

# Automated Application Tests for Lotus Notes

---

or

How to rest easy during the next roll out

Oct. 7 2009, Sponsor Session



# About us

## Lucius Bobikiewicz



**Managing Director  
Smart-Toucan GmbH**

[Lucius.Bobikiewicz@Smart-Toucan.com](mailto:Lucius.Bobikiewicz@Smart-Toucan.com)

[www.smart-toucan.com](http://www.smart-toucan.com)

## Christian Klümper



**Head of test management & test  
development at proClients GmbH**

[Christian.Kluemper@proClients.de](mailto:Christian.Kluemper@proClients.de)

[www.proClients.com](http://www.proClients.com)



# Live Demo of AutoUser

The screenshot shows two overlapping windows. The background window is 'Luca 8 - Inbox - IBM Lotus Notes', displaying a mail inbox with a list of messages from 'Luca 8'. The foreground window is 'AutoUser Creator - Developer Edition', which is a script editor. It contains a VBS script named '001\_Switch\_ID.vbs' with the following code:

```
72  
73 ShowUp "Accessing menu command 'Switch ID '"  
74 Click menuFile  
75 Click menuSecurity  
76 Click menuSwitchID  
77  
78 ShowUp "Entering filename for sample ID", 3  
79 ' bug workaround  
80 KeyPress Key.Backspace  
81 ' end  
82 WriteTo navigationFile, "John.id"  
83 Click buttonOpen  
84 sleep 2  
85  
86 ShowUp "Entering a mock password", 3  
87 WriteTo textPassword, "xxxxx"  
88 sleep 2
```

Below the code is an 'Execution Log' field and a 'Ready' status indicator. The AutoUser Creator window also features a menu (File, Edit, Help), a toolbar with 'Delay' (0), 'On Error' (Stop), and 'SHOW ALL' buttons, and a control panel with 'RUN', 'SINGLE STEP', and 'STOP' buttons.



# QA for complex or mission-critical Lotus Notes applications

- **Function tests**
  - Dependencies
  - Workflow tests
  - Role permissions
- **Load test**

Improve your database design before launching it
- **Performance tests**

Live monitoring 24/7 - exchange vague user feedback for hard data



# QA based on automated tests provides significant advantages

- **Reliable**  
No human errors when working through the QA case manual
- **Fast**  
Automated testing is significantly faster than manual testing
- **Repeatable**  
Tests can be executed as often as needed, no manpower bottleneck
- **Verifiable and traceable**  
You get machine-generated, reproducible results  
(Sarbanes-Oxley anyone??)



# Case study: Automated tests for mission-critical software at proClients GmbH

---

**Application to be tested:** Domino Storage Optimizer

**Installations (Example):** DFS/German Flight Control

**QA level required:** High

**Responsible QA Engineer:** Christian Klümper

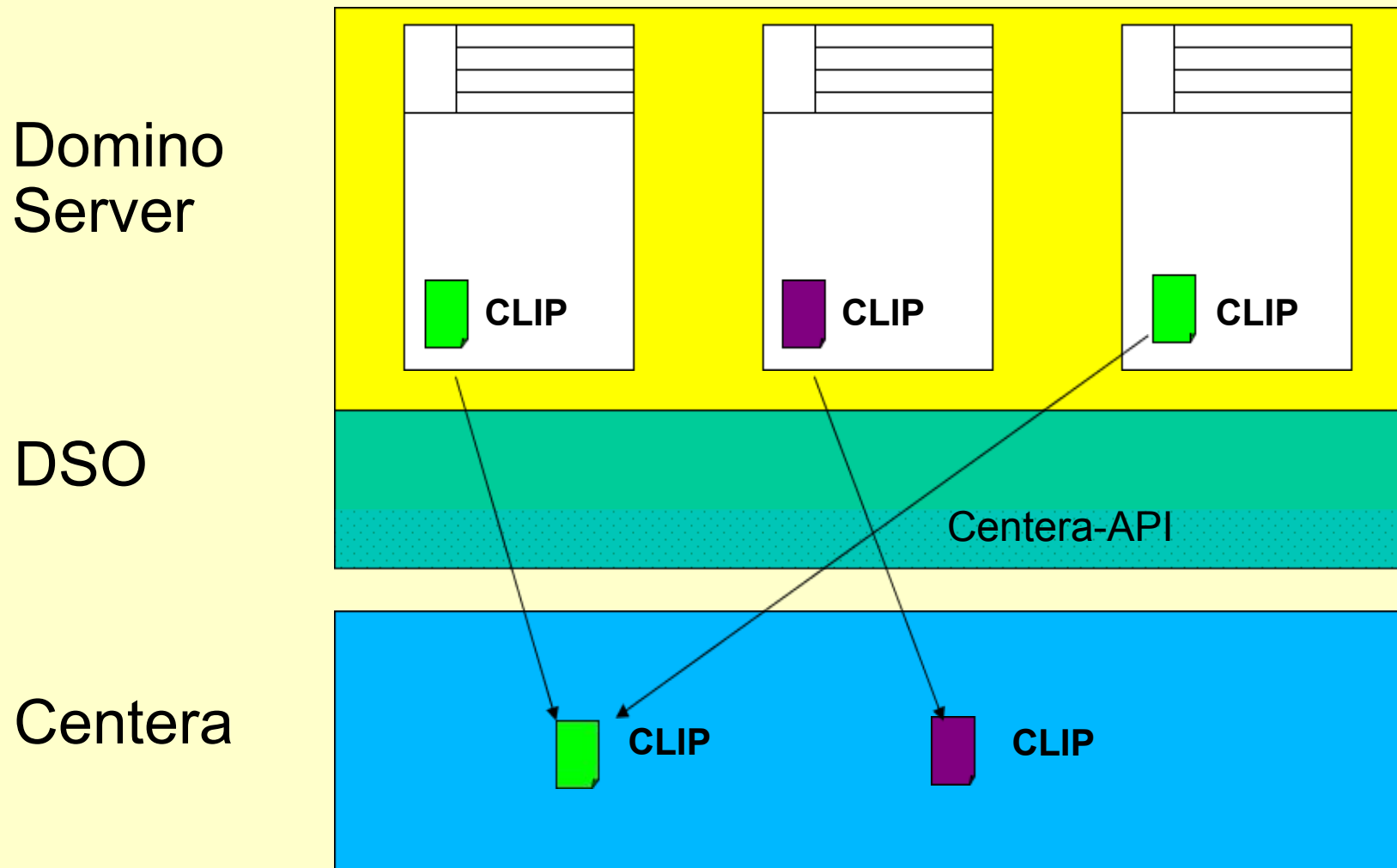


# Domino Storage Optimizer (DSO)

- **Reduces data volume** in Notes databases dramatically - approximately 80% of data volume for mail databases are allocated for mail attachments
- **DSO moves attachments** from Domino applications **to the storage system** Centera by EMC<sup>2</sup>
- Identical attachments will be **stored only once** but referenced as often as needed
- **Automated restoration** of attachments **at runtime** when user reopens an optimised document

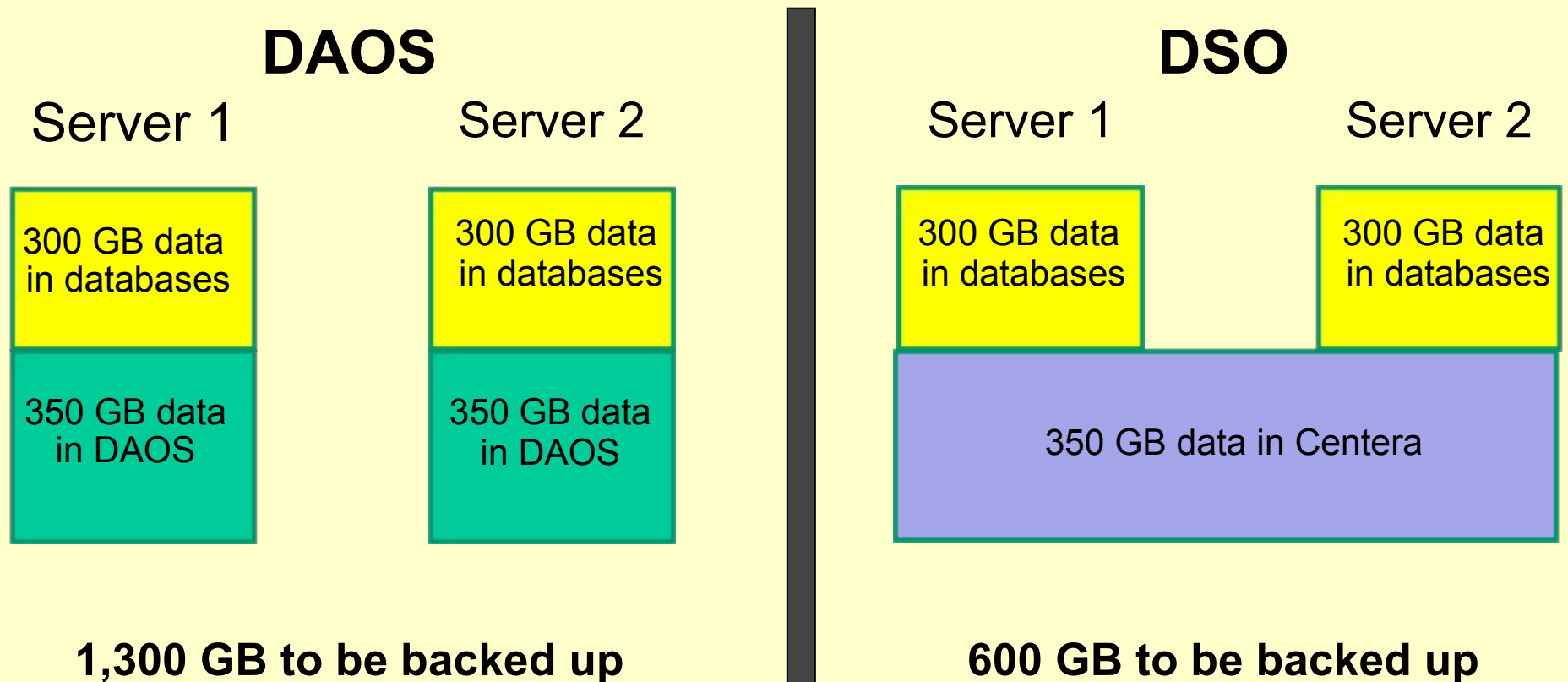


# System architecture of Domino Storage Optimizer



# DSO vs. DAOS

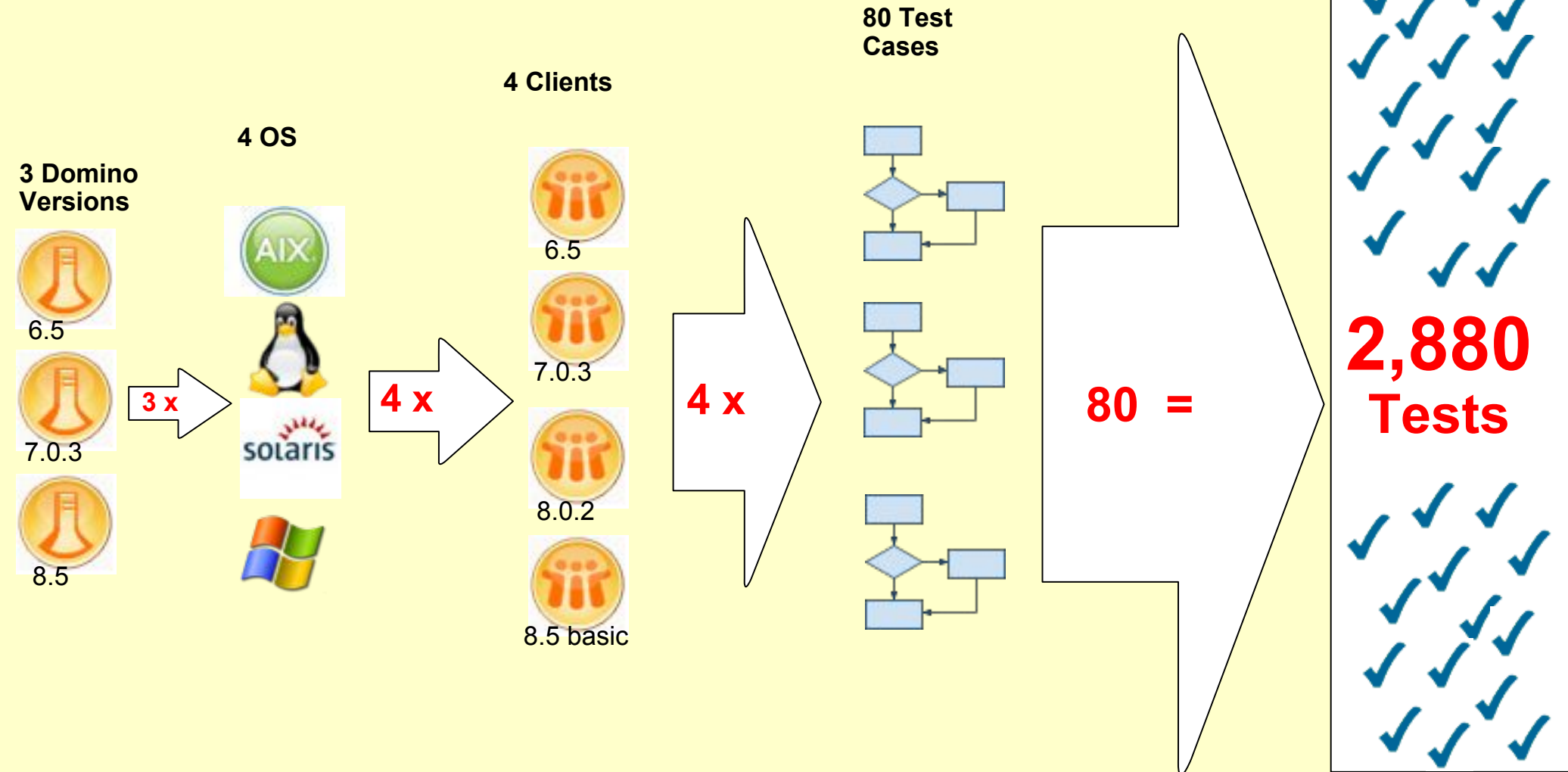
## (1 Domino cluster with 2 x 1 terabyte)



**700 GB less to back up - every day!**



# The test scenario for DSO

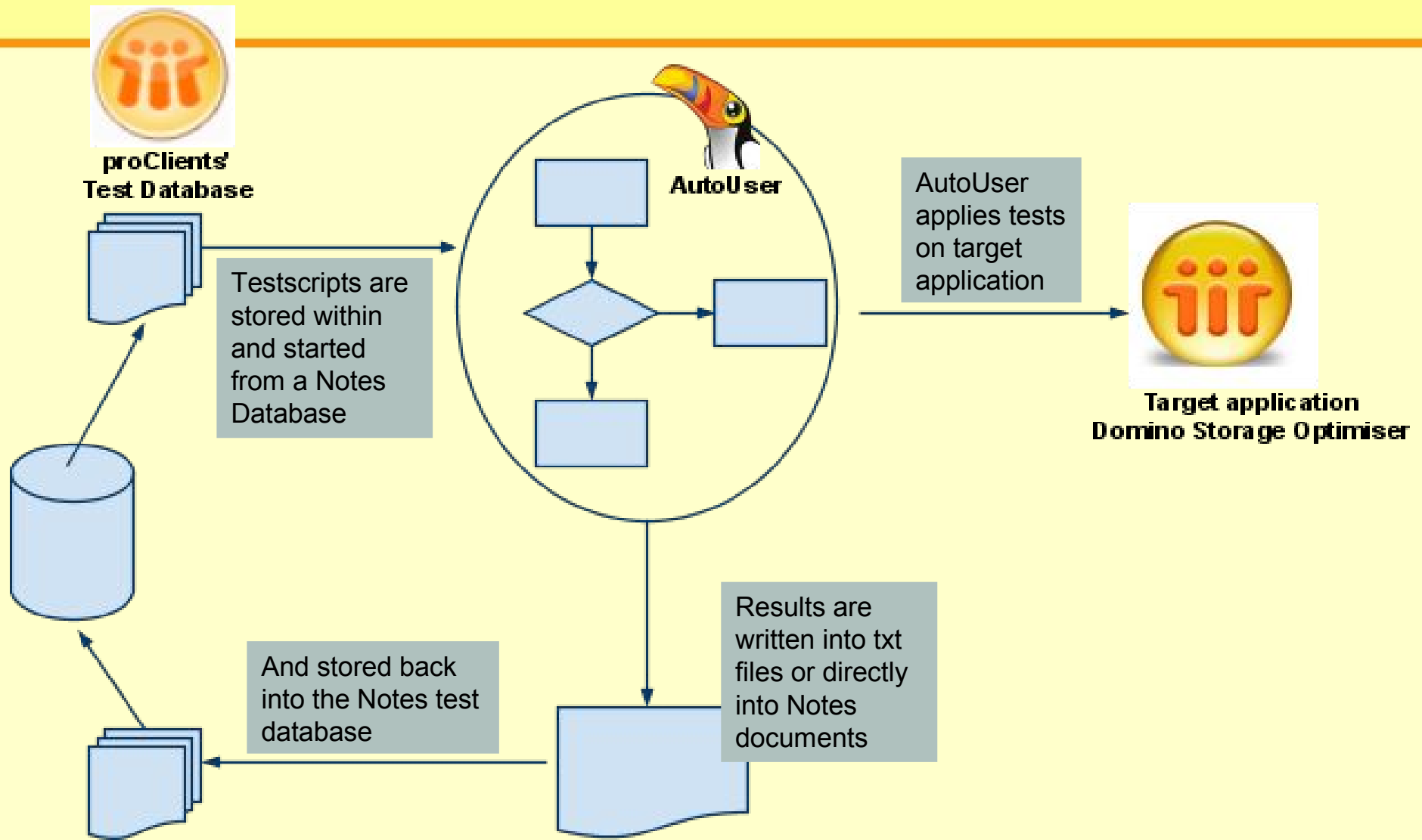


# Our QA approach for DSO

- **80 defined test cases with detailed**
  - start conditions
  - actions to be performed
  - expected results
- **5 categories of test cases**
  - document actions
  - deletion actions
  - attachment actions
  - mailing actions
  - database actions
- **Outcome of all tests runs are recorded, history is traceable**



# Architecture of the automated test system



# Automated testing is ~8 to 10 times faster

- **Manual testing**

80 tests require **12 hours**

All 2,880 tests required (\*) **54.0 days**

- **Automated testing**

80 test take **1.5 hours**

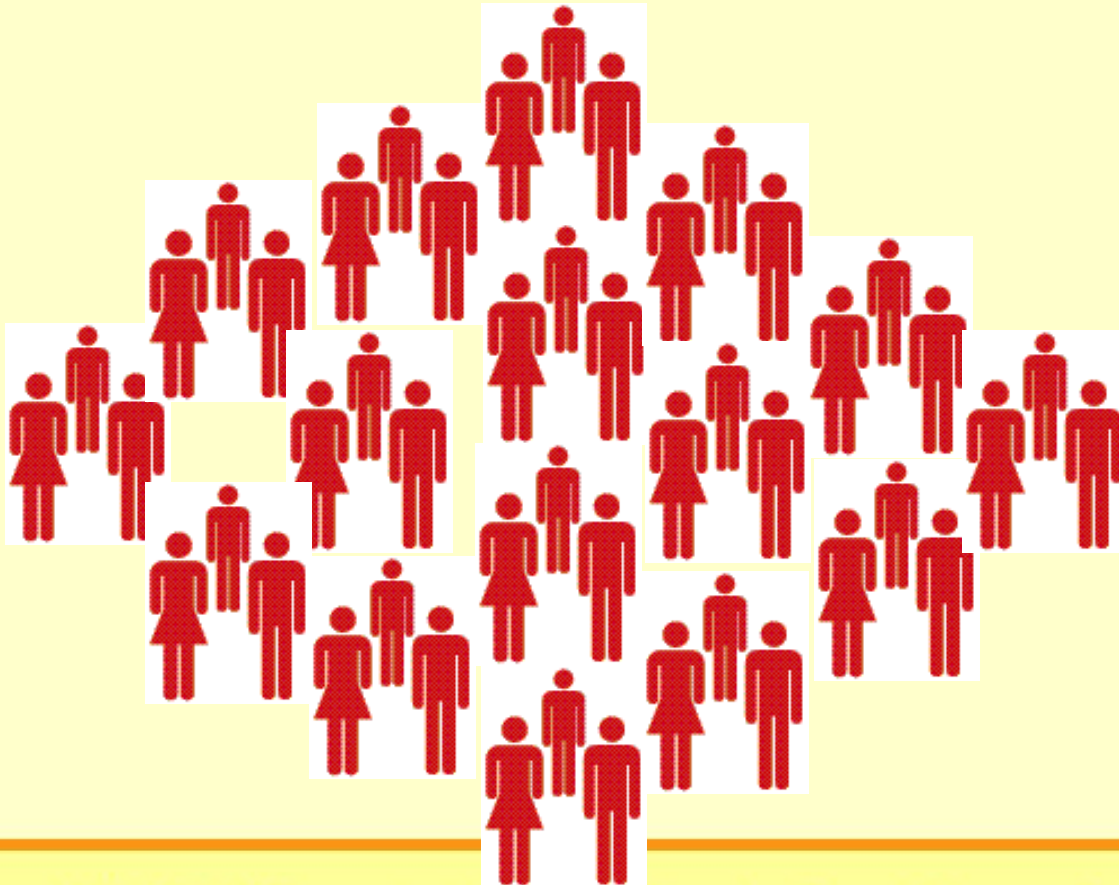
All 2,880 test take **2.2 days**

(\*) in manual testing we calculate 8h per work day

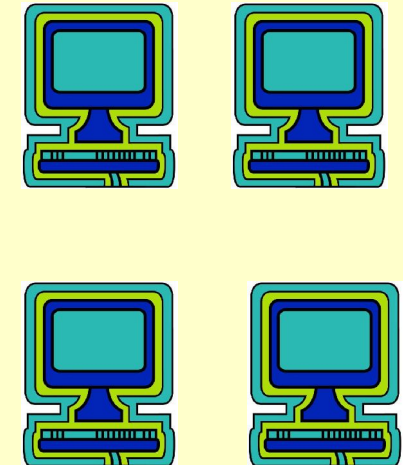


# AutoUser enables, for the first time, full-scale testing for every build

Not an option:  
50 testers working  
night shift



No problem at all:  
Running  
4 PCs



# Amortisation will be achieved after one single test run

## Tasks

- creating a library with general test functions
- writing control scripts for all test case categories
- integrating these scripts into the test case database

## Total investment for implementation

- about 500 to 600 hours

## Amortization

- one (1!) complete test run of the 2,880 testcases almost amortises the investment



# Lessons learned

- **Administration** of test cases / test definitions (not scripts) requires a tool of its own (at proClients, this is a Notes database)
- **A single repository** helps a lot. We store test definitions, scripts, and results into a single, centralised database.
- **Test execution should be as flexible as possible, run**
  - all test cases
  - only test cases of one category
  - only one single test case
- **Test libraries also need to be designed**  
Maintenance and integration of new tests should be a no-brainer



# Impact on the future development processes at proClients GmbH

After having implemented the "basic" test scenario, we are going to:

- Increase the number of tests
- Enhance the complexity of tests
- Implement additional test scenarios
- Set up auto-execution for all cases in all scenarios
- Implement load tests
- Set up continuous performance monitoring

**QA has now turned from bug searching into an active process of improving the quality of our software!**



# Smart-Toucan's AutoUser

- **can dramatically reduce costs for executing test case**
- **enables completely new QA strategies**
- **provides new business opportunities for IBM partners**

For more information, please contact:  
Lucius Bobikiewicz@Smart-Toucan.com

(For more information on DSO,  
meet Christian at the Smart-Toucan booth)



# Thank you

