Quality Assurance For Mathematical Modeling Systems

Armin Pruessner, Michael Bussieck,
Steven Dirkse, Alex Meeraus

GAMS Development Corporation
1217 Potomac Street NW
Washington, DC 20007

ICS 2005, Annapolis

Agenda

- Motivation
- Definitions and Components
- Challenges
- Software Quality Assurance at GAMS
- Testing new Solver Links
- Client Model Testing
- Summary

Motivation

Quality Assurance

- Essential component in most industries
- Important in most software engineering sectors

Mathematical Programming

- Less attention to quality assurance (small community)
- Specific QA issues for modeling systems (initially expensive)
- Different focus for industry and academic

Focus on reliability

Focus on performance

Definitions

Quality: *The totality of features and characteristics of a product or service that bear on its ability to satisfy specified or implied needs* (ISO 8402)

Software Quality Assurance (SQA): "*Set of systematic activities providing evidence of the ability of the software process to produce a software product that is fit to use*" (Schulmeyer and McManus)

Definitions
cont‘d

Key components of SQA (which includes monitoring of *products* and *processes*):

- Software configuration management (SCM): All activities related to version control and change control
- Quality control and testing: monitoring the products
  - Focus on the quality of product ***within each phase*** of the software development lifecycle
  - Objective: identify and remove defects throughout the lifecycle, as ***early as possible***

Components

Software configuration management: All activities related to version control and change control

Goals:

- Identify and control changes
- Ensure that change is being properly implemented
- Report changes in the software to others who may need to know of them

GAMS

Components

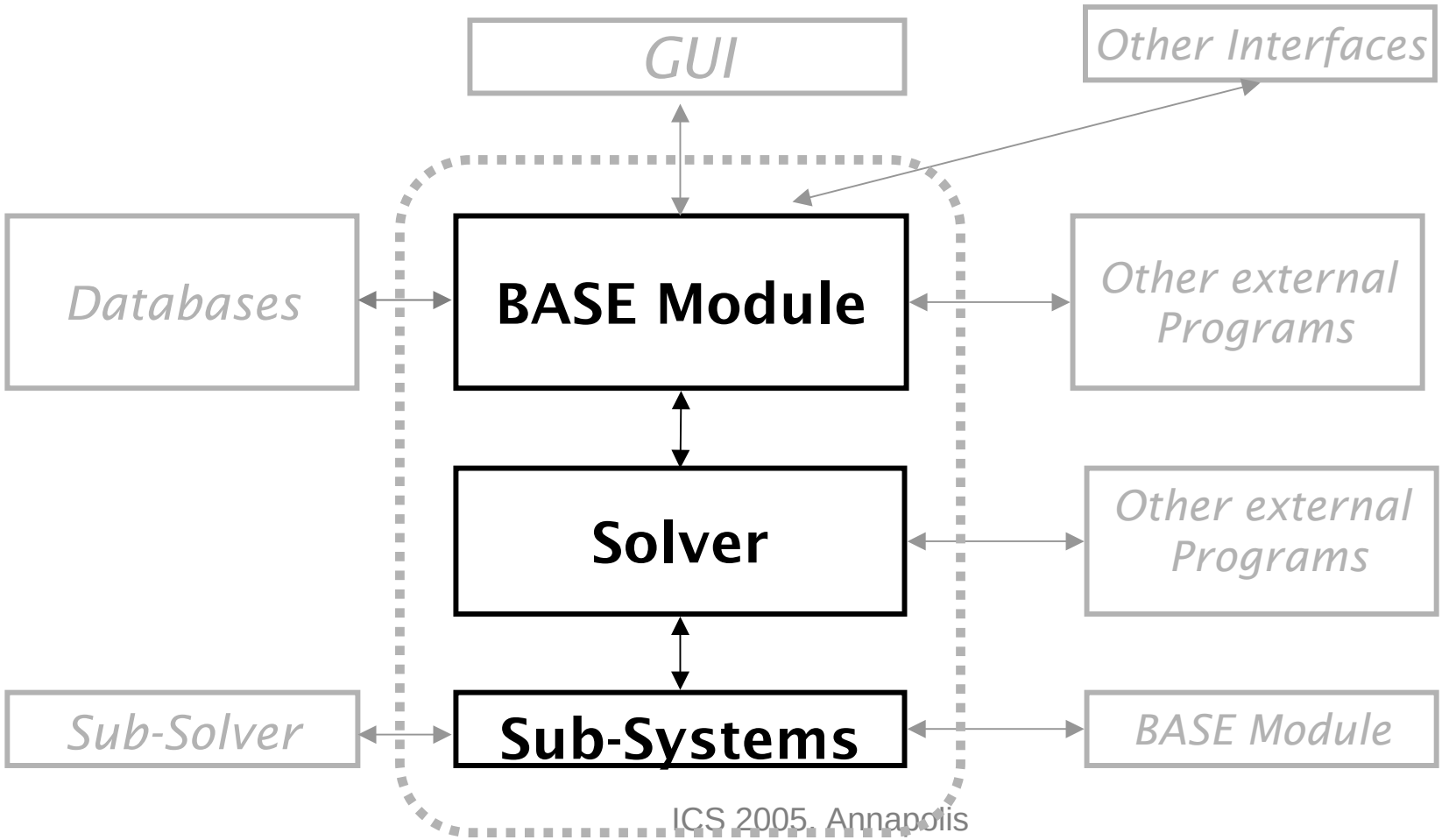
Quality control and testing of the product

- Unit testing (initial tests by developer)
- Regression testing (interaction with other modules)
- System Integration Testing (full scale testing)
- Metrics
- Bug tracking tools

Goal: Uncover defects during the complete life cycle of the software as *early as possible*

Algebraic Modeling Systems

Architecture



Algebraic Modeling Systems

Basic Principles

- Separation of model and solution methods
- Balanced mix of declarative and procedural approaches
- Computing platform independence
- Multiple model types, solvers and platforms

Algebraic Modeling Systems

Multiple Model Types

- LP Linear Programming
- MIP Mixed Integer Programming
- NLP Nonlinear Programming
- QCP/MIQCP Quadratic Programming
- Conic Programming
- MCP Mixed Complementarity Programming
- MINLP Mixed Integer Nonlinear Programming
- MPEC NLP with Complementarity Constraints
- MPSGE General Equilibrium Models
- Stochastic Optimization
- Global Optimization

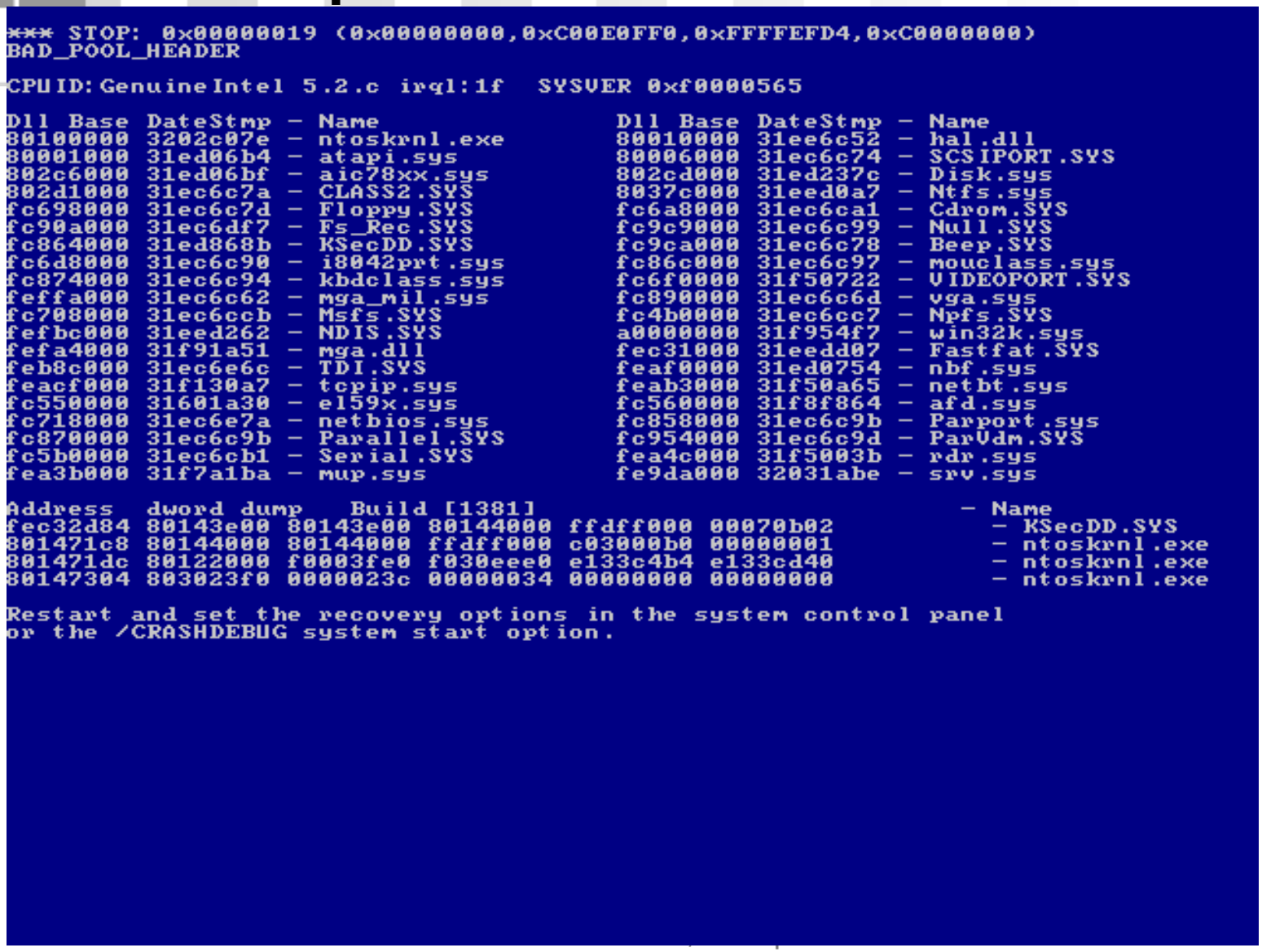
Challenges

- Expense: Rigorous SQA is initially expensive
  - MP industry is small
- Limited control about certain parts of the system
  - solvers are black box modules
- Distributed development of the system and various components

Challenges: Chance of Failure

QA issues specific to Modeling Systems:

Implementation Defect



Blue Screen of Death in Microsoft Windows NT

Not acceptable

Solver Failure

Solver does not find a solution

Acceptable

```
               S O L V E      S U M M A R Y

     MODEL   one                     OBJECTIVE  output
     TYPE    NLP                     DIRECTION  MAXIMIZE

**** SOLVER STATUS     4 TERMINATED BY SOLVER
**** MODEL STATUS      6 INTERMEDIATE INFEASIBLE
**** OBJECTIVE VALUE              948403.4844

 RESOURCE USAGE, LIMIT          1.906      1000.000
 ITERATION COUNT, LIMIT         0         10000
 EVALUATION ERRORS              0             0
```

Challenges: Chance of Failure

QA issues specific to Modeling Systems:

- Additionally, must protect the user in case of solver failure
- Complex metric for return codes is necessary
- Complicates QA activities since this adds an additional level of complexity

SQA at GAMS

1. **Software configuration management**
2. **Quality control and tests of the product**
3. **Client model testing**
4. *Performance comparison tools:*
   - *Paver*
   - *Bench*
5. *Solution verification tool: Examiner*
6. *Model converter and “encryption“ tool: Convert*

SQA at GAMS (SCM)

Software configuration management

- Audit strings

```
C O N O P T 3 Jan 19, 2004 WIN.CO.CO 21.3 015.050.041.VIS Library 313C
C O N O P T 3 BETA 28Jul04 WIN.CO.CO 21.4 015.051.041.VIS Library 314D
```

- Build automation tools: Automatic build of the whole system every week - *build early and build often*
- Simple source management system (automatic version and audit string checks)

Version Check of Source

More than 46 components!

GAMS Distribution Version Information

DATE: Mon Aug 30 18:49:41 EDT 2004

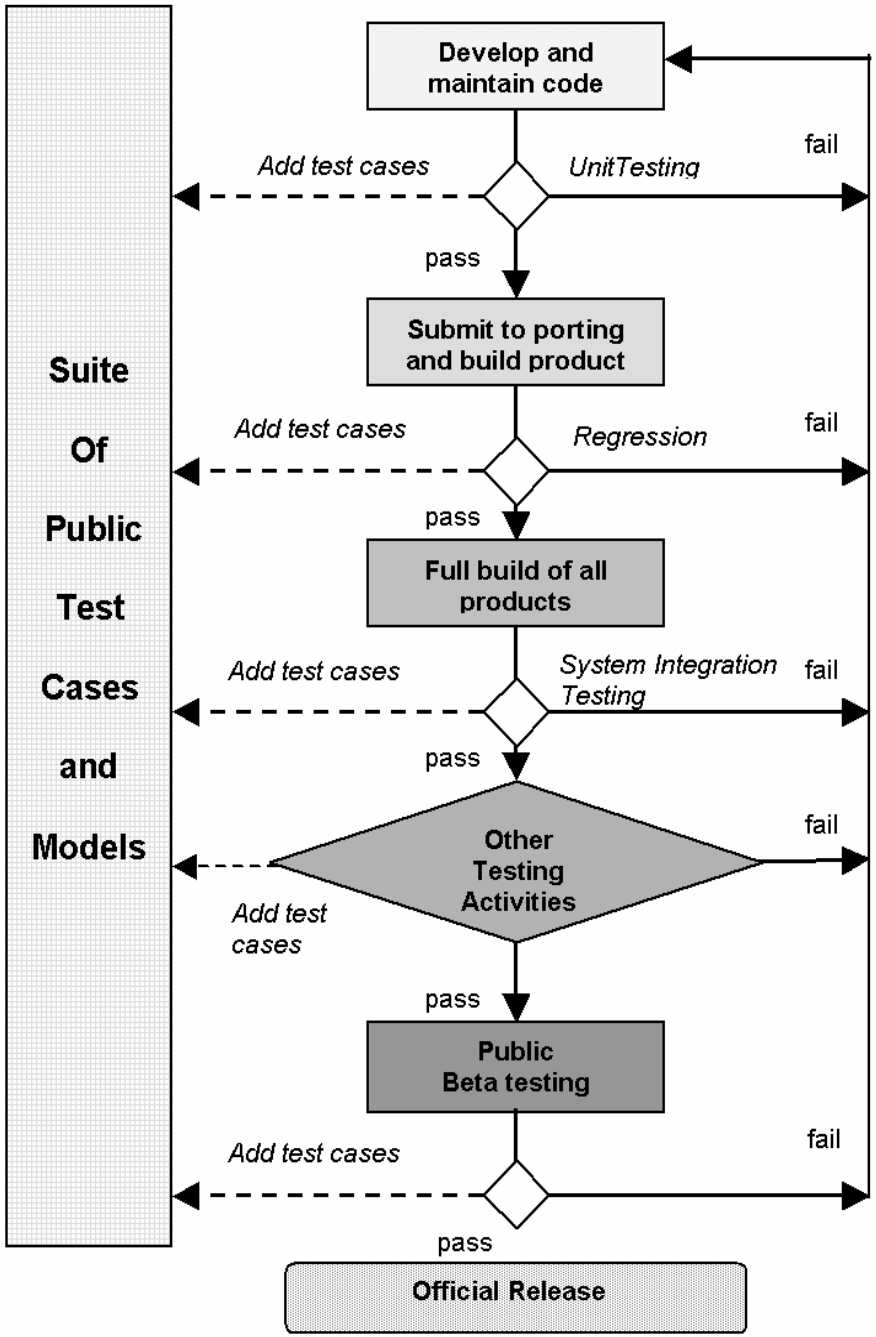
	207	210	211	212	213	BETA 214	ALFA 215
AMPLLINK:	002	002	002	003	004	005	005
AMPLM:	---	001	001	002	002	002	002
AUDITTEST:	002	002	002	002	002	002	002
BARON:	003	004	005	005	006	007	007
BARONLIB:	005	006	006	007	008	009	009
BDMLP:	057	057	057	058	058	059	059
BENCH:	---	001	001	003	004	005	005
CGAMSLIB:	003	004	004	005	006	007	007
CIOLIB:	024	025	025	026	027	028	028
CLICELIB:	006	007	007	008	008	009	010
COIN:	---	---	---	---	---	001	001
CONLIB1:	043f	043f	043f	043f	043f	043f	043f
CONLIB2:	071j	071k	071k	071k	071k	071k	071k
CONLIB3:	01f	11c	11c	12a	13c	14d	14d
CONLINK:	009a	010x	010x	010x	010y	010z	010z
CONLINK3:	011f	014a	014a	015a	015c	015d	015d
CONVERT:	006	007	008	009	010	011	012
CPLEX:	022	023	023	023	025	026	026
CPLEXLIB:	libs7.5	libs8.1	libs8.1	libs8.1	libs9.0	libs9.0	libs9.0
CPLIB:	019	020	020	021	022	023	023
DEA:	003	004	004	004	004	005	005
DECIS:	010	012	012	013	013	015	015
DECISC:	---	001	001	001	002	003	003
DECISM:	---	000	000	001	002	002	002
DICOPT:	031	035	035	037	037	039	039
DICTDLL:	---	000	000	001	002	002	002
DTOOLS:	003	005	005	006	007	008	008
EXAMINER:	009	010	010	011	012	013	013
F2CLIB:	---	001	001	002	002	002	002
F90IOLIB:	046	048	048	049	050	051	051
F90LICELIB:	006	007	007	008	008	009	010

SQA at GAMS (Testing)

Quality control and tests of the product

- Goal: Continuous quality improvement using automated and reproducible tests
- Test libraries (available online):
  - GAMS Model Library (solver tests)
  - GAMS Quality Test Models Library (modeling system test)
- Continuous addition of new test models

General testing process



Continuous addition of new test models throughout life cycle

SQA at GAMS

Quality Test Models Library

- Include tests to verify proper behavior of the system
- More than 180 quality test models, each containing numerous pass/fail tests:

```
...
abort$card(delta) 'time routines have an error';
...
```

- Automatic generated test summaries with different level of information

SQA at GAMS

Summary of two quality runs

```
*** Status: Normal completion
--- quality.gms(284) 4 Mb
--- quality.gms(287) 4 Mb 1 Error
There were errors: 4 out of 267 tests failed.
See the file failures.gms to reproduce the failed runs
--- Putfile this D:\support\testlib\onetest.gms
--- quality.gms(287) 4 Mb 1 Error
*** Status: Execution error(s)

==============================================

*** Status: Normal completion
--- quality.gms(284) 4 Mb
--- quality.gms(295) 4 Mb
Congratulations!  All 267 tests passed.
See the file alltests.gms to reproduce all the runs
--- Putfile this D:\support\testlib\onetest.gms
*** Status: Normal completion
```

Testing New Solver Links

Solver developer has connected his solver to GAMS (e.g. a COIN solver)

- Automated tests to check basic functionality of the solver and the link to GAMS:

```
$title Simple level and sign test (LP02,SEQ=67)
* In this test series we status if a solver gets the levels
* and marginals right.
..
abort$( abs(cost.m-cost_m) > tol) 'bad cost.m';
```

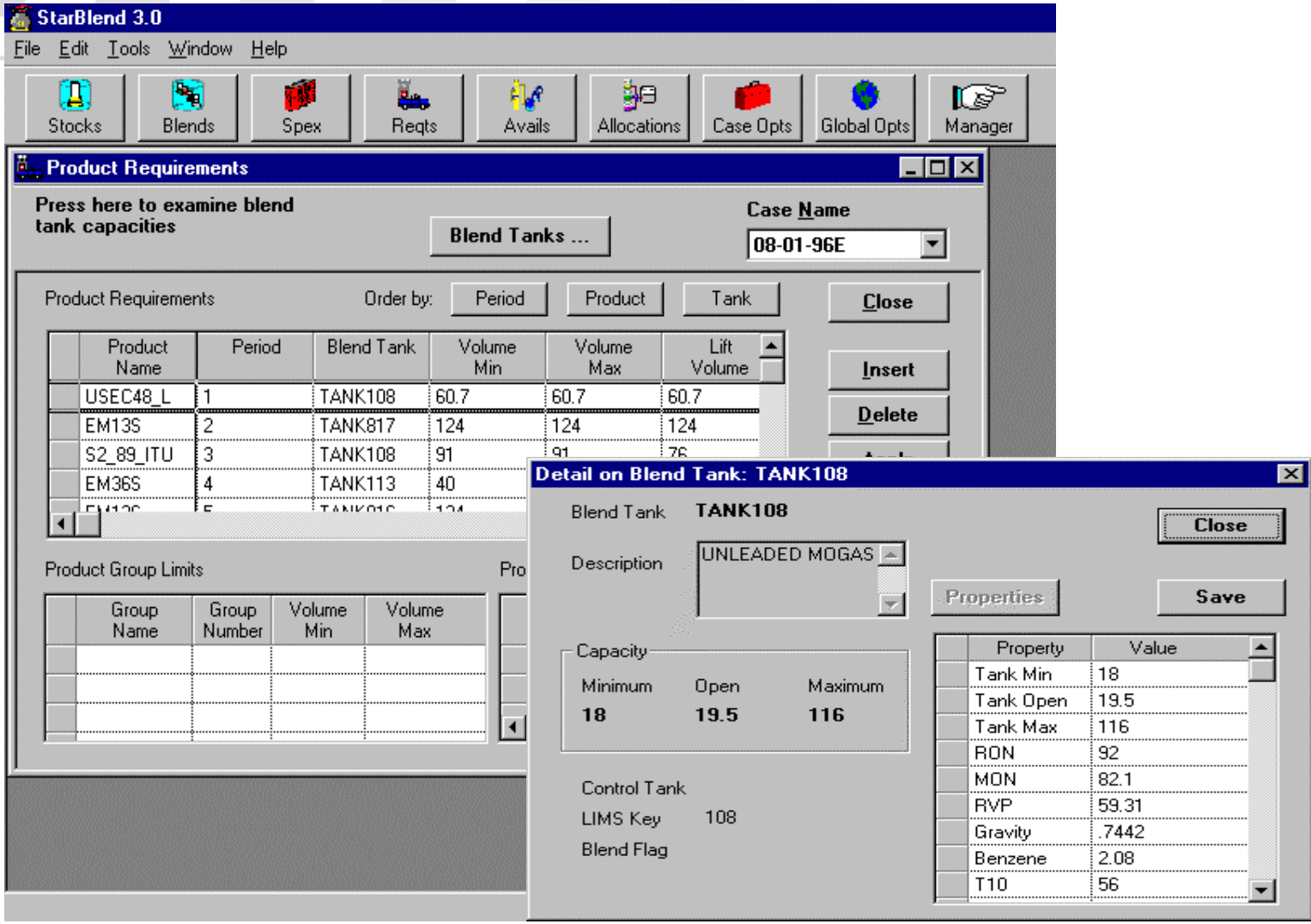
- Gives developer and users assurance about the basic functionality of the link and the solver

Client Model Testing

- Client with complex application (gasoline blending system)
- New GAMS version available:
  - Relevant new features?
  - Performance gains?
  - No surprises?
    - Bugs
    - Different results (MIP models)

Client Model Testing

Client Application: Gasoline Blending



Client Model Testing

- Want guarantee that their application will work with the new version
- Only limited ressources to do major testing themselves
- *Confidentiality issues: Running tests without having access to internal model structures and model data (in a human readable format)*

Client Model Testing

- Gives clients assurance that their application will also work with new GAMS releases
- Includes:
  - Ability to solve (= no bugs)
  - Returns the same solution back
  - Similar or better performance
- Requires changes to the model of the clients to allow automated pass/failure tests
- Improves communication between development team and clients (specific wishes)

Lessons

- *Automate* the QA process and the certification
- Build early and *build often*
- *Incorporate* QA tools into the software and share the QA process
- Make the QA process *transparent and reproducible*
- *Involve solver developers and clients* into the QA process

Summary

- SQA becomes more and more important for MP industry
- Our focus is on automated configuration management and automated testing (reproducible full life cycle testing)
- Most of the test components are available as transparent GAMS models
- Involvement of the clients in the QA process is essential
- Benefits both for solver developers, for clients and for GAMS

References

- Bussieck, M.R., Drud, A.S., Meeraus, A. and Pruessner, A. (2002), Quality Assurance and Global Optimization, C. Bliek, C. Jermann, A. Neumaier, eds. Global Optimization and Constraint Satisfaction, First International Workshop on Global Constraint Optimization and Constraint Satisfaction, COCOS 2002, LNCS2861. Springer Verlag, Heidelberg Berlin, (223-238).
- Bussieck, M.R., Dirkse, S.D., Meeraus, A. and Pruessner, A. (2004), Software Quality Assurance for Mathematical Modeling Systems, forthcoming
- International Organization for Standardization (2004), online at:http://www.iso.org.
- PAVER Server (2004), online at http://www.gamsworld.org/performance/paver
- Pruessner, A. (2002), Automated Performance Testing and Analysis, online at http://www.gams.com/presentations/present_automation.pdf
- Schulmeyer, G.G. and McManus, J.I. (1998), Software Quality Handbook, Prentice Hall