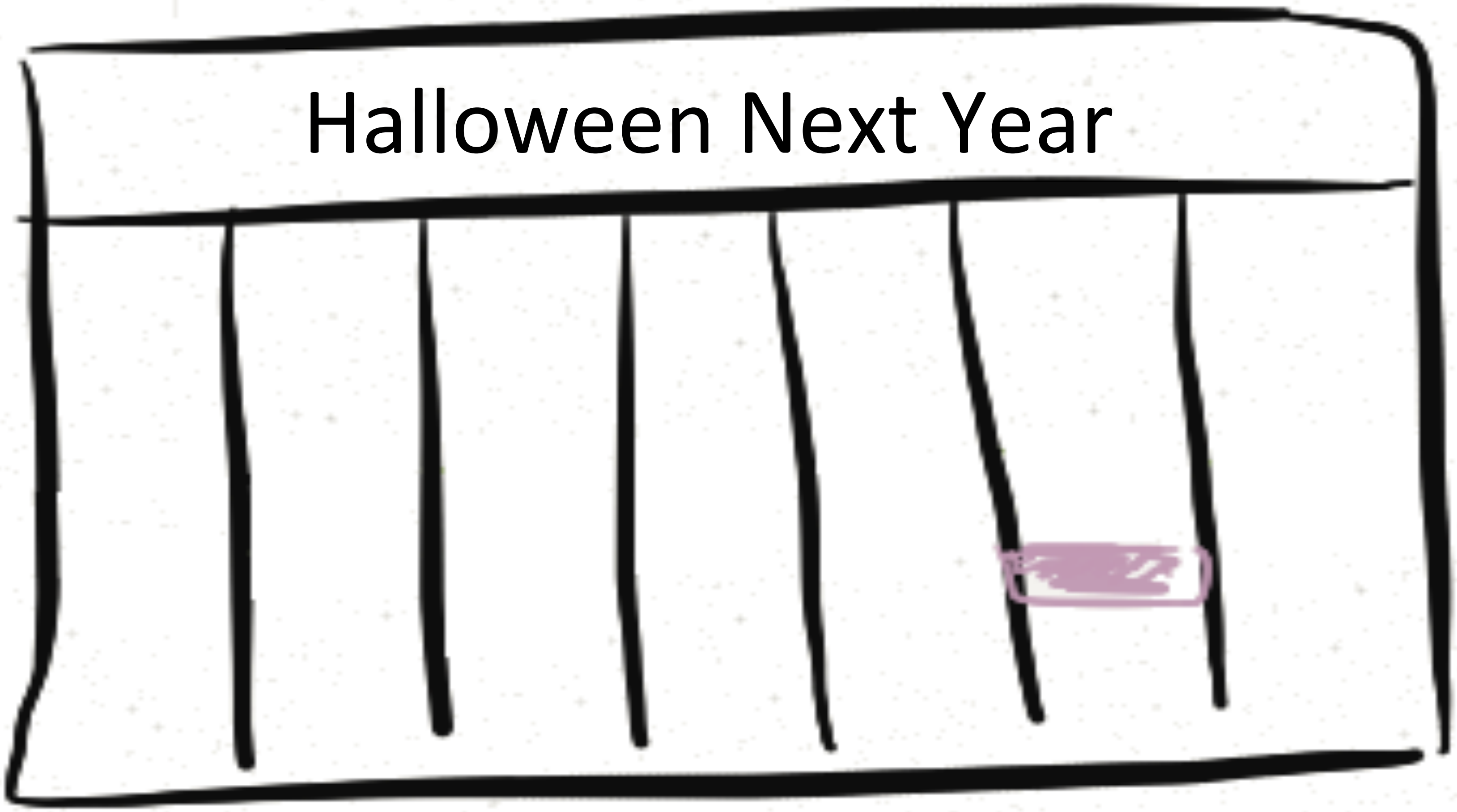
Our Customers

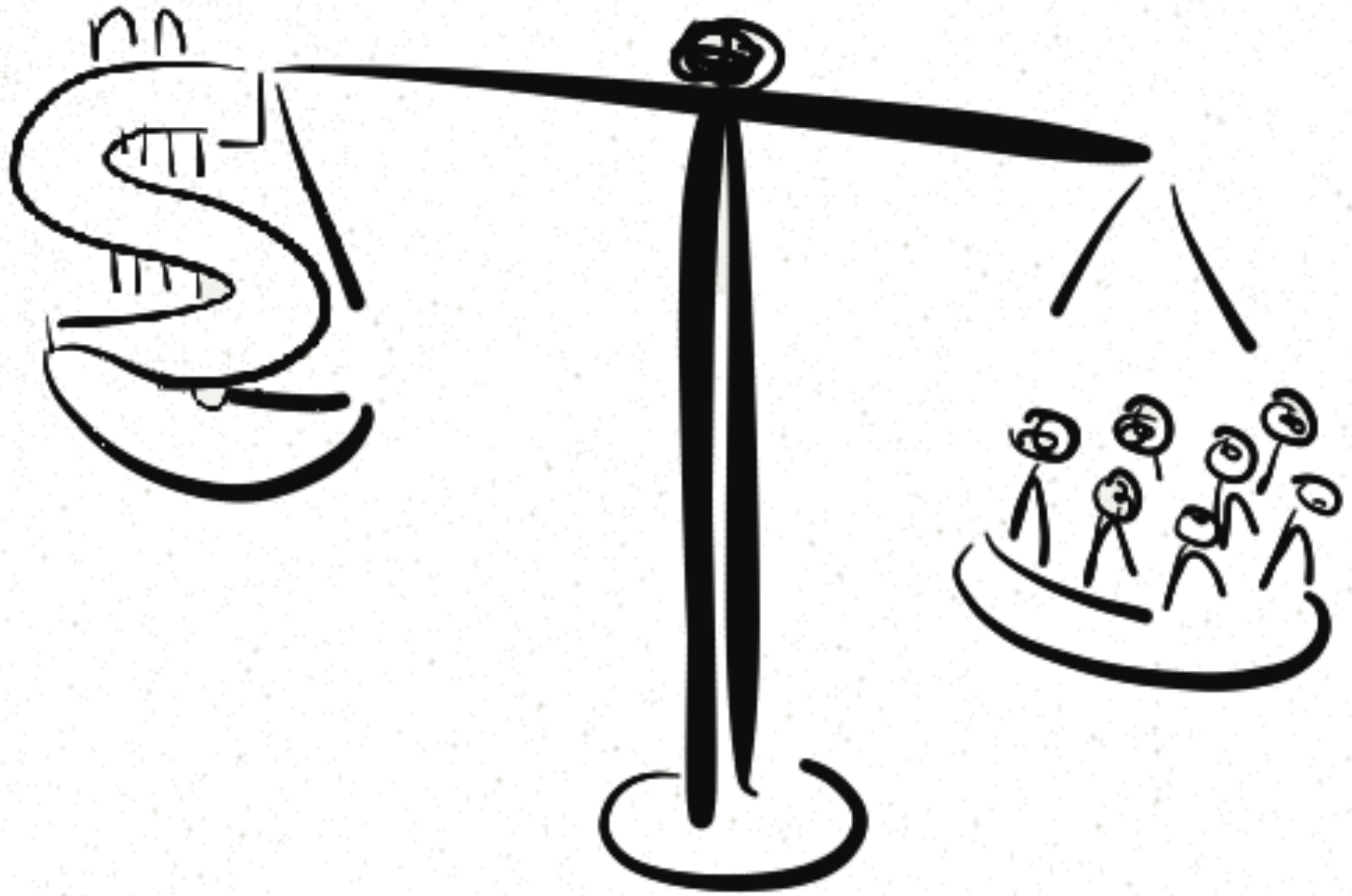
Technical Product Information

MEDIAGENIX

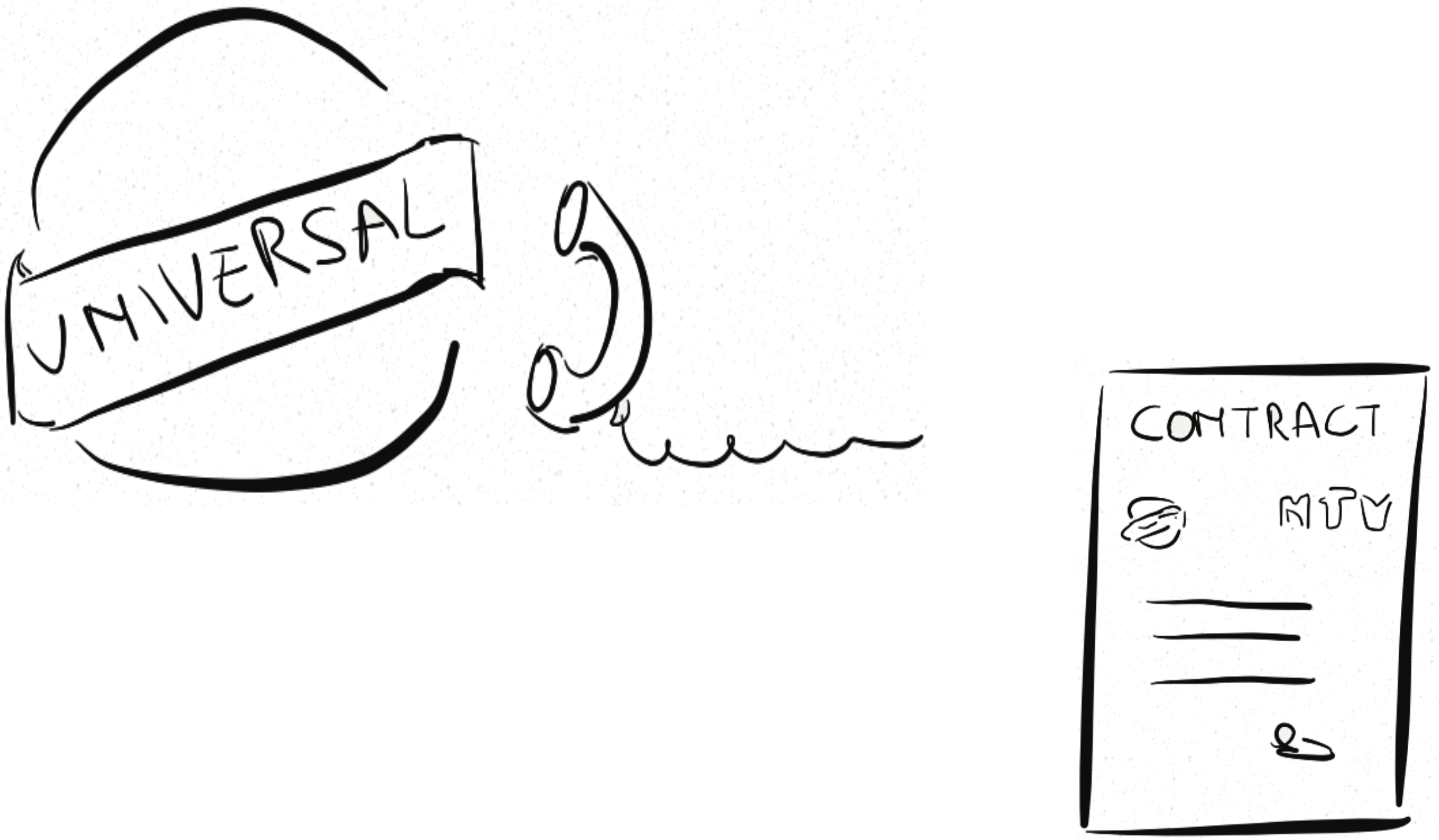
PRODUCT COMPANY
20 YEARS OF
WHATS'ON

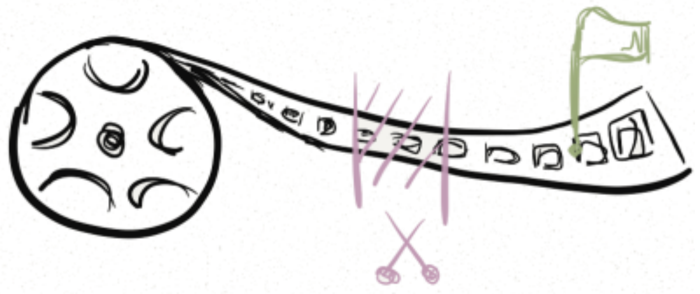

MTV

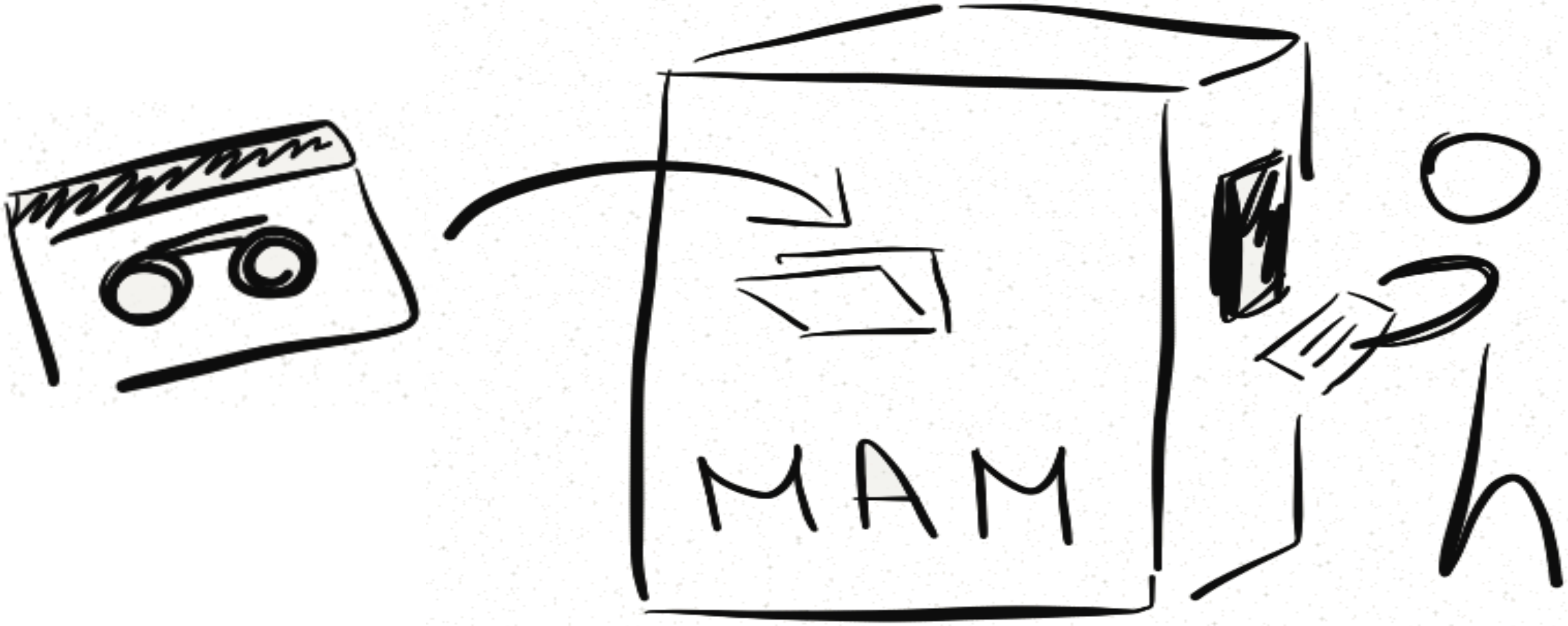

Halloween Next Year

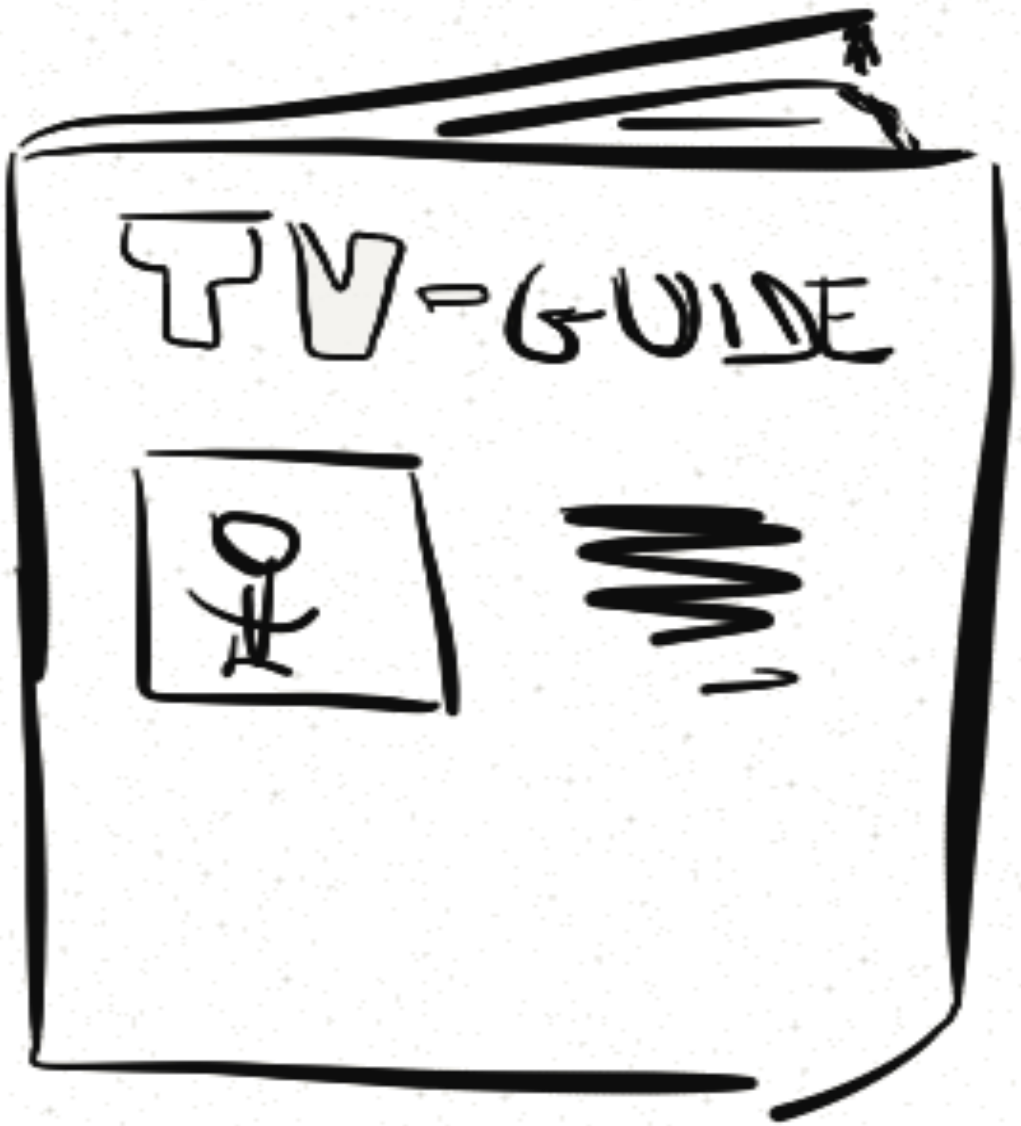


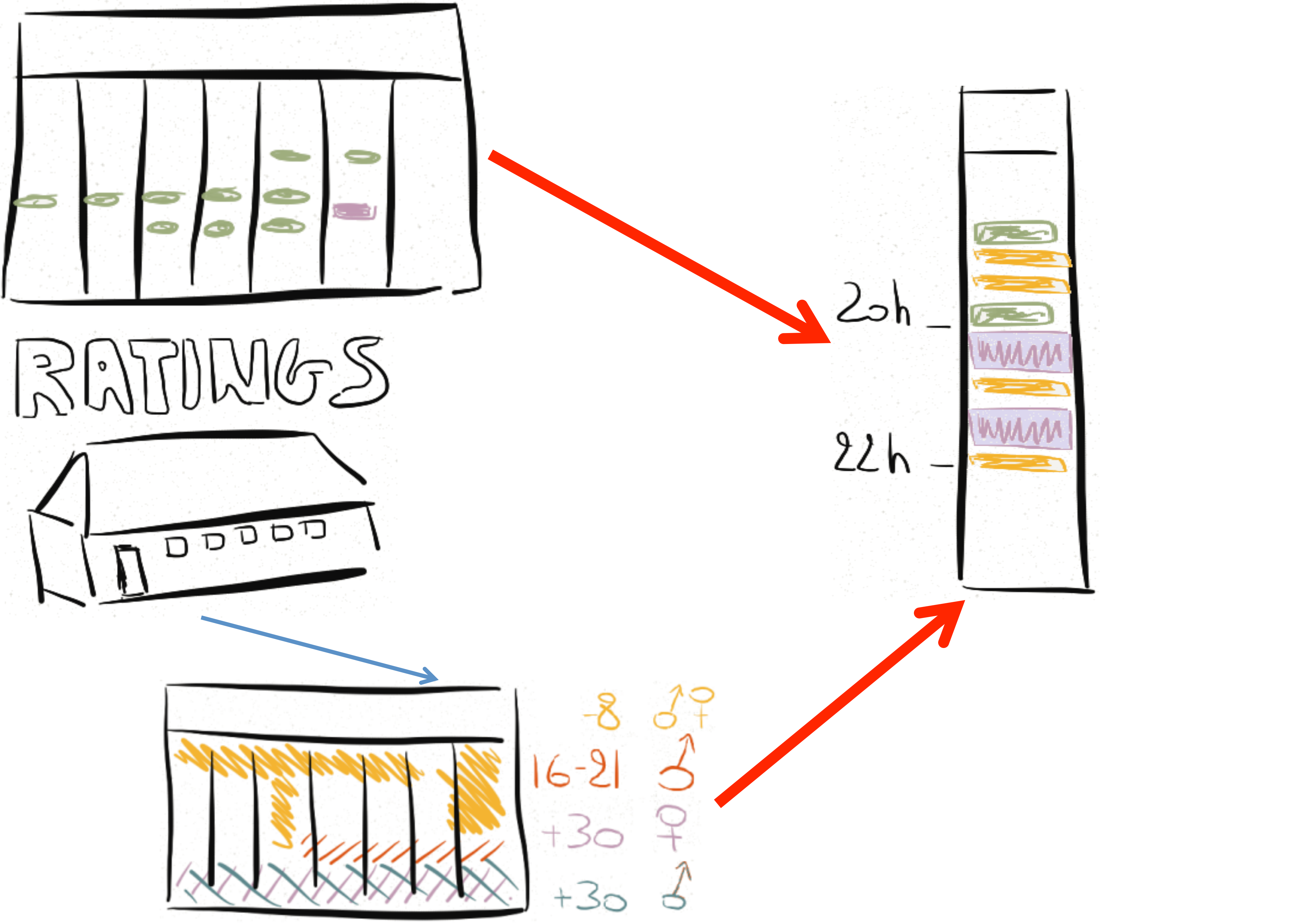
BUYING
IN ORDER
HALLOWEEN

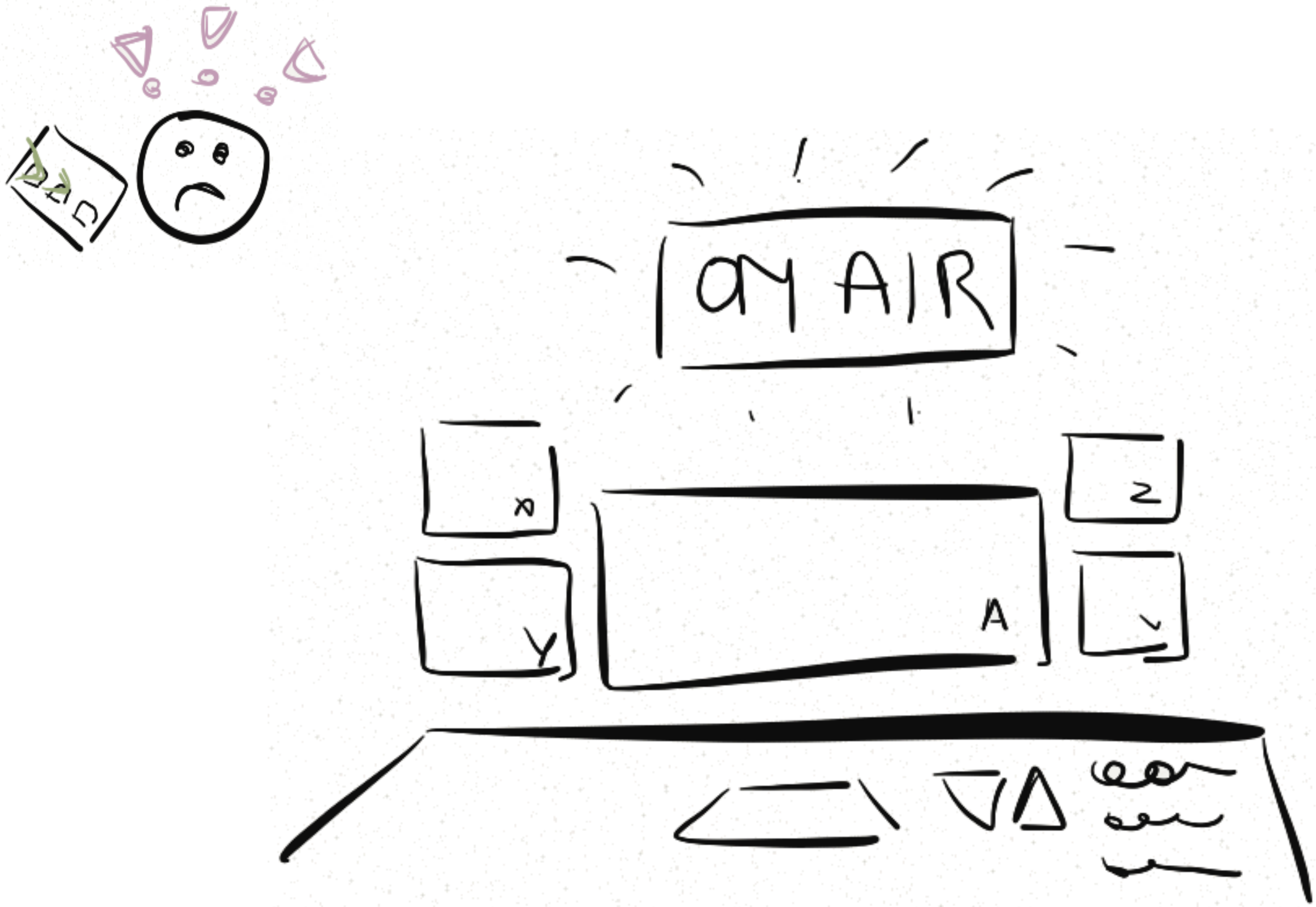

UNIVERSAL
CONTRACT
MTV

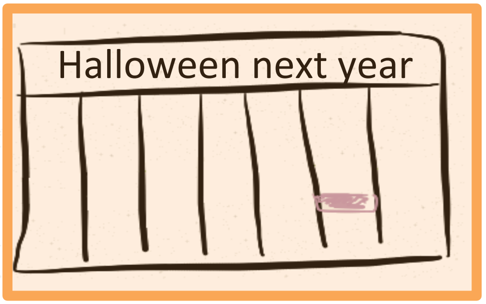



MAM

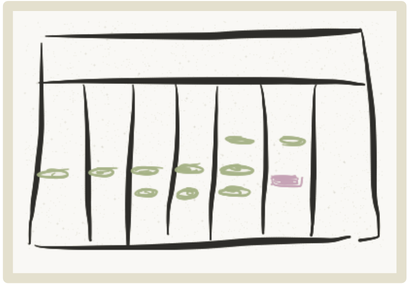

TV-GUIDE

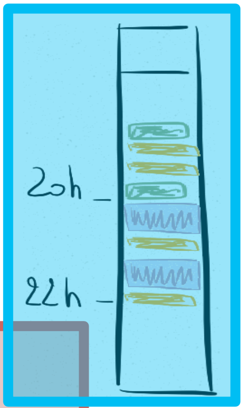

RATINGS
20h
22h
-8 ♂♀
16-21 ♂
+30 ♀
+30 ♂


ON AIR
X
Y
A
Z

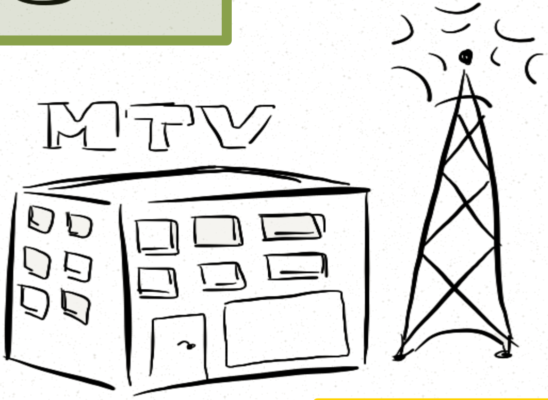

Halloween next year

CONTRACT
MTV

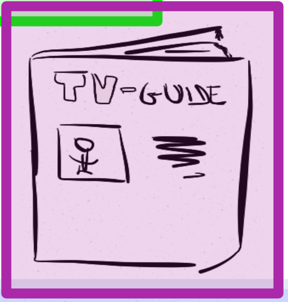


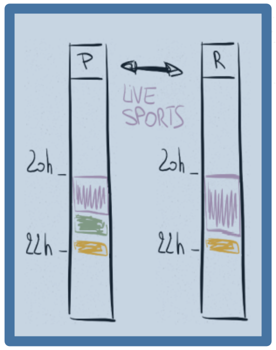

20h
22h

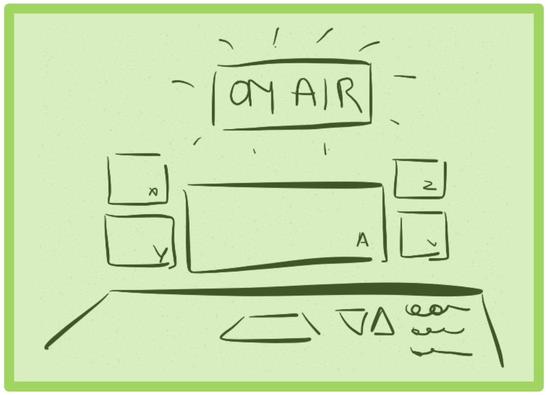
BUYING ORDER
HALLOWEEN

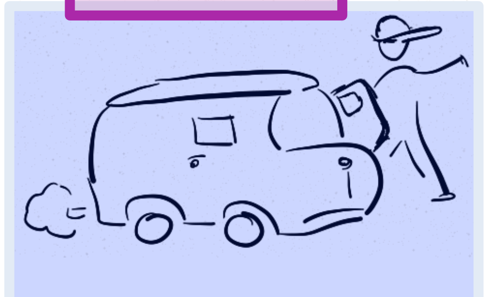

MTV

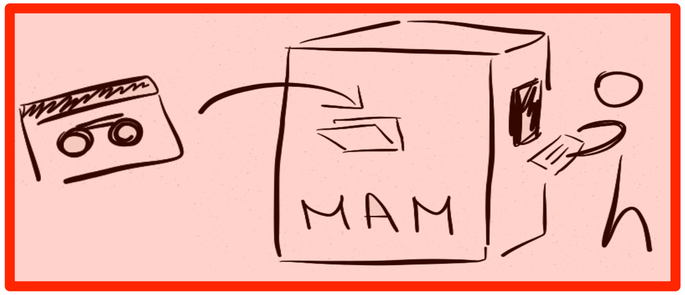
RATINGS
-8
16-21
+30
+30


TV-GUIDE

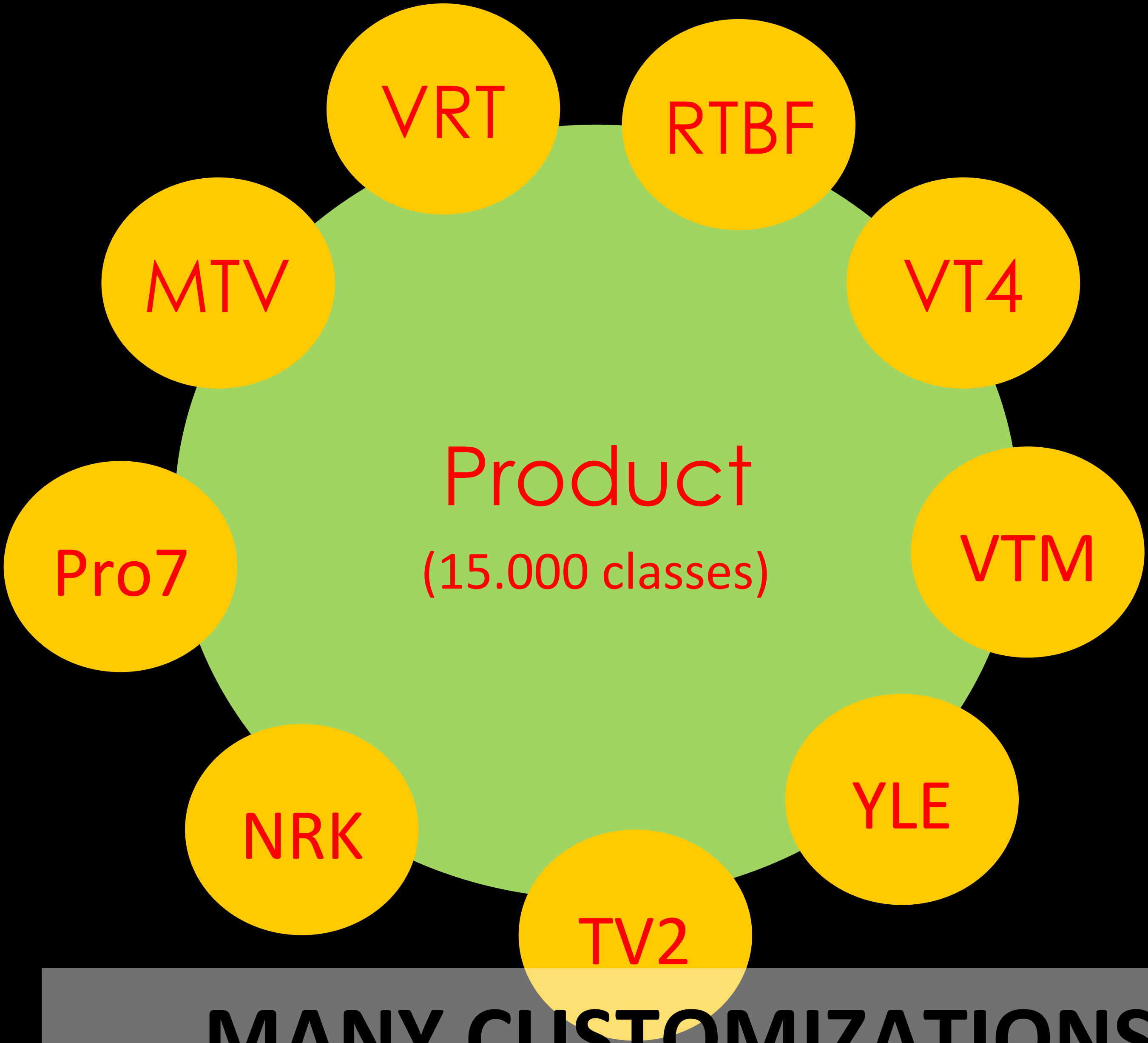

P
R
LIVE SPORTS
20h
22h
20h
22h


ON AIR
X
Y
Z
A




MAM

CUSTOMERS



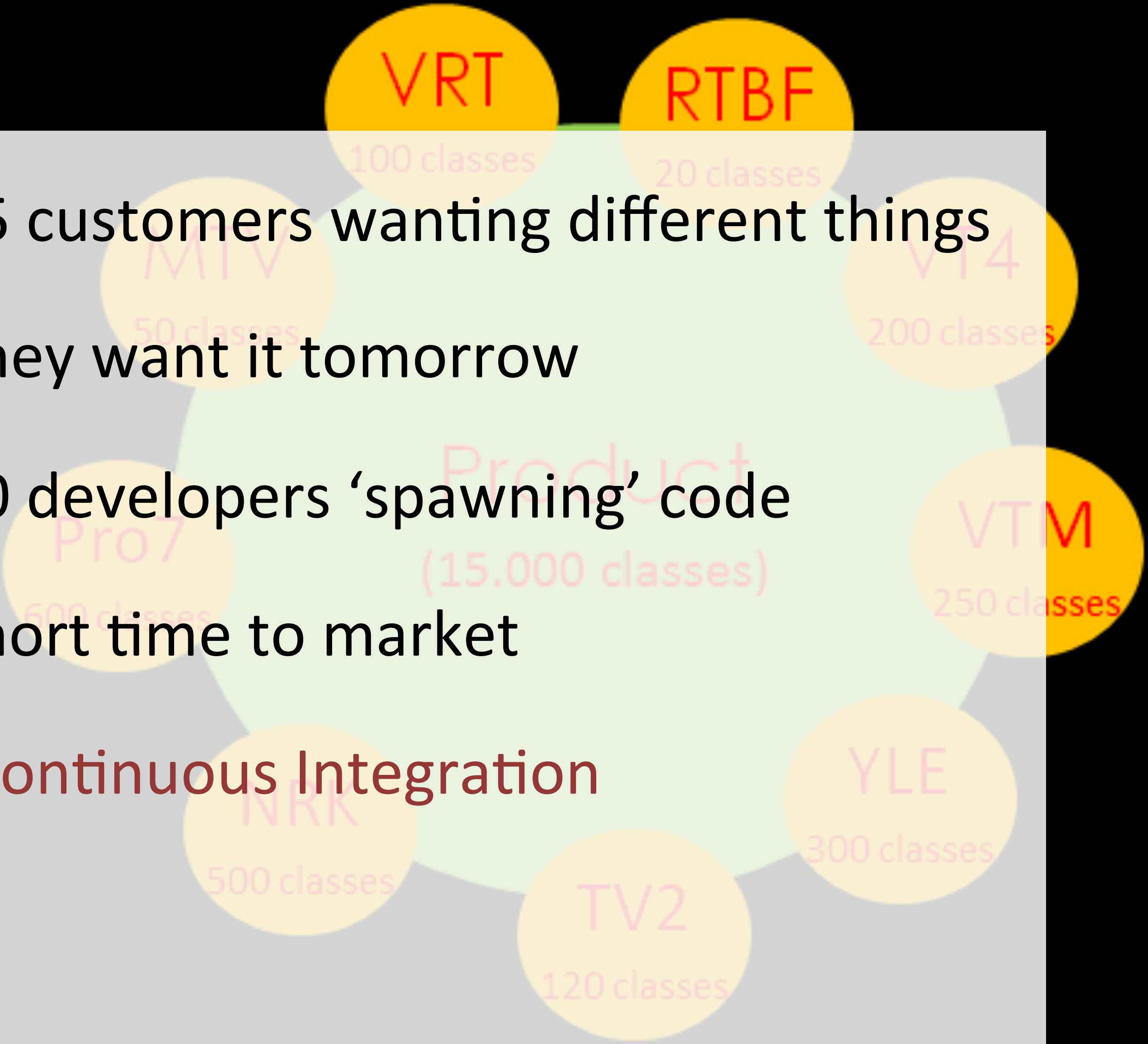
MANY CUSTOMIZATIONS

90 Employees

- 30 Software Developers
- 6 Project Managers
- 13 Functional Analysts
- 17 Customer Support Analysts
- Other …

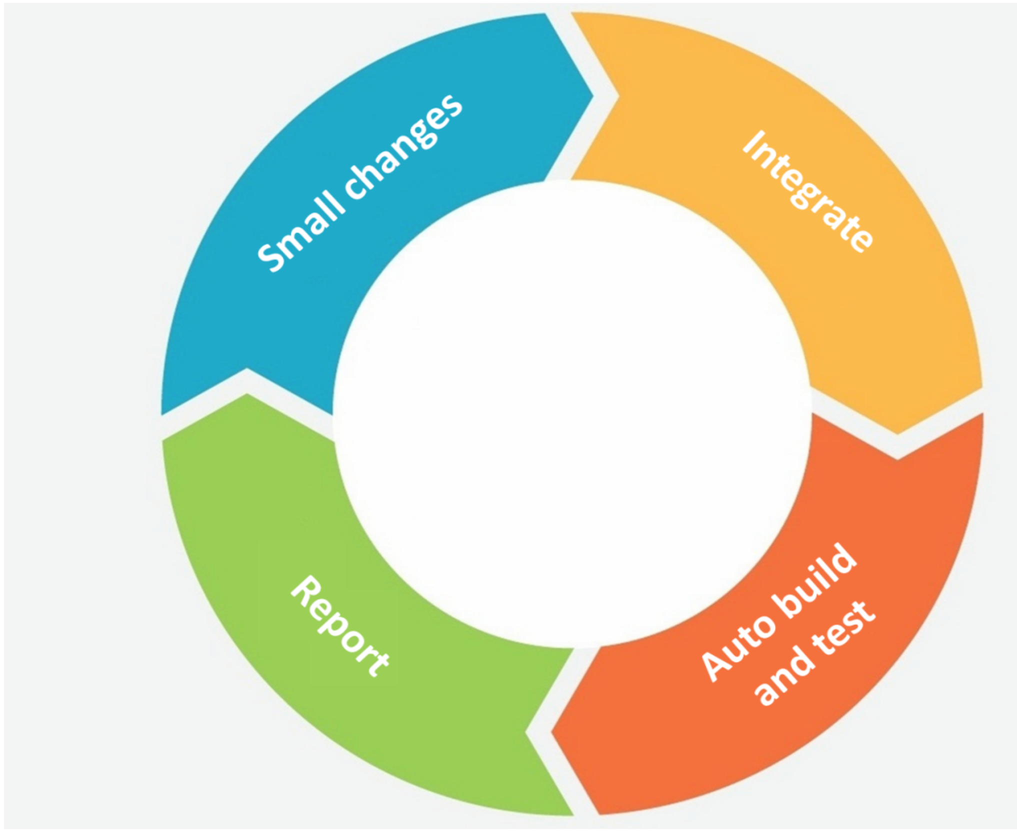
Application Overview

- 1.9 Million lines of code
- 15.000 own classes
- 200.000 methods
- 1000 database tables



- 35 customers wanting different things
- They want it tomorrow
- 30 developers ‘spawning’ code
- Short time to market

=> Continuous Integration



Context

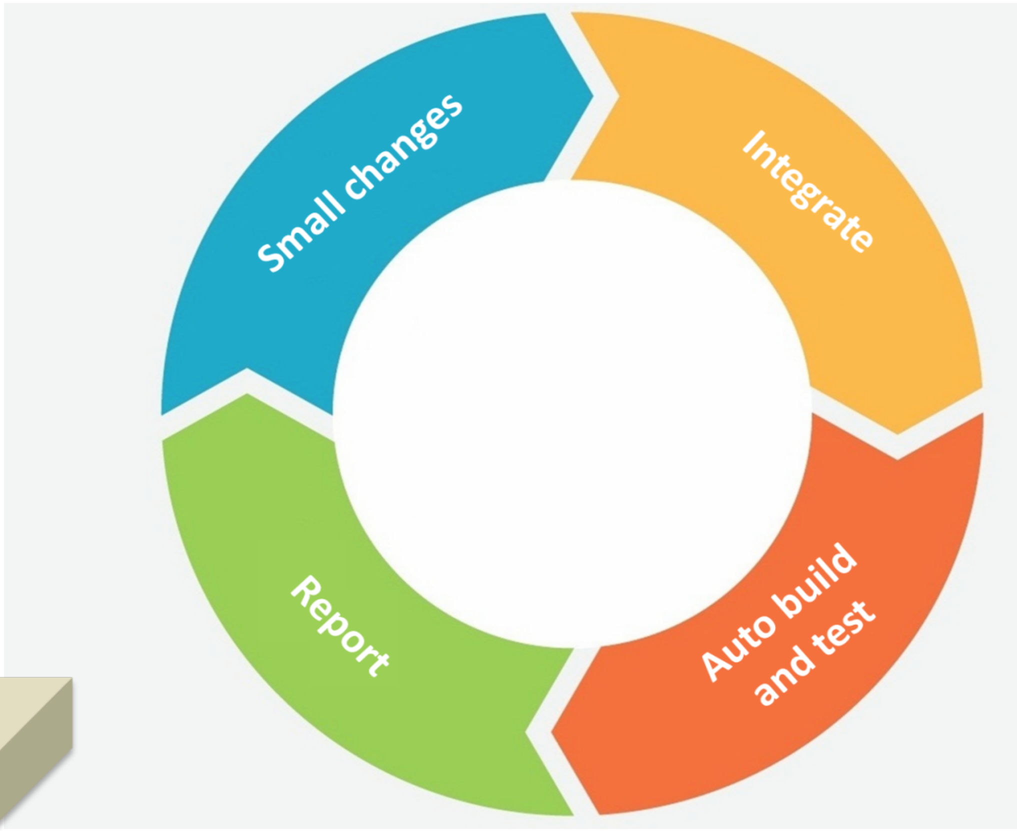
Current testing system

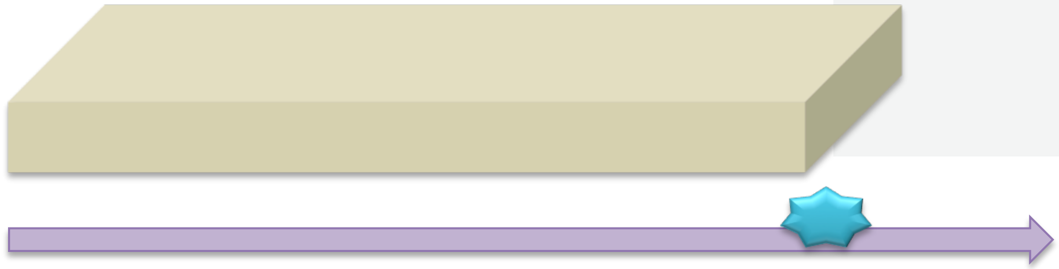
Tools

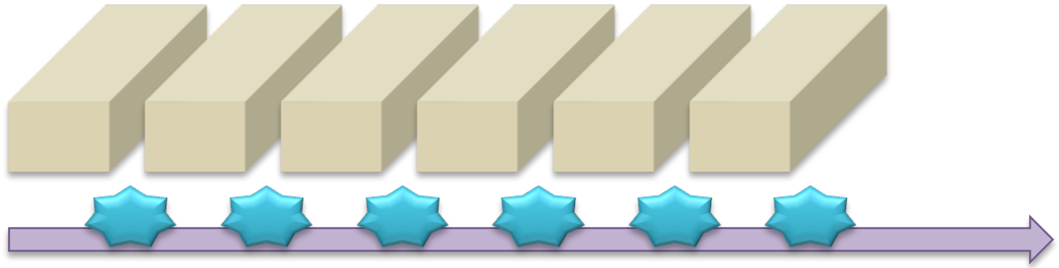
CONTINUOUS INTEGRATION

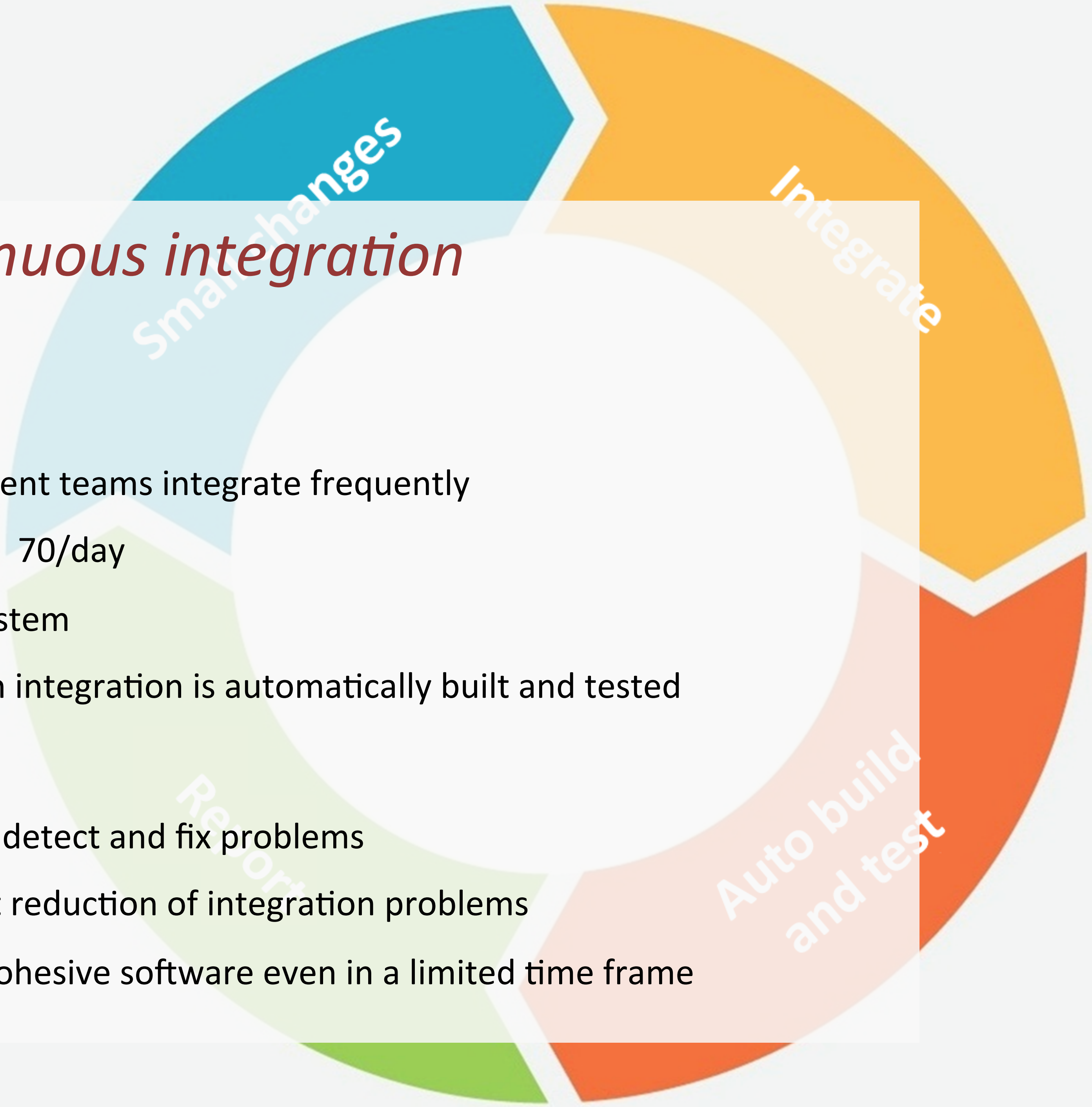
Continuous integration

- Many isolated changes
- Many integrations
- Each is immediately tested
- Detect and fix problems soon









Continuous integration

Development teams integrate frequently

- Avg: 70/day

Testing system

- Each integration is automatically built and tested

Efficiently detect and fix problems

Significant reduction of integration problems

Develop cohesive software even in a limited time frame

What do we test?

As much as possible

- Unit tests
- Consistency checks
- UI tests and Acceptance tests in general
- Refactoring tests

Sequential running of tests take over 12 hours

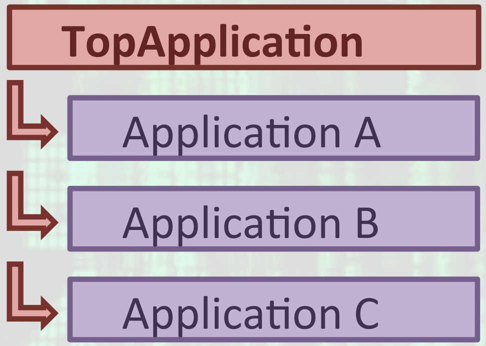
Code consistency checks

Check if code guidelines have been followed

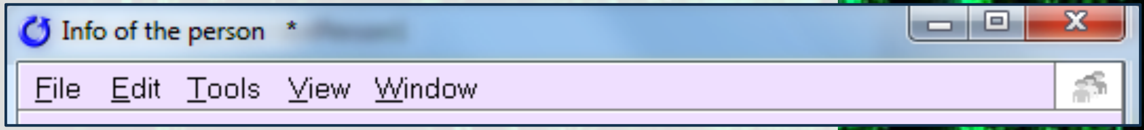
Why ?

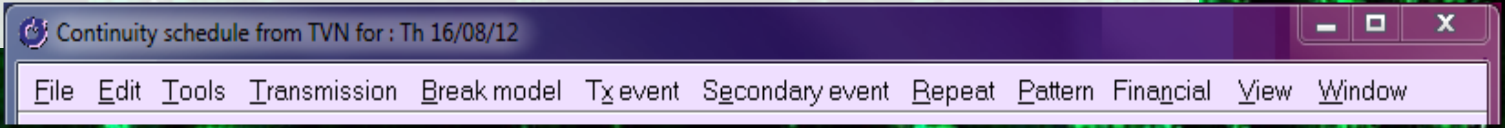
- Consistent code makes life easy
- Reduce number of bugs

Code consistency checks



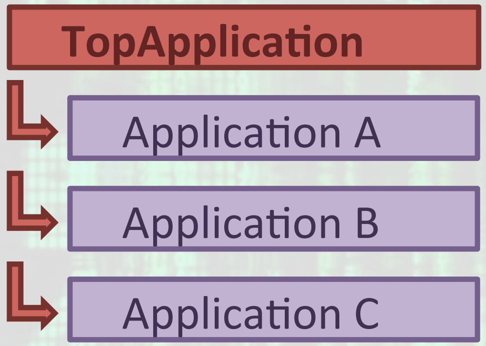
```
buildMenuBar

    | menu |
    (menu := Menu new)
        addItem: self fileMenuItem;
        addItem: self editMenuItem;
        addItem: self toolsMenuItem.

    self addMenuBarItemsTo: menu.
    menu
        addItem: self viewMenuItem;
        addItem: self windowsMenuItem.
    ^menu
```





Continuity schedule from TVN for : Th 16/08/12
File Edit Tools Transmission Break model Tx event Secondary event Repeat Pattern Financial View Window

Code consistency checks



```
test_topApplication_buildMenuBar
    "Don't implement #buildMenuBar.
    use #addMenuBarItemsTo:"

    self assert: (self implementorsOf: #buildMenuBar
                            inSubclassesOf: TopApplication)
    isEmpty.
```

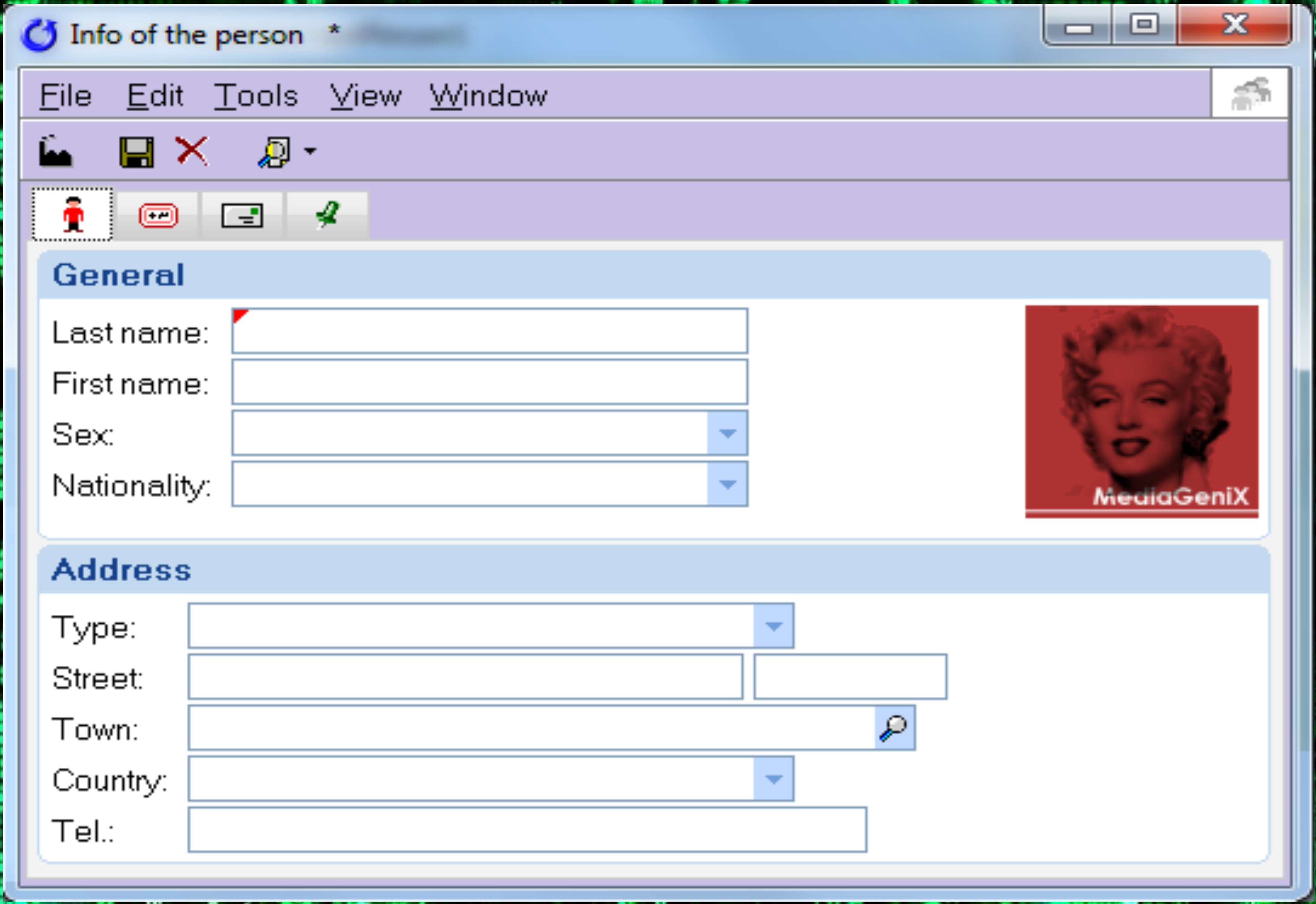
UI tests and Acceptance tests

Why ?

- Unit test
    - Test a method
    - Change the method, change the test

- Acceptance test
    - Emulate a user process
    - More stable tests

=> Never change functionality, unless user asks for it

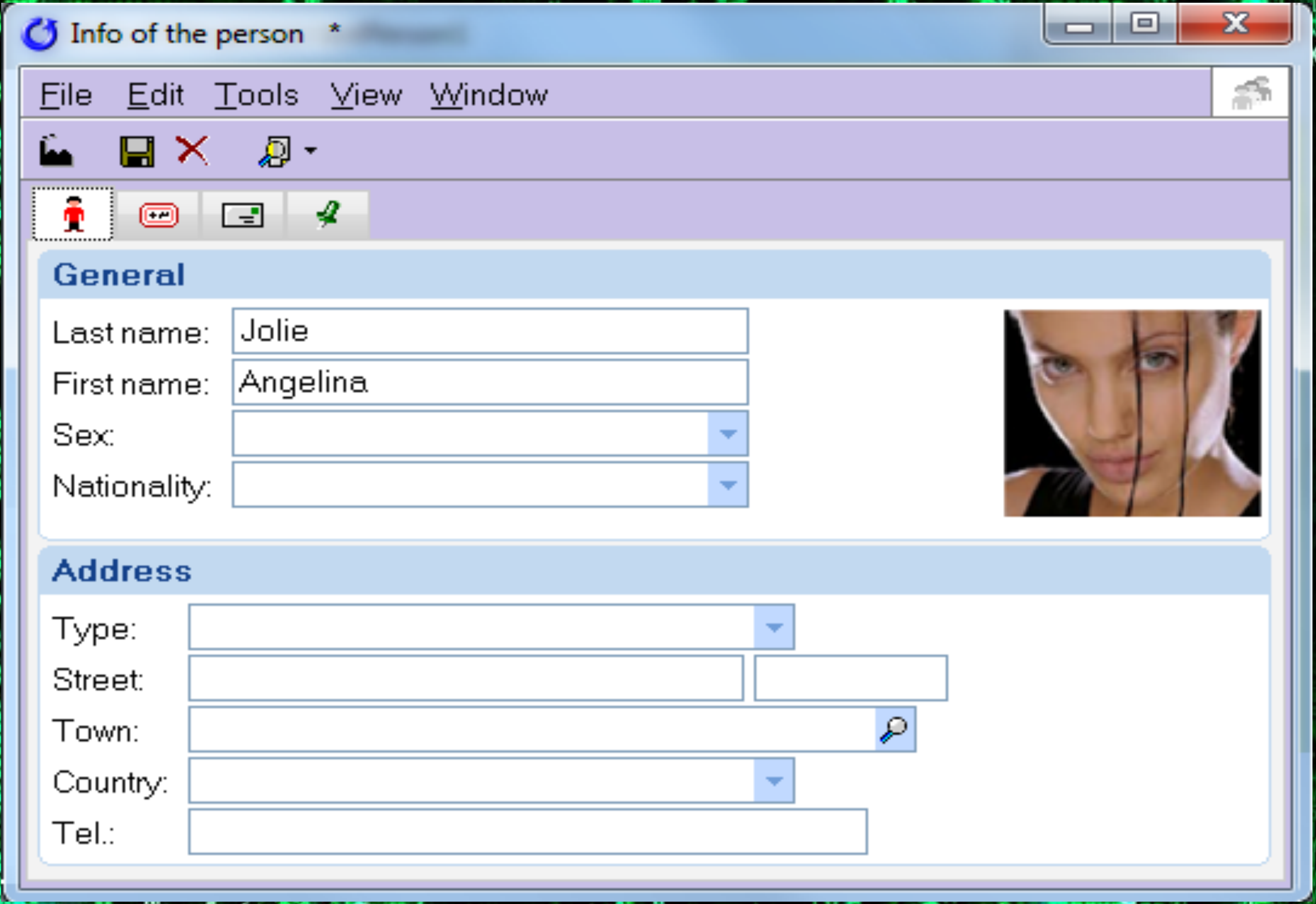


UI Testing – A Person Editor

```
test_editPerson
        | person|
        person := Person withFirstName: 'Angelina'
                andLastName: 'Jolie'.
        self deny: (Person existsWhere:
                [:p| (p firstName = 'Bruce')
                & (p lastName = 'Willis')]).
        self openPersonEditorOn: person.
        self editField: #lastName value: 'Willis'.
        self editField: #firstName value: 'Bruce'.
        self toolBarPress: #saveCommand.
        self assert: (Person existsWhere:
                [:p | (p firstName = 'Bruce')
                & (p lastName = 'Willis')])
```

```
test_editPerson
	| person|
	person := Person withFirstName: 'Angelina'
			andLastName: 'Jolie'.
	self deny: (Person existsWhere:
			[:p| (p firstName = 'Bruce')
			& (p lastName = 'Willis')]).
	self openPersonEditorOn: person.
	self editField: #lastName value: 'Willis'.
	self editField: #firstName value: 'Bruce'.
	self toolBarPress: #saveCommand.
	self assert: (Person existsWhere:
			[:p | (p firstName = 'Bruce')
			& (p lastName = 'Willis')])
```

```
test_editPerson
        | person|
        person := Person withFirstName: 'Angelina'
                andLastName: 'Jolie'.
        self deny: (Person existsWhere:
                [:p| (p firstName = 'Bruce')
                & (p lastName = 'Willis')]).
        self openPersonEditorOn: person.
        self editField: #lastName value: 'Willis'.
        self editField: #firstName value: 'Bruce'.
        self toolBarPress: #saveCommand.
        self assert: (Person existsWhere:
                [:p | (p firstName = 'Bruce')
                & (p lastName = 'Willis')])
```

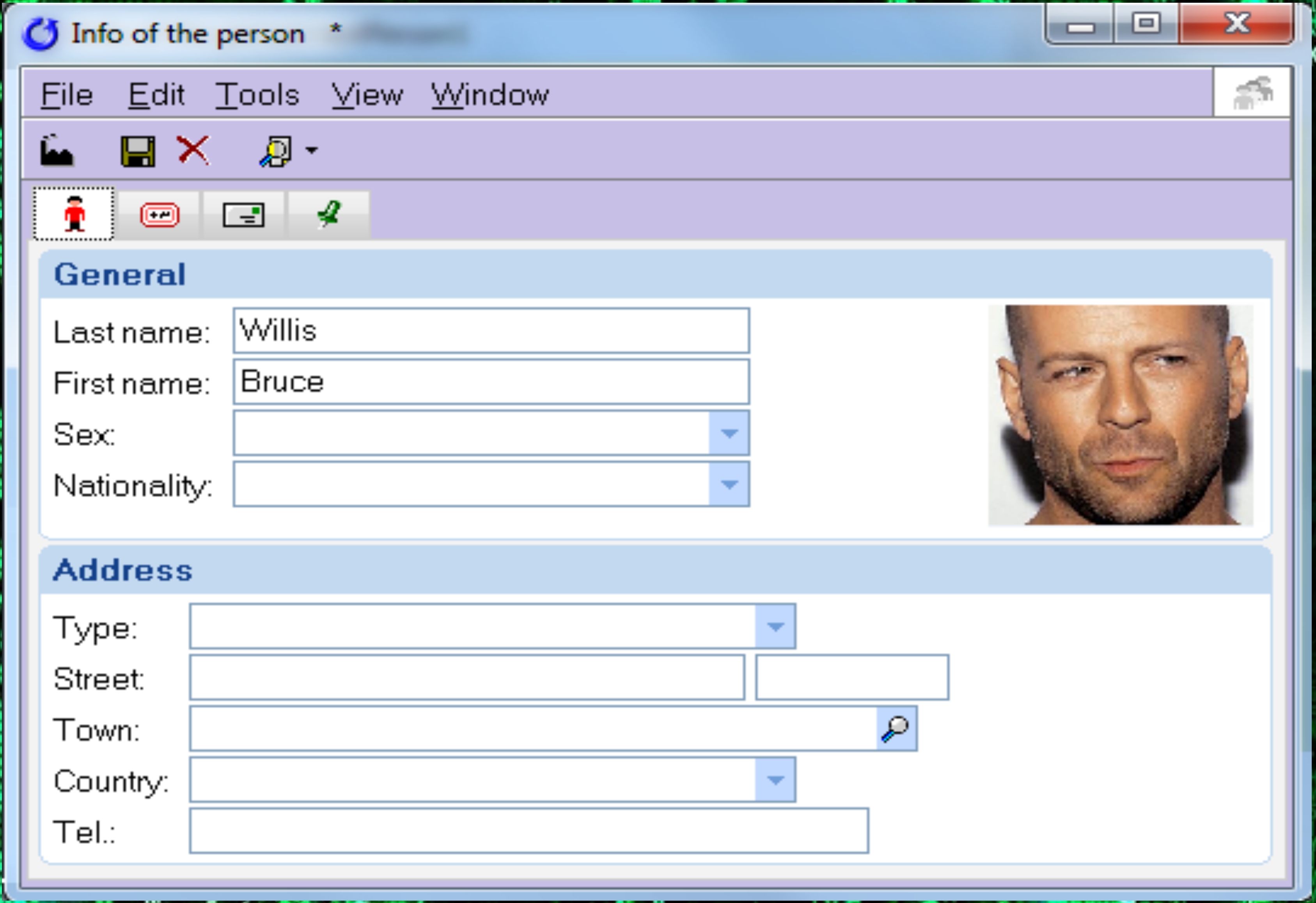


Last name:	Jolie
First name:	Angelina
Sex:	
Nationality:	

Type:	
Street:	
Town:	
Country:	
Tel.:	

UI Testing – A Person Editor

```
test_editPerson
        | person|
        person := Person withFirstName: 'Angelina'
                andLastName: 'Jolie'.
        self deny: (Person existsWhere:
                [:p| (p firstName = 'Bruce')
                & (p lastName = 'Willis')]).
        self openPersonEditorOn: person.
        self editField: #lastName value: 'Willis'.
        self editField: #firstName value: 'Bruce'.
        self toolBarPress: #saveCommand.
        self assert: (Person existsWhere:
                [:p | (p firstName = 'Bruce')
                & (p lastName = 'Willis')])
```



General

Last name:	Willis
First name:	Bruce
Sex:	
Nationality:	

Address

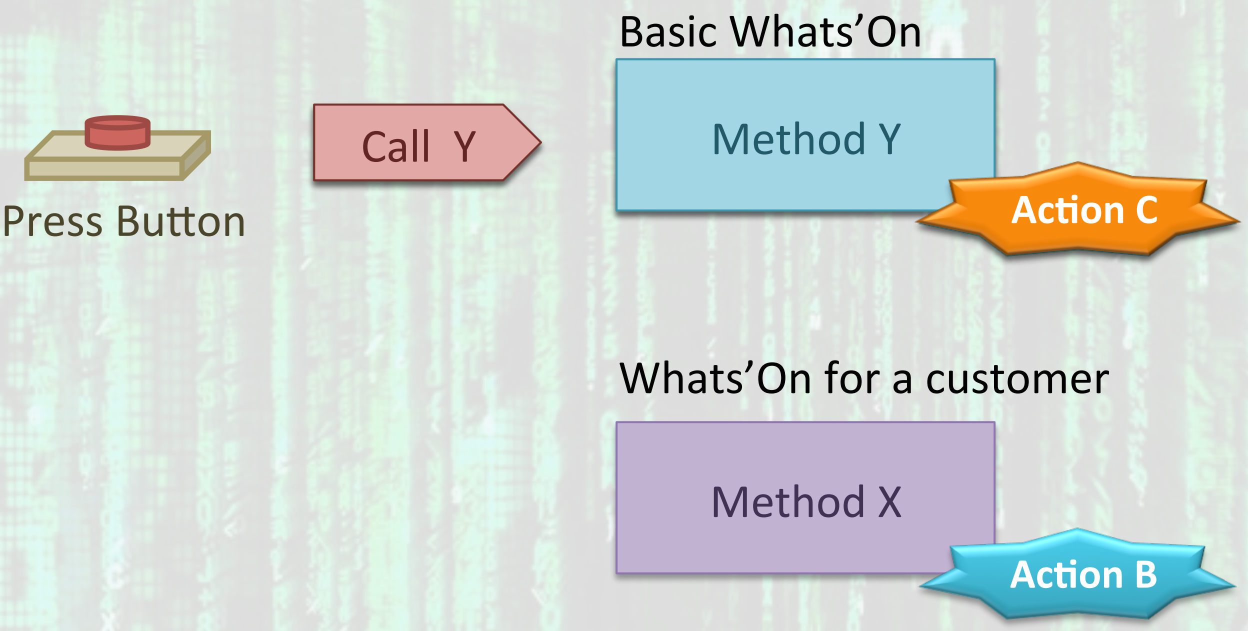
Type:	
Street:	
Town:	
Country:	
Tel.:	

UI Testing – A Person Editor

```
test_editPerson
        | person|
        person := Person withFirstName: 'Angelina'
                andLastName: 'Jolie'.
        self deny: (Person existsWhere:
                [:p| (p firstName = 'Bruce')
                & (p lastName = 'Willis')]).
        self openPersonEditorOn: person.
        self editField: #lastName value: 'Willis'.
        self editField: #firstName value: 'Bruce'.
        self toolBarPress: #saveCommand.
        self assert: (Person existsWhere:
                [:p | (p firstName = 'Bruce')
                & (p lastName = 'Willis')])
```

Refactoring tests

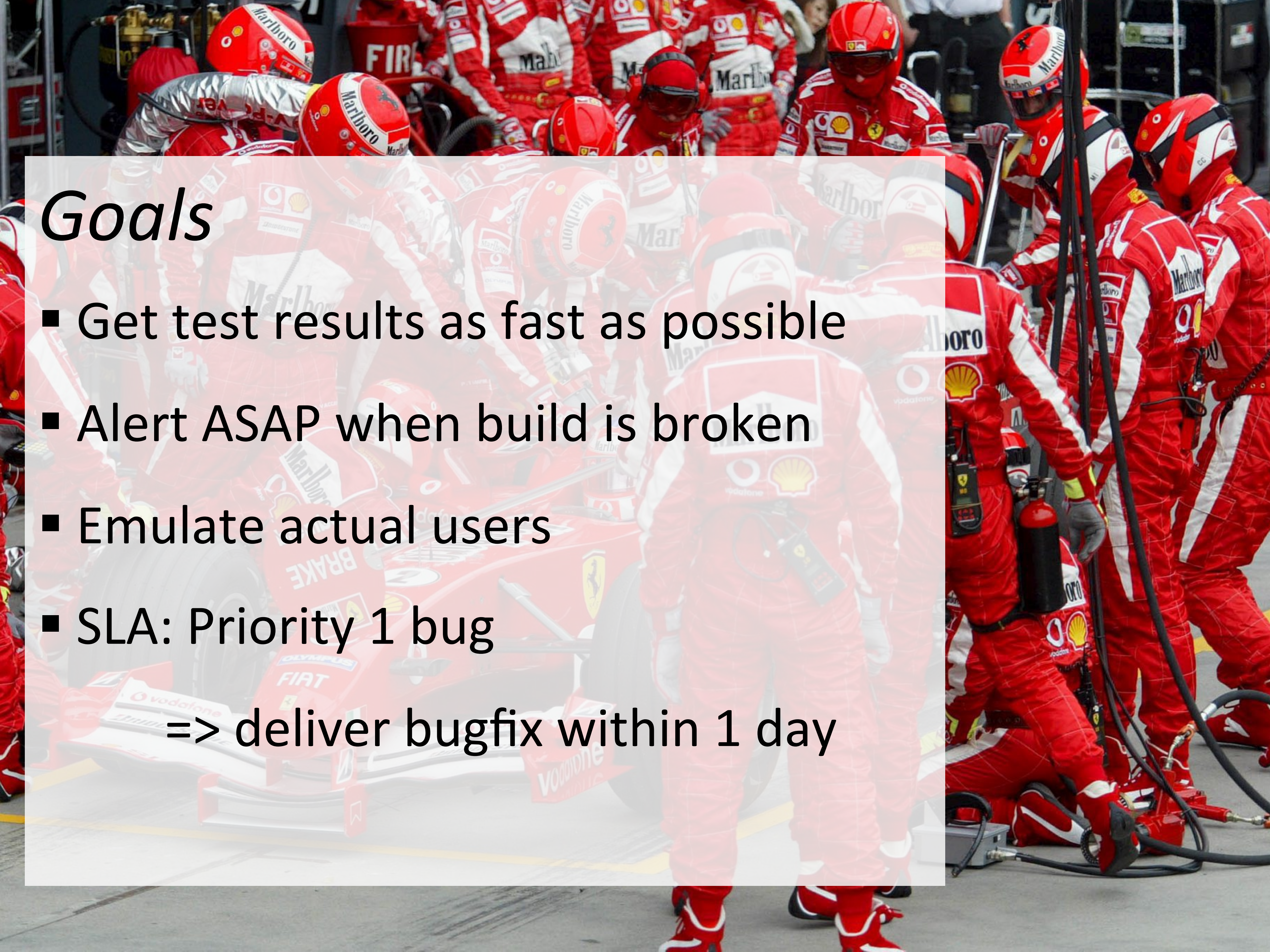
Why ?



=> Important during upgrades

Refactoring tests

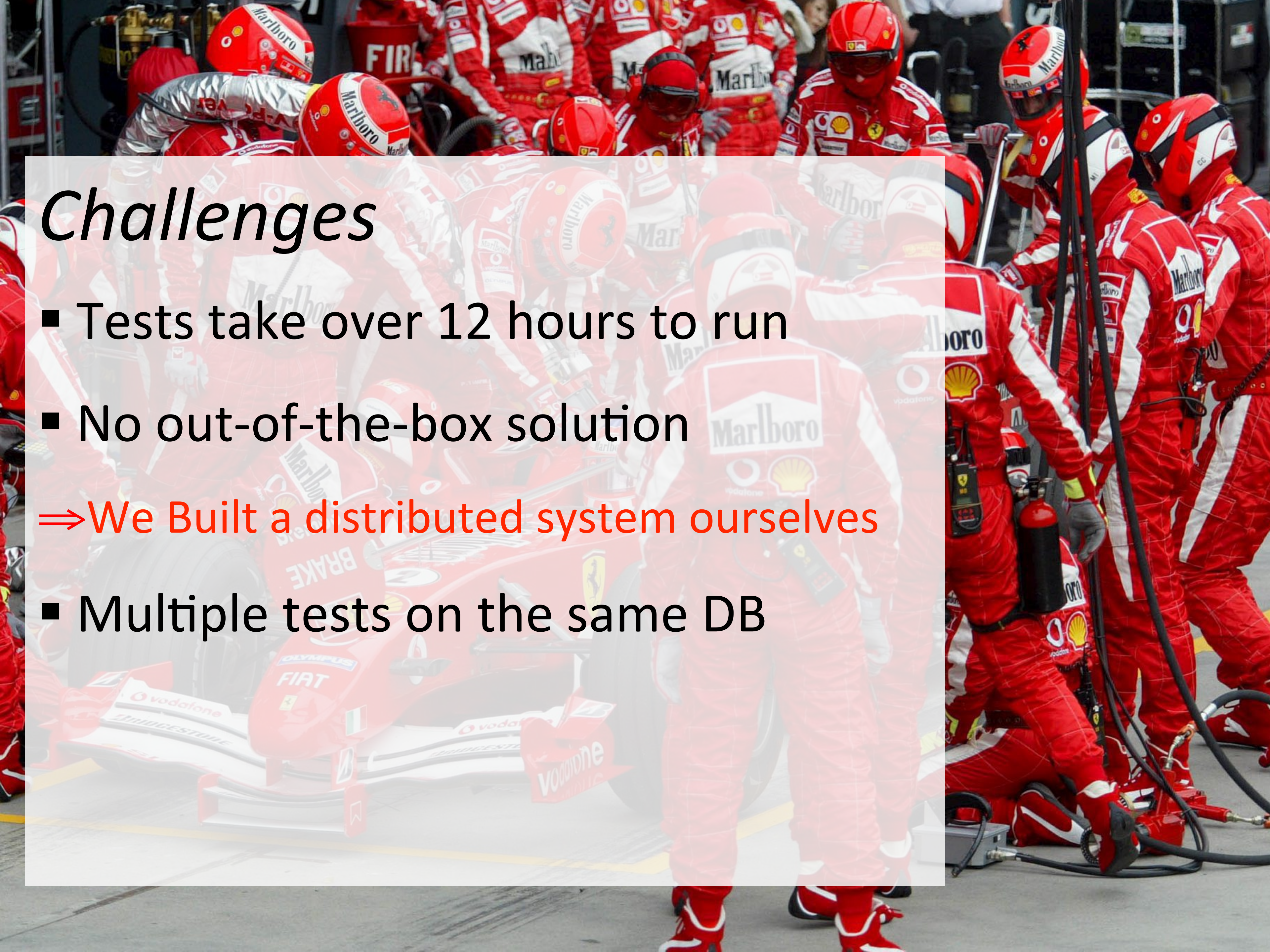
```
test_methodX_deprecated
	"use methodY"
	self assertNoImplementorsOf: ‘methodX‘.
	self assertNoReferencesTo: ‘methodX’.
```



Goals

- Get test results as fast as possible
- Alert ASAP when build is broken
- Emulate actual users
- SLA: Priority 1 bug

  => deliver bugfix within 1 day

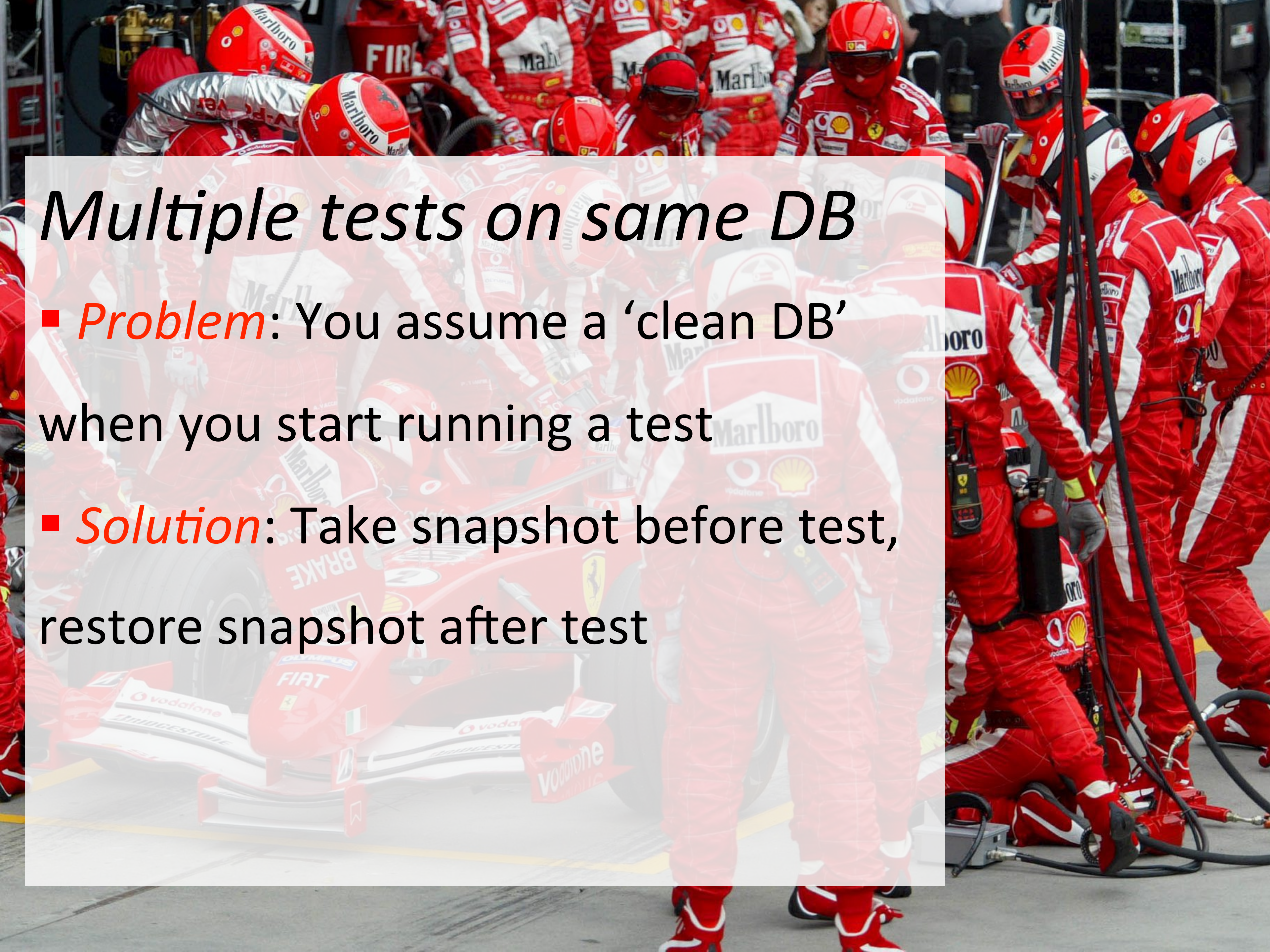


Challenges

- Tests take over 12 hours to run
- No out-of-the-box solution

⇒We Built a distributed system ourselves

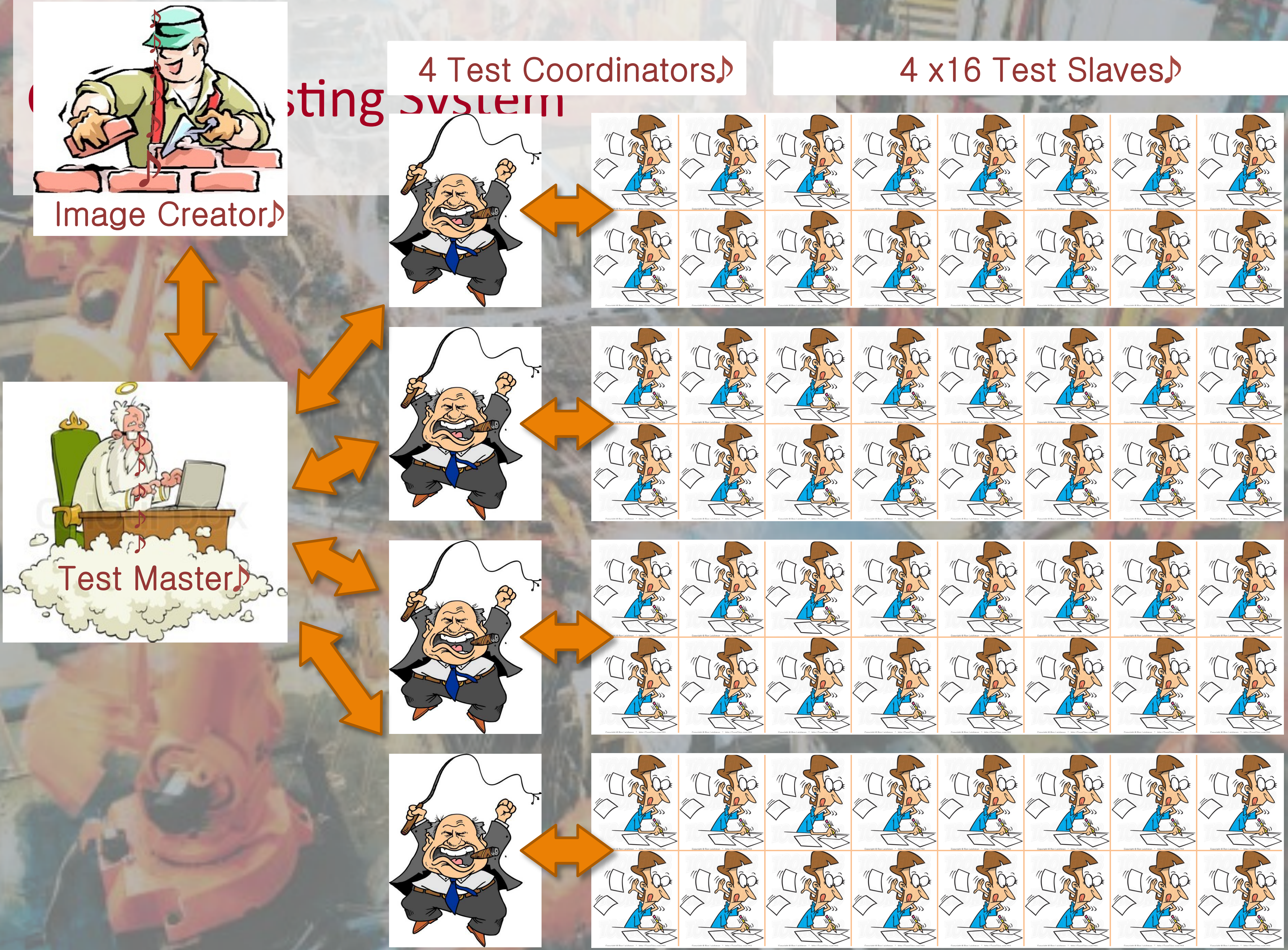
- Multiple tests on the same DB



Multiple tests on same DB

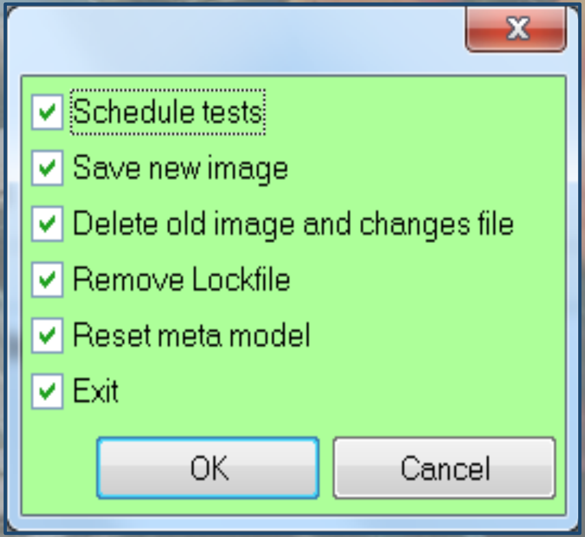
- *Problem*: You assume a 'clean DB' when you start running a test
- *Solution*: Take snapshot before test, restore snapshot after test

sting System

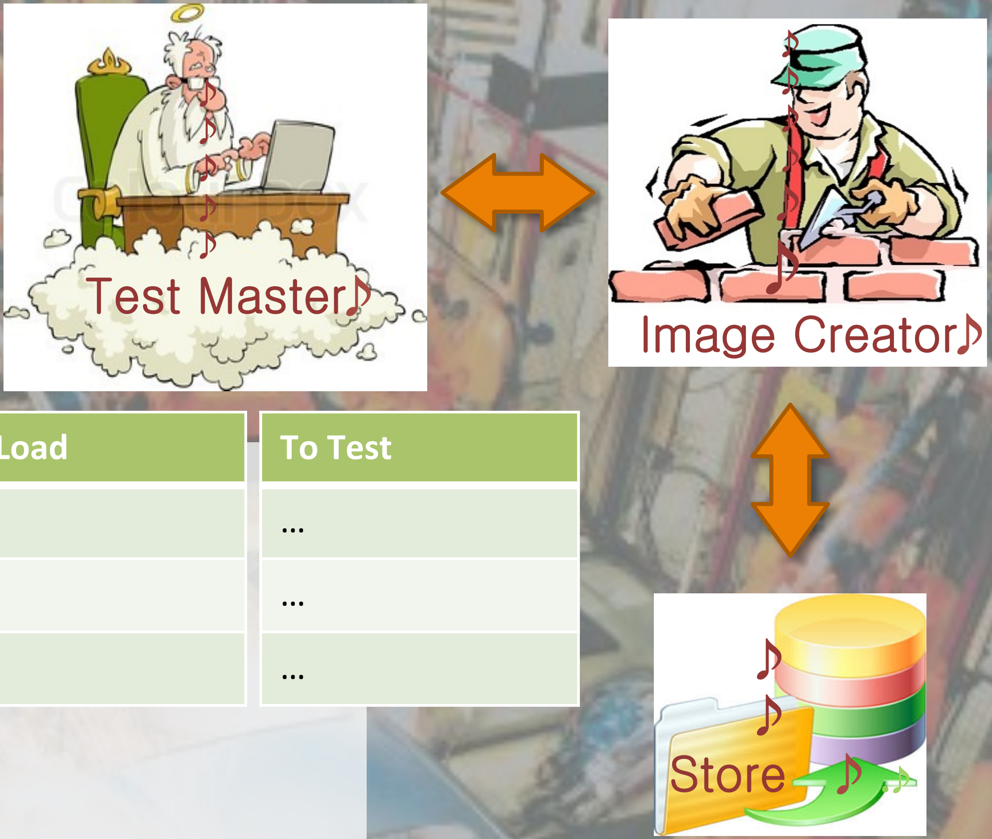


Testing a version

- Step by step:
  - The version has to be loaded
  - Create a database
  - The version has to be tested
  - The developer gets the results
- The test master orchestrates this whole process



- ✓ Schedule tests
- ✓ Save new image
- ✓ Delete old image and changes file
- ✓ Remove Lockfile
- ✓ Reset meta model
- ✓ Exit



To Load	To Test
…	…
…	…
…	…

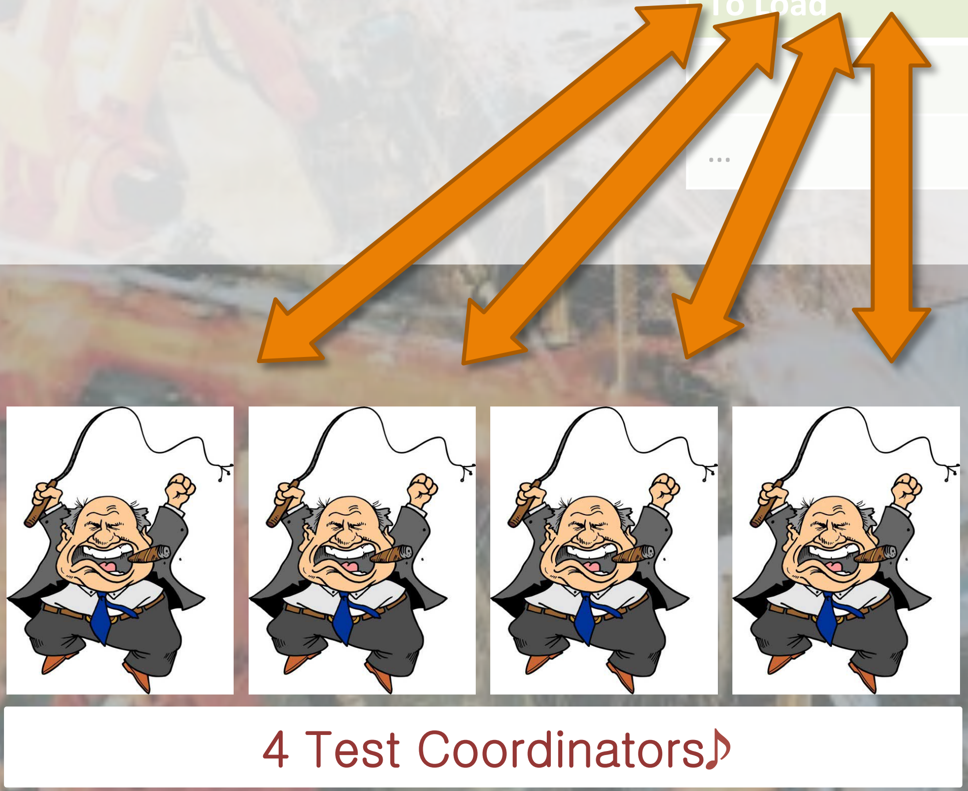
Image Creator

- TM instructs IC to load your version
- Get customers code from Store
- After loading
    - Puts the new image on a fileserver
    - IC notifies TM he's done

Test Coordinator

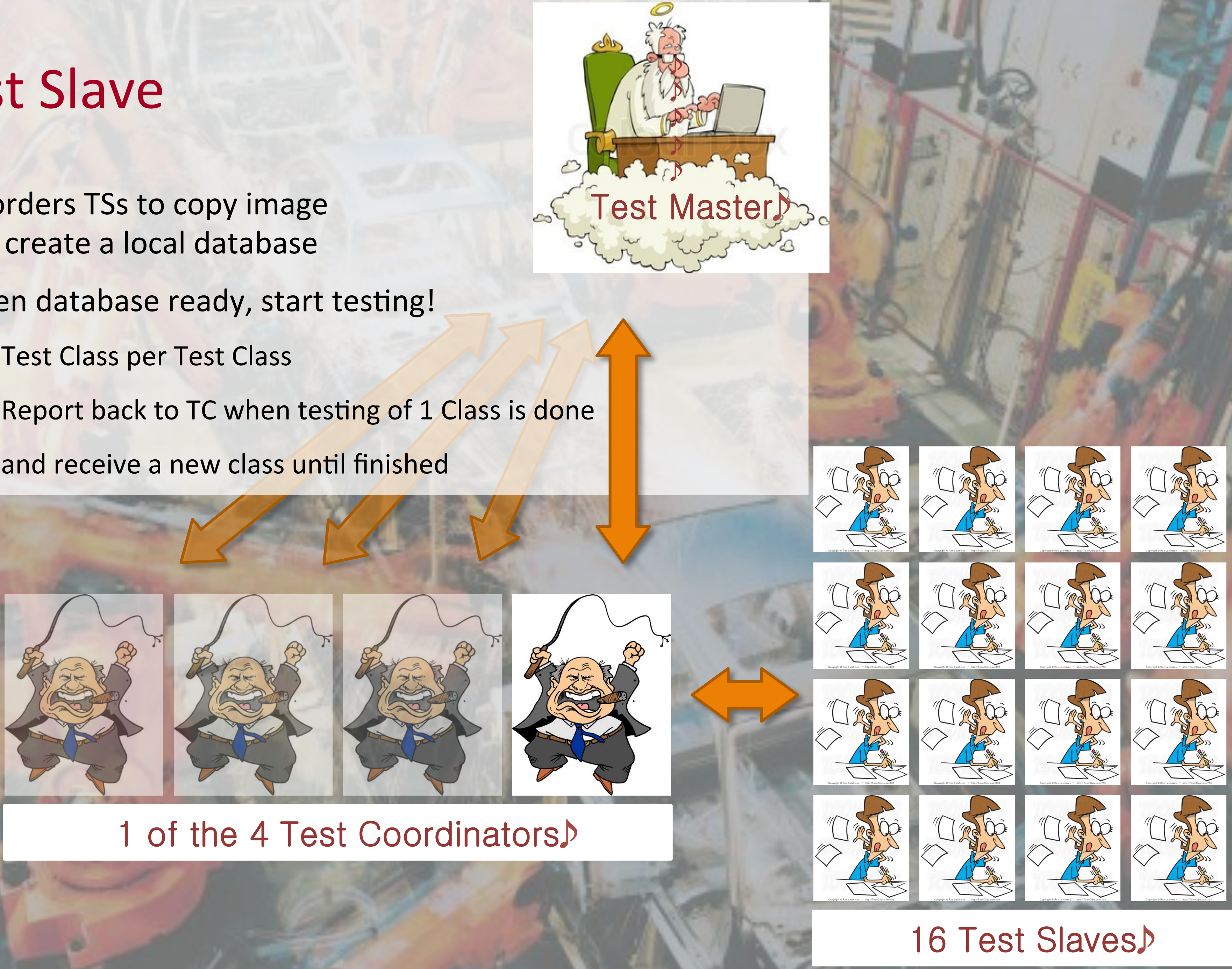
- TM instructs a TC to start testing a version
- A TM is responsible for 1 whole version

To Load	To Test
	Your version
…	…

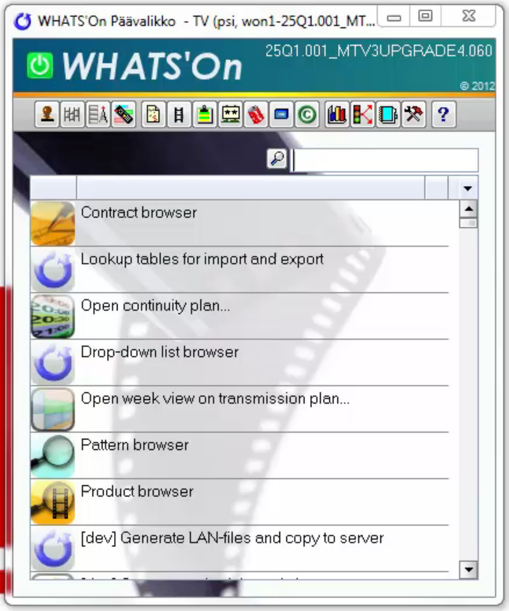


Test Slave

- TC orders TSs to copy image and create a local database
- When database ready, start testing!
  - Test Class per Test Class
  - Report back to TC when testing of 1 Class is done and receive a new class until finished



MediaGenix

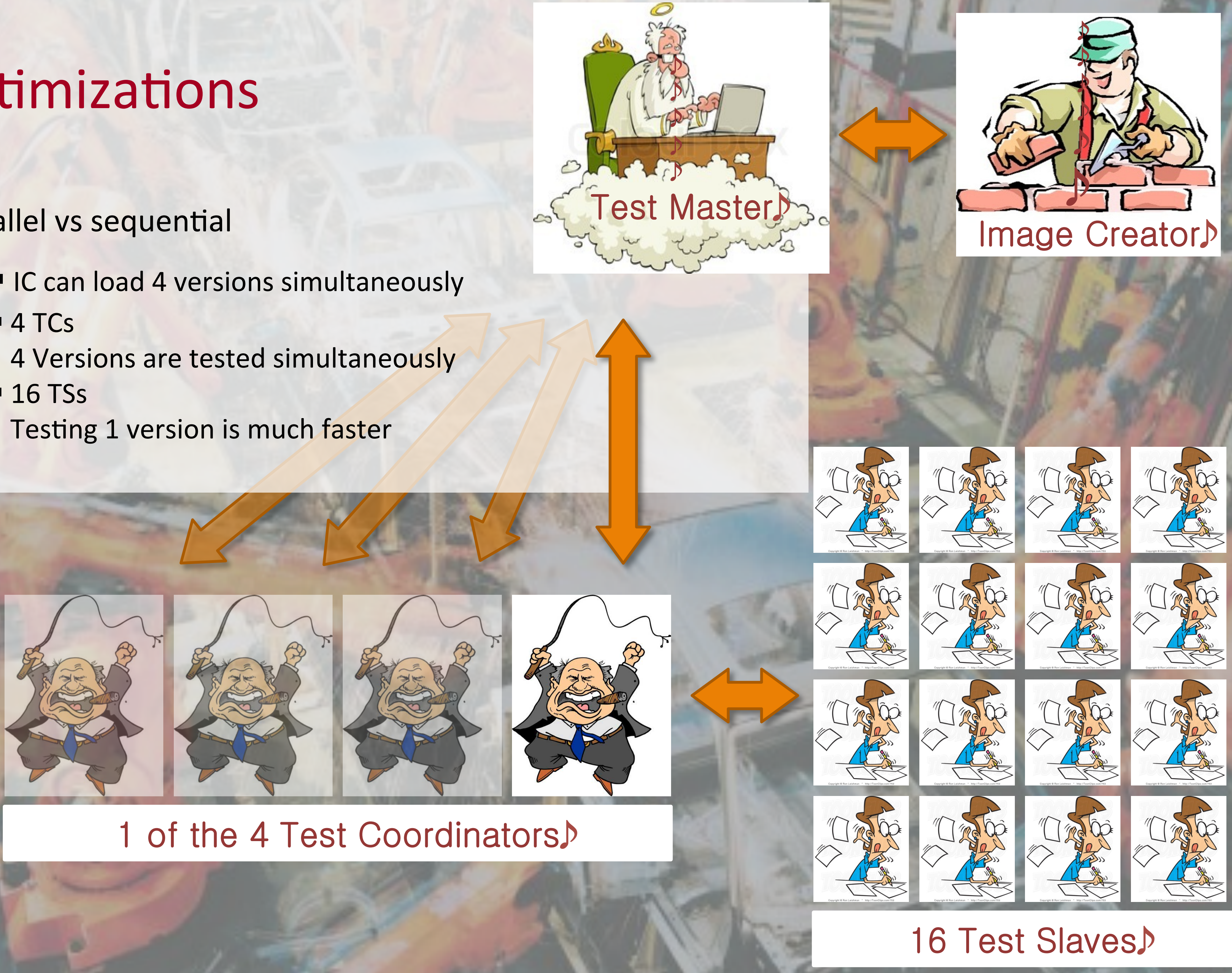

WHATS'On Päävalikko - TV (psi, won1-25Q1.001_MT...

WHATS'On 25Q1.001_MTV3UPGRADE4.060 © 2012

- Contract browser
- Lookup tables for import and export
- Open continuity plan...
- Drop-down list browser
- Open week view on transmission plan...
- Pattern browser
- Product browser
- [dev] Generate LAN-files and copy to server

Optimizations

- Parallel vs sequential
  - IC can load 4 versions simultaneously
  - 4 TCs
    4 Versions are tested simultaneously
  - 16 TSs
    Testing 1 version is much faster

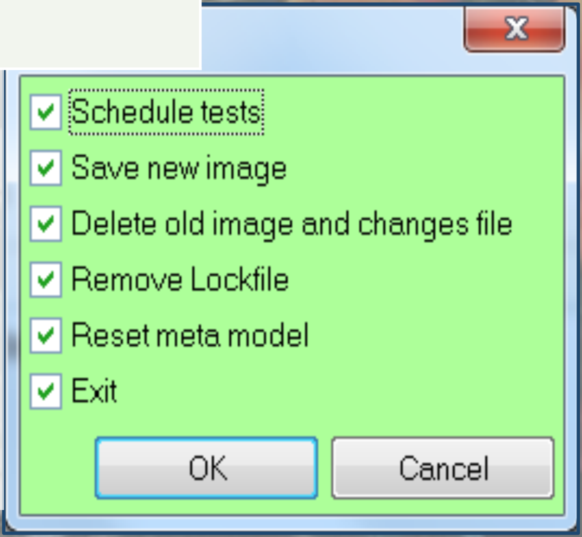


To Load
Version z for customer C
…

To Test
~~Version x for customer A~~
Version y for customer B
Version (x+1) for **customer A**
…

Optimizations

- Dialog 'Would you like to schedule the tests'
- Only test latest available version
- TM queues can be manipulated
  - Prioritize
  - Unschedule



- [✓] Schedule tests
- [✓] Save new image
- [✓] Delete old image and changes file
- [✓] Remove Lockfile
- [✓] Reset meta model
- [✓] Exit

Optimizations

- Time-outs
  - IC after 30minutes
  - TC after 75minutes
  - TS after +/- 10minutes

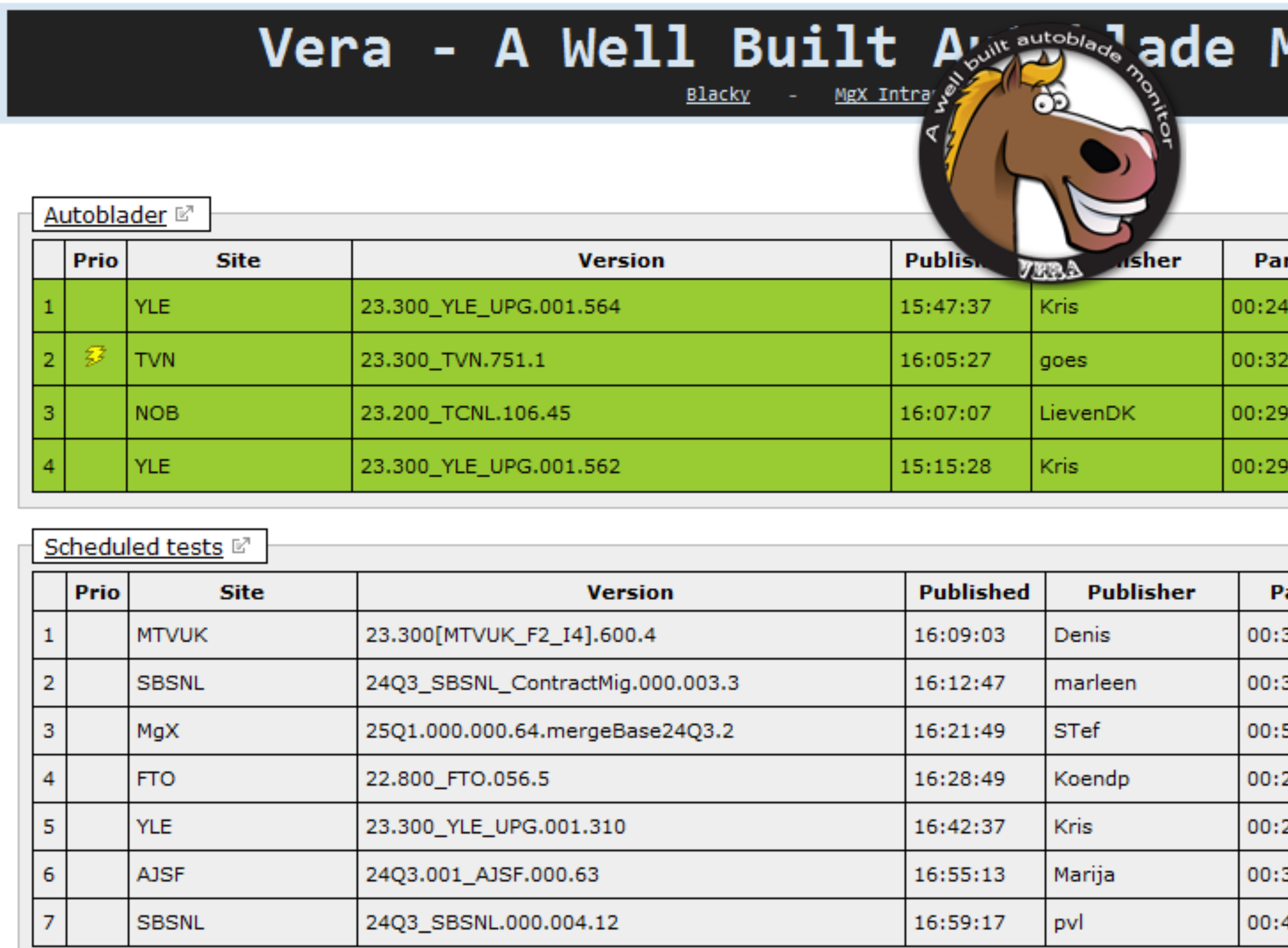
Optimizations

- Assign test classes in a smart order
- ‘Nuke’ tests

16 Test Slaves♪

Test Results

- What? Timing?
  - Build failed < 45minutes
  - After tests have successfully run
    - X total tests, Y failures, Z errors, T not run
    - With waiting in a queue +/- 4hours
    - Without waiting in a queue **< 1hour**
- **Saved on Store!**



Vera - A Well Built A lade M

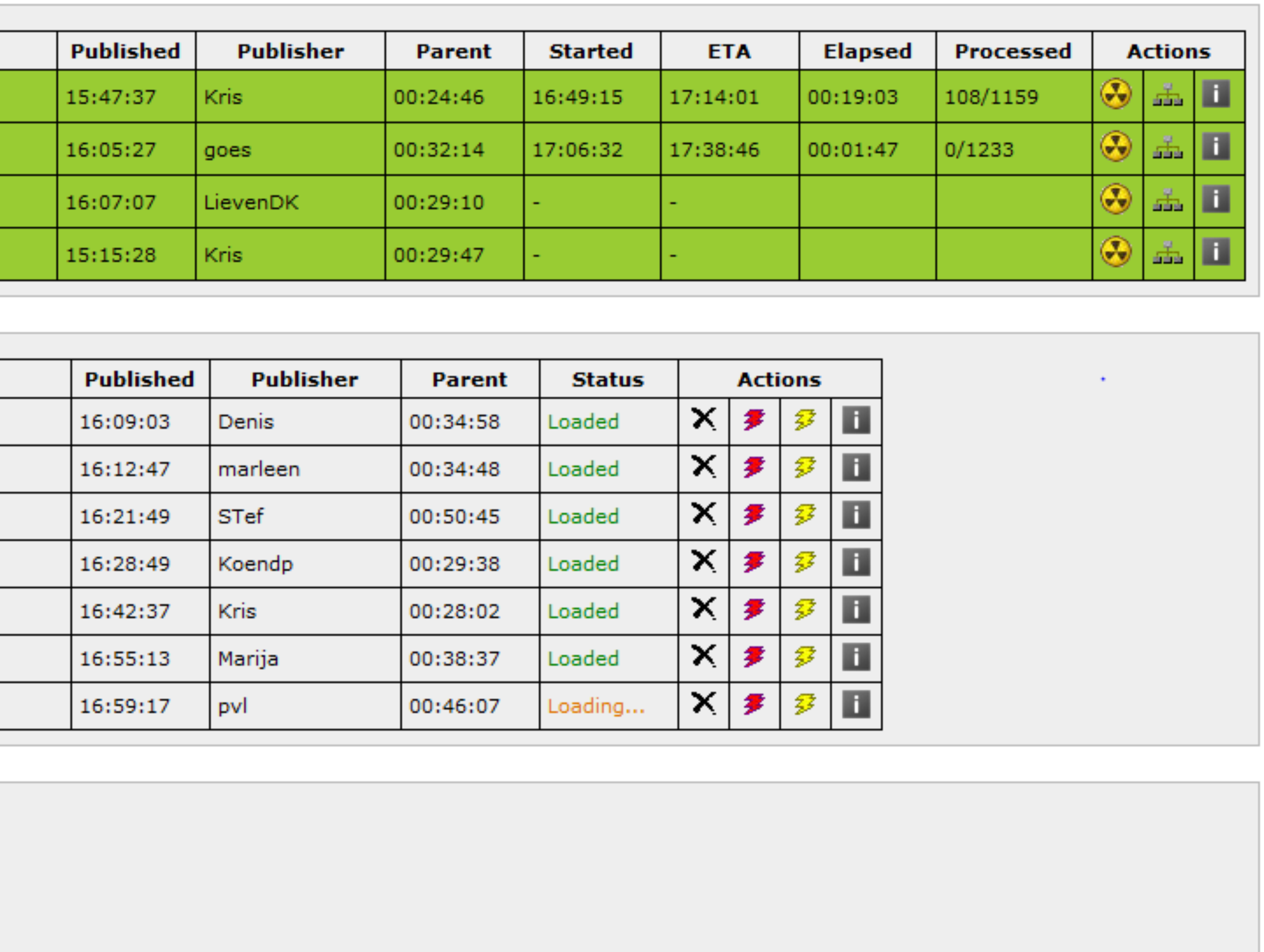
Blacky - MgX Intra

Autoblader

	Prio	Site	Version	Publis	isher	Par
1		YLE	23.300_YLE_UPG.001.564	15:47:37	Kris	00:24
2	⚡	TVN	23.300_TVN.751.1	16:05:27	goes	00:32
3		NOB	23.200_TCNL.106.45	16:07:07	LievenDK	00:29
4		YLE	23.300_YLE_UPG.001.562	15:15:28	Kris	00:29

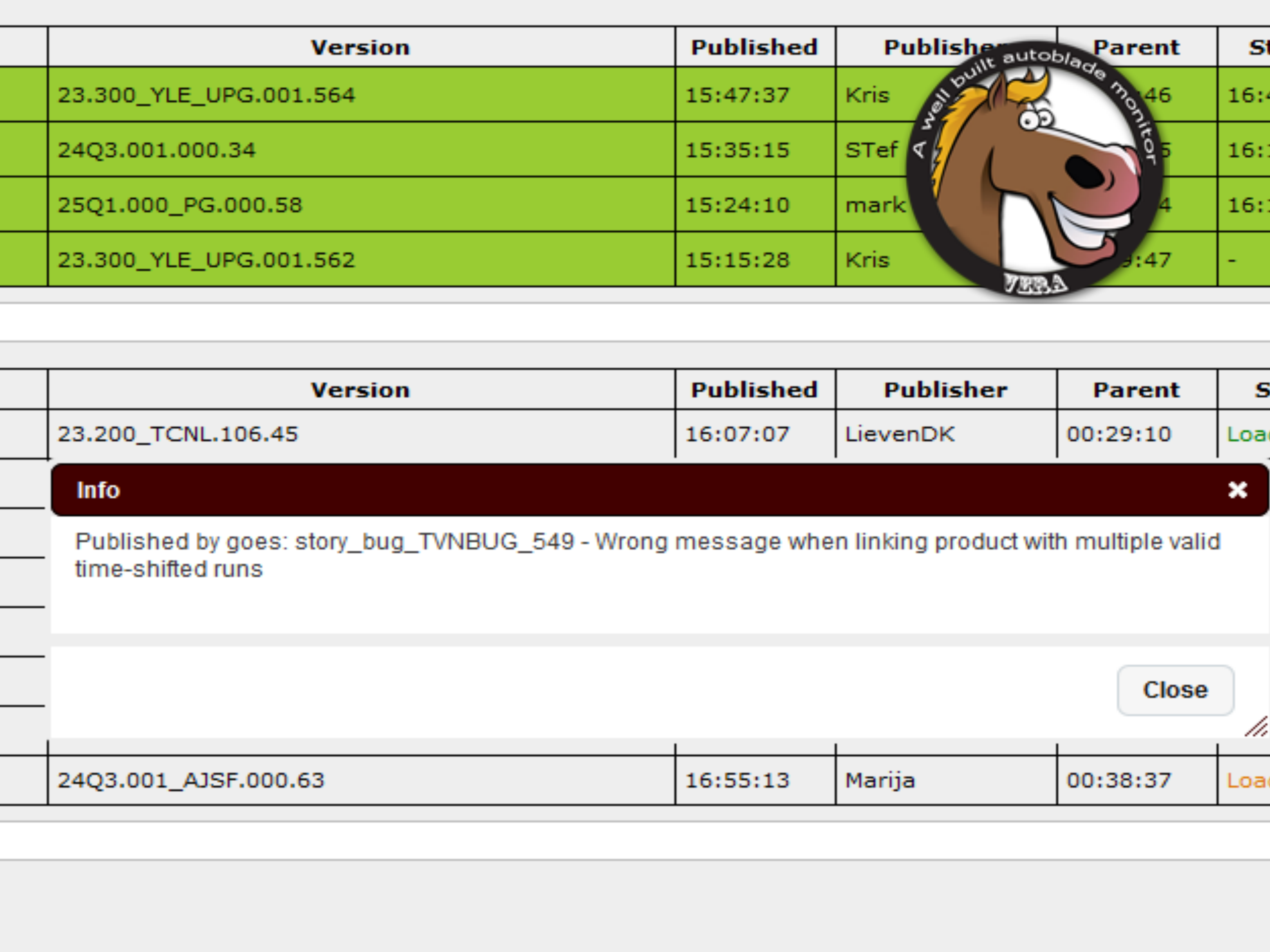
Scheduled tests

	Prio	Site	Version	Published	Publisher	Pa
1		MTVUK	23.300[MTVUK_F2_I4].600.4	16:09:03	Denis	00:3
2		SBSNL	24Q3_SBSNL_ContractMig.000.003.3	16:12:47	marleen	00:3
3		MgX	25Q1.000.000.64.mergeBase24Q3.2	16:21:49	STef	00:5
4		FTO	22.800_FTO.056.5	16:28:49	Koendp	00:2
5		YLE	23.300_YLE_UPG.001.310	16:42:37	Kris	00:2
6		AJSF	24Q3.001_AJSF.000.63	16:55:13	Marija	00:3
7		SBSNL	24Q3_SBSNL.000.004.12	16:59:17	pvl	00:4



Published	Publisher	Parent	Started	ETA	Elapsed	Processed	Actions		
15:47:37	Kris	00:24:46	16:49:15	17:14:01	00:19:03	108/1159	☢	🖧	ℹ
16:05:27	goes	00:32:14	17:06:32	17:38:46	00:01:47	0/1233	☢	🖧	ℹ
16:07:07	LievenDK	00:29:10	-	-			☢	🖧	ℹ
15:15:28	Kris	00:29:47	-	-			☢	🖧	ℹ

Published	Publisher	Parent	Status	Actions			
16:09:03	Denis	00:34:58	Loaded	✕	⚡	⚡	ℹ
16:12:47	marleen	00:34:48	Loaded	✕	⚡	⚡	ℹ
16:21:49	STef	00:50:45	Loaded	✕	⚡	⚡	ℹ
16:28:49	Koendp	00:29:38	Loaded	✕	⚡	⚡	ℹ
16:42:37	Kris	00:28:02	Loaded	✕	⚡	⚡	ℹ
16:55:13	Marija	00:38:37	Loaded	✕	⚡	⚡	ℹ
16:59:17	pvl	00:46:07	Loading...	✕	⚡	⚡	ℹ

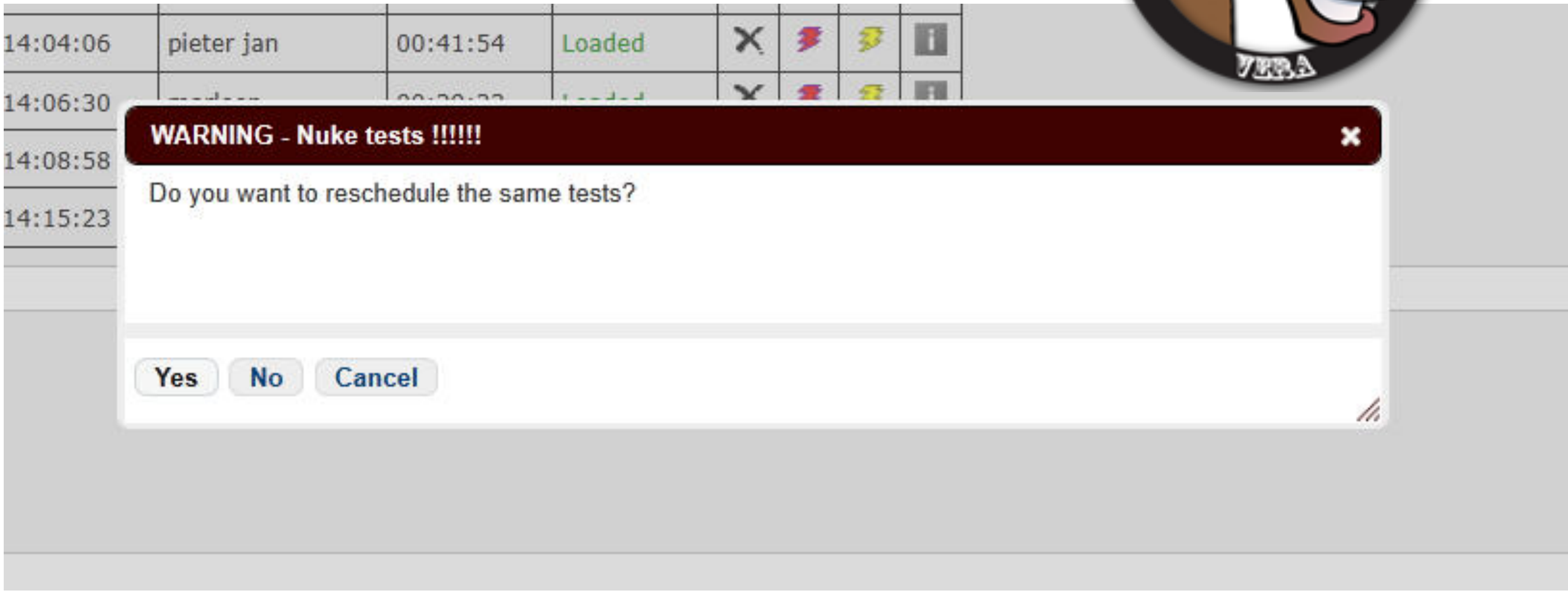


Version	Published	Publisher	Parent	St
23.300_YLE_UPG.001.564	15:47:37	Kris	46	16:
24Q3.001.000.34	15:35:15	STef	5	16:
25Q1.000_PG.000.58	15:24:10	mark	4	16:
23.300_YLE_UPG.001.562	15:15:28	Kris	:47	-

Version	Published	Publisher	Parent	S
23.200_TCNL.106.45	16:07:07	LievenDK	00:29:10	Loa
24Q3.001_AJSF.000.63	16:55:13	Marija	00:38:37	Loa

Info

Published by goes: story_bug_TVNBUG_549 - Wrong message when linking product with multiple valid time-shifted runs

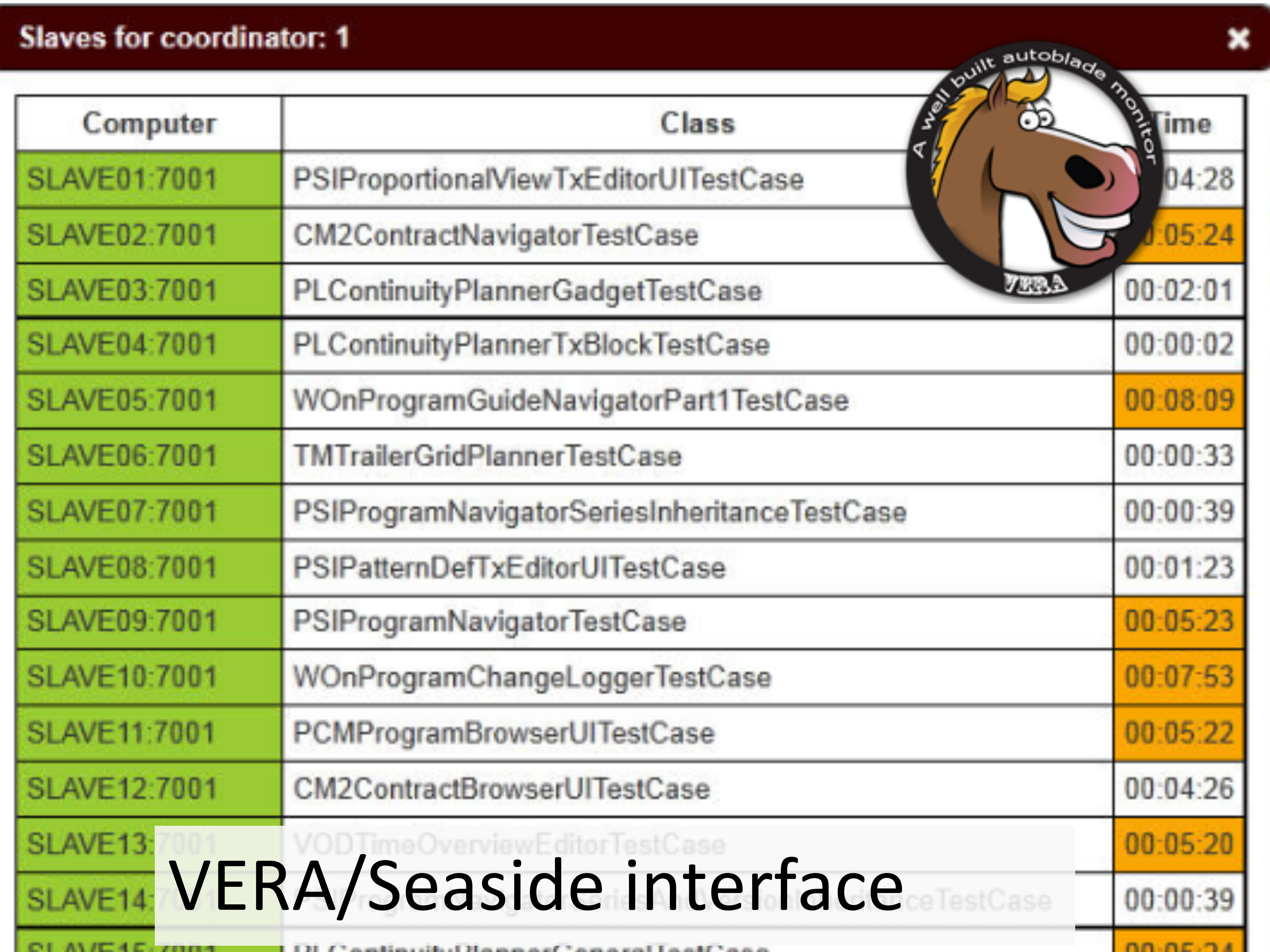


14:04:06	pieter jan	00:41:54	Loaded
14:06:30			
14:08:58			
14:15:23			

WARNING - Nuke tests !!!!!!

Do you want to reschedule the same tests?

VERA/Seaside interface



Slaves for coordinator: 1

A well built autoblade monitor

VERA

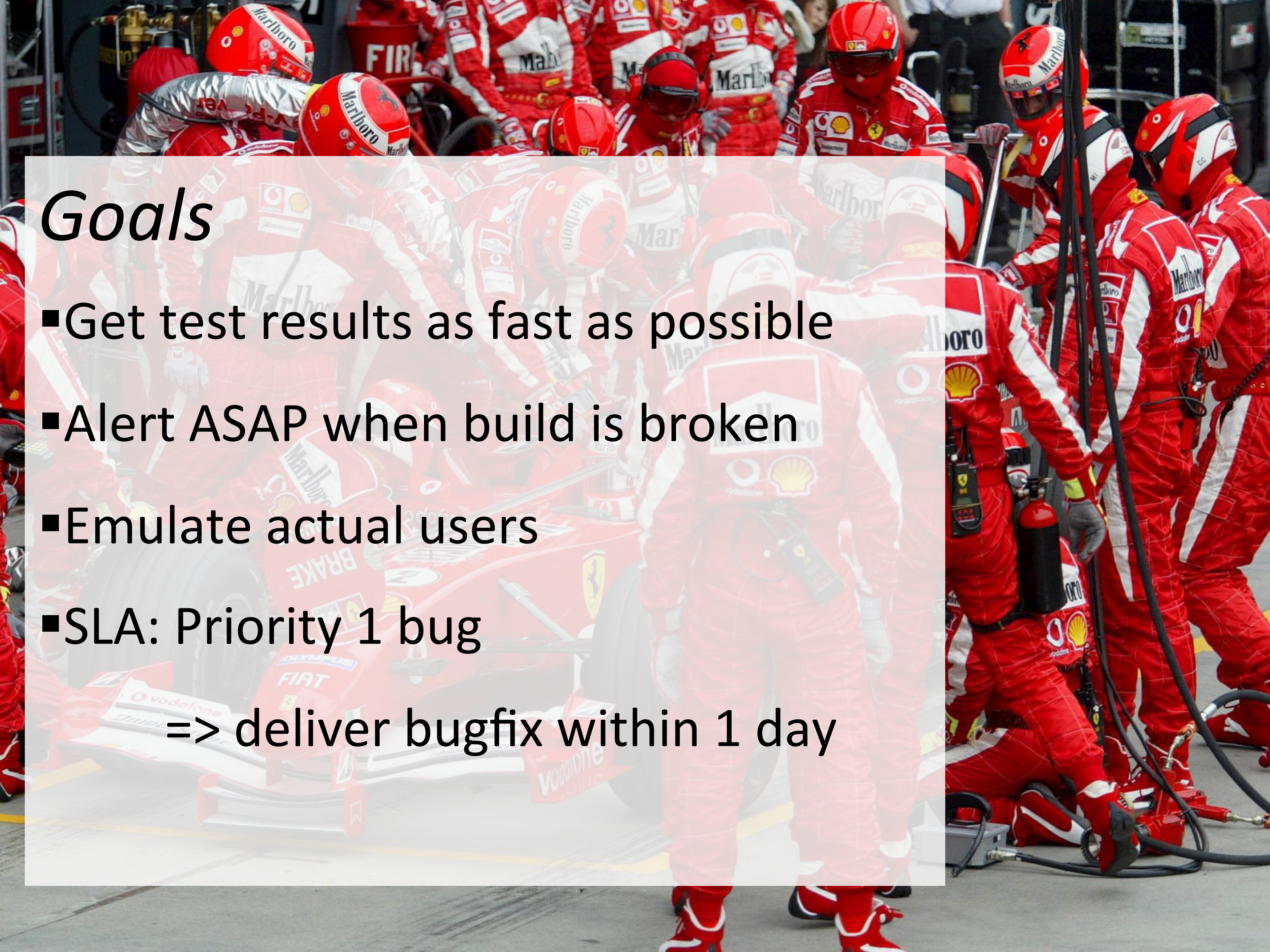
Computer	Class	Time
SLAVE01:7001	PSIProportionalViewTxEditorUITestCase	04:28
SLAVE02:7001	CM2ContractNavigatorTestCase	:05:24
SLAVE03:7001	PLContinuityPlannerGadgetTestCase	00:02:01
SLAVE04:7001	PLContinuityPlannerTxBlockTestCase	00:00:02
SLAVE05:7001	WOnProgramGuideNavigatorPart1TestCase	00:08:09
SLAVE06:7001	TMTrailerGridPlannerTestCase	00:00:33
SLAVE07:7001	PSIProgramNavigatorSeriesInheritanceTestCase	00:00:39
SLAVE08:7001	PSIPatternDefTxEditorUITestCase	00:01:23
SLAVE09:7001	PSIProgramNavigatorTestCase	00:05:23
SLAVE10:7001	WOnProgramChangeLoggerTestCase	00:07:53
SLAVE11:7001	PCMProgramBrowserUITestCase	00:05:22
SLAVE12:7001	CM2ContractBrowserUITestCase	00:04:26
SLAVE13:7001	VODTimeOverviewEditorTestCase	00:05:20
SLAVE14:7001	[illegible]ceTestCase	00:00:39
SLAVE15:7001	PLContinuityPlannerGeneralTestCase	00:05:24

VERA/Seaside interface

Integrated Tools

- Open SUnit on failed tests
- Find origin failing test

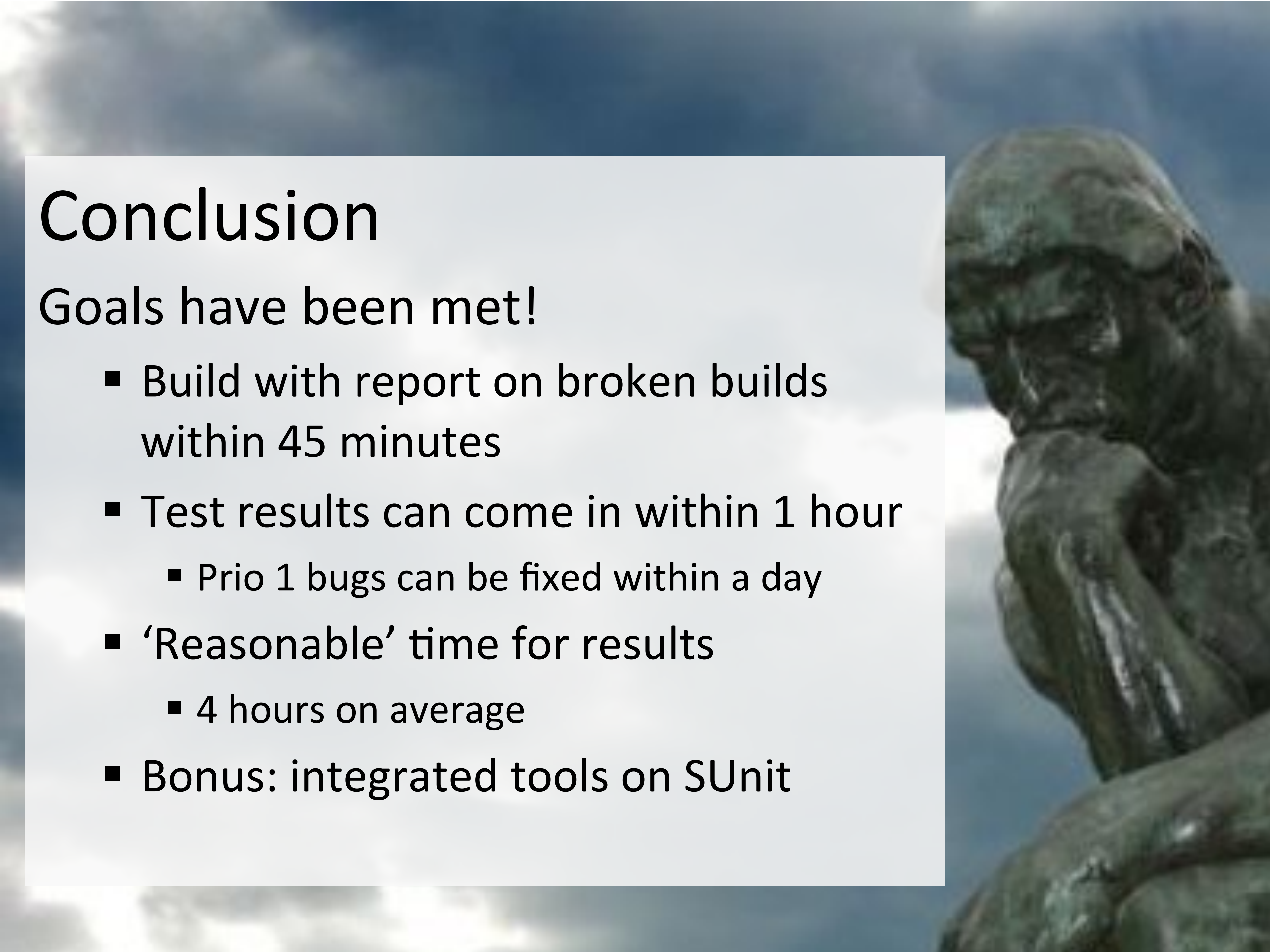
Demo



Goals

- Get test results as fast as possible
- Alert ASAP when build is broken
- Emulate actual users
- SLA: Priority 1 bug

  => deliver bugfix within 1 day



Conclusion

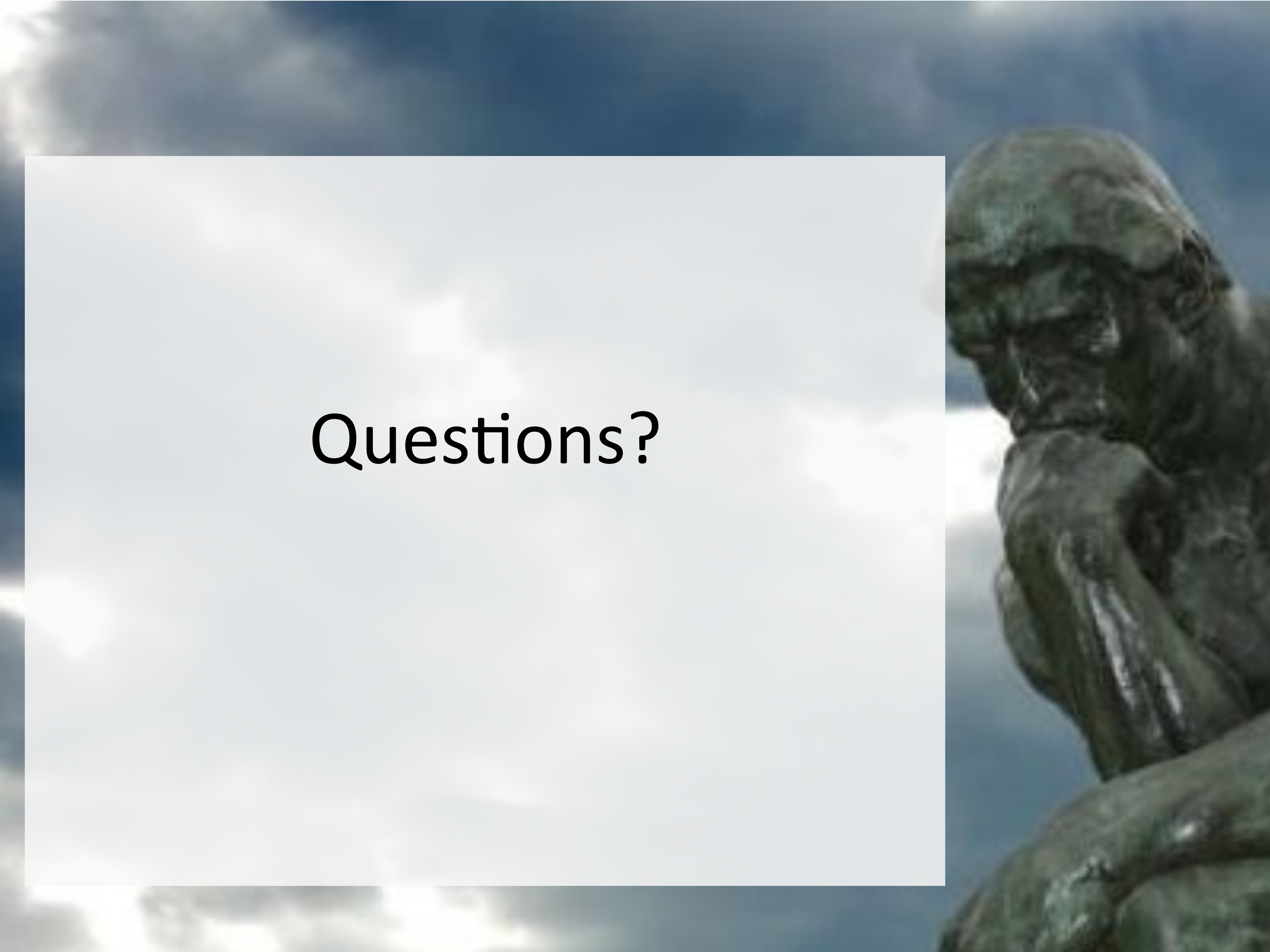
Goals have been met!

- Build with report on broken builds within 45 minutes
- Test results can come in within 1 hour
  - Prio 1 bugs can be fixed within a day
- ‘Reasonable’ time for results
  - 4 hours on average
- Bonus: integrated tools on SUnit

Conclusion

Future areas of improvement …

- Test every version
  - Better pin-point where an error started
- Speed up through-put of unprioritised tests



Questions?

Q: Which customisations do customers want?

A: See next presentation

Q: Why does the product need to change that often?

A: Technical evolution, target different markets, new regulations, …