Design Principles that Make the Difference

Toni Lastusilta: tlastusilta@gams.com
Franz Nelissen: fnelissen@gams.com
Company Background

Roots: World Bank, 1976

GAMS Development Corporation (Washington)

Tool Provider: General Algebraic Modeling System

Went commercial in 1987

GAMS Software GmbH (Cologne, Braunschweig) 1996
Agenda

What is GAMS?

• Algebraic Modeling Languages – A Success Story
• GAMS – Highlights and Design Principles
• Striving for Innovation and Compatibility
  • New Modeling and Solution Concepts
  • Software Quality Assurance

A Simple Example

Some Applications
The Vision

RESULT:
- Limited drain of resources
- Same representation of models for humans and machines
- Model representation is also model documentation
Algebraic Modeling Languages (AML)

1. High-level computer programming languages
   - Formulation of mathematical optimization problems
   - Notation similar to algebraic notation

2. Do not solve problems directly, but offer links to state-of-the-art algorithms ("solver-links")

Source: http://en.wikipedia.org/wiki/Algebraic_modeling_language
2012 INFORMS Impact Prize

Originators of Algebraic Modeling Languages

36 Years later
Impact of Algebraic Modeling Languages

1. Simplified model development, changes, and transfer
2. Added value to existing applications
3. Increased productivity, quality, reliability, and maintainability
4. Made a scarce resource (good modelers) more productive
5. Important vehicle to make mathematical optimization available to a broader audience, e.g., domain-specific experts

Why not use current big data systems?
What does he have to think about?

1. Problem
2. Mathematics
3. Programming
4. Performance
5. Scalability
6. Connectivity
7. Deployment
8. Maintenance (Life Cycle)
9. ...

Why should he use GAMS?
Broad User Community and Network

GAMS used in more than 120 countries

25+ Years GAMS Development
GENERAL ALGEBRAIC MODELING SYSTEM

Broad User Community and Network

More than 10,000 licenses

50% academic users, 50% commercial users

6,000+ monthly downloads of the free system

25+ Years
GAMS Development
## Broad Range of Application Areas

<table>
<thead>
<tr>
<th>Agriculture Economics</th>
<th>Applied General Equilibrium</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chemical Engineering</td>
<td>Economic Development</td>
</tr>
<tr>
<td>Econometrics</td>
<td>Energy</td>
</tr>
<tr>
<td>Environmental Economics</td>
<td>Engineering</td>
</tr>
<tr>
<td>Finance</td>
<td>Forestry</td>
</tr>
<tr>
<td>International Trade</td>
<td>Logistics</td>
</tr>
<tr>
<td>Macro Economics</td>
<td>Military</td>
</tr>
<tr>
<td>Management Science/OR</td>
<td>Mathematics</td>
</tr>
<tr>
<td>Micro Economics</td>
<td>Physics</td>
</tr>
</tbody>
</table>

25+ Years  
GAMS Development
General Algebraic Modeling System

Strong Development Environment

GAMS IDE

- Project management
- Editor / Syntax coloring / Spell checks
- Listing file / Tree view / Syntax-error navigation
- Model Debugging / Profiling
- Solver selection / Option selection
- Data viewer (GDX)
  - Export
  - Charting
- GAMS Processes Control
- Model Libraries
Free Model Libraries

More than 1250 models!
Design Principles

1. Simple modeling language with a balanced mix of declarative and procedural elements

2. Open architecture and interfaces to other systems, independent layers
Simple Declarative Language

1. Few basic language elements: sets, parameters, variables, equations, models
2. Language similar to mathematical notation
3. Easy to learn
4. Model is executable description of the problem
5. Lot’s of code optimization under the hood
Mix of Declarative and **Procedural Elements**

Procedural elements like loops, for, if, macros and functions

Allow to build complex problem algorithms within GAMS

Interaction with other systems:
- Job control
- Data exchange
Independence of Model and Operating System

Platforms supported by GAMS:

Models can be moved between platforms with ease!
Independence of Model and Solver

One environment for a wide range of model types and solvers

- All major commercial LP/MIP solver
- Open Source Solver (COIN)
- Also solver for NLP, MINLP, global, and stochastic optimization

Switching between solvers with one line of code!
Independence of **Model and Data**

- **Declarative Modeling**
- **ASCII**: Initial model development
- **GDX**: Data layer ("contract") between GAMS and applications
  - Platform independent
  - No license required
  - Direct GDX interfaces and general API
  - ...

**Model**

- Platform
- Solver
- Data
- Interface
Independence of Model and User Interface

API’s

- **Low Level**
- **Object Oriented**: .Net, Java, Python
- No modeling capability: Model is written in GAMS
- Wrapper class that encapsulates a GAMS model
Simple Encapsulation of a GAMS Model

Very simple interface:

- Properties to communicate input data and results
- Properties to change options like the solver to use
- Run() method to run the model
Smart Links to other Applications

- User keeps working in his productive tool environment
- Application accesses all optimization capabilities of GAMS through API
- Visualization and analysis of model data and results in the application
Striving for **Innovation and Compatibility**

**Models must benefit from:**
- Advancing hardware / New Platforms
- Enhanced / new solver and solution technology
- Improved / upcoming interfaces to other systems
- New Modeling Concepts

**Protect investments of Users**
- Life time of a model: 15+ years
- New maintainer, platform, solver, user interface
- Backward Compatibility
- Software Quality Assurance
New Modeling and Solution Concepts

Examples:
- Disjunctive Programs
- Bilevel Programs
- Extended Nonlinear Programs
- Stochastic Programming
- ...

Issues:
- Breakouts of traditional Mathematical Programming classes
- No conventional syntax
- Limited support with common model representation
- Incomplete/experimental solution approaches
- Lack of reliable/any software

GAMS is conservative when it comes to syntax extensions
The “GAMS” – Approach

Extended Mathematical Programming

- Experimental framework for automated mathematical programming reformulations
- Keep the language simple: Do not overload existing GAMS notation
- Use existing language features to specify additional model features, structure, and semantics
- Express extended model information in symbolic (source) form and apply existing modeling/solution technology
- Package new tools with the production system
**Stochastic Programming in GAMS**

**EMP/SP**
- Simple interface to add uncertainty to existing deterministic models
- (EMP) Keywords to describe uncertainty include: discrete and parametric random variables, stages, chance constraints, Value at Risk, ...
- Available solution methods:
  - Automatic generation of **Deterministic Equivalent** (can be solved with any solver)
  - Specialized commercial algorithms (DECIS, LINDO)
Software Quality Assurance

Perspectives:
• What is the impact of new features?
• What is the impact of updated or new solvers?
• Is the new distribution backward compatible?
• ...
Quality Test Models Library

- Tests to verify proper behavior of the system
- More than 600 quality test models, each containing numerous pass/fail tests
- Automatically executed every night for all solver combinations: → 12,000 runs / platform (all tests)
- Automatically generated test summaries with different level of information
- Assurance about the basic functionality of the software!

Latest GAMS System Builds and Test Results

<table>
<thead>
<tr>
<th>nightly</th>
<th>System</th>
<th>Libraries</th>
<th>Build</th>
<th>Rev</th>
<th>Status and Time (UTC)</th>
<th>Initial Tests</th>
<th>Full Tests</th>
<th>Comments?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Friday</td>
<td>lnx</td>
<td>Download</td>
<td>24.3.0</td>
<td>45152</td>
<td>Test done 22Mar2014 12:42:54</td>
<td>805 runs 0 failures (q=0,s=0)</td>
<td>Report 11780 runs 0 failures (q=0,s=0)</td>
<td>Report</td>
</tr>
<tr>
<td>Saturday</td>
<td>leg</td>
<td>Download</td>
<td>24.3.0</td>
<td>45152</td>
<td>Test done 22Mar2014 22:03:49</td>
<td>804 runs 0 failures (q=0,s=0)</td>
<td>Report 2198 runs 0 failures (q=0,s=0)</td>
<td>Report</td>
</tr>
<tr>
<td>Friday</td>
<td>yas</td>
<td>Download</td>
<td>24.3.0</td>
<td>45152</td>
<td>Test started 22Mar2014 01:12:52</td>
<td>947 runs 0 failures (q=0,s=0)</td>
<td>Report results pending</td>
<td>Report</td>
</tr>
<tr>
<td>Friday</td>
<td>wei</td>
<td>Download</td>
<td>24.3.0</td>
<td>45152</td>
<td>Test started 22Mar2014 03:47:14</td>
<td>944 runs 1 failures (q=1,s=0)</td>
<td>Report results pending</td>
<td>Report</td>
</tr>
</tbody>
</table>
Why GAMS?

- Experience of 25+ years
- Broad user community from different areas
- Lots of model templates
- Strong development interface

- Consistent implementation of design principles
  - Simple, but powerful modeling language
  - Independent layers
  - Open architecture: Designed to interact with other applications

- Open for new developments
- Protecting investments of users
Agenda

What is GAMS?

A Simple Example

Some Applications
A Simple Transportation Problem

What does this example show?

- It gives a first glimpse of how a problem can be formulated in GAMS
- It shows how easy it is to change model type and, consequently, solver technology
Model types in this example

- **LP**
  - Determine minimum transportation cost.
  - Result: city to city shipment volumes.

- **MIP**
  - Allows discrete decisions,
    - e.g. if we ship, then we ship at least 100 cases.

- **MINLP**
  - Allows non-linearity,
    - e.g. a smooth decrease in unit cost when shipping volumes grows.
A Simple Transportation Problem

Canning Plants (supply) \[\rightarrow\] shipments (Number of cases) \[\rightarrow\] Markets (demand)

Seattle (350)
San Diego (600)
Topeka (285)
Chicago (300)
New York (325)

Freight: $90 case / thousand miles
Minimize transportation cost
subject to demand satisfaction at markets
Supply constraints
Indices: $i$ (Canning plants)  
             $j$ (Markets)  
Decision variables: $x_{ij}$ (Number of cases to ship)  
Parameter: $c_{ij}$ (Transport cost per case)  

\[
\min \sum_{i} \sum_{j} c_{ij} \cdot x_{ij}
\]

(subject to  
\[
\sum_{j} x_{ij} \leq a_{i}, \quad \forall \ i
\]
\[
\sum_{i} x_{ij} \geq b_{j}, \quad \forall \ j
\]
\[
x_{ij} \geq 0, \quad \forall i, j
\]
\[
i, j \in \mathbb{N}
\]

(Minimize total transportation cost)  
(Shipments from each plant $\leq$ supply capacity)  
(Shipments to each market $\geq$ demand)
GAMS Algebra (LP Model)

Variables
x(i,j) shipment quantities in cases
z total transportation costs in thousands of dollars;

Positive Variable x:

Equations
cost define objective function
supply(i) observe supply limit at plant i
demand(j) satisfy demand at market j:

cost .. z =e= sum((i,j), c(i,j)*x(i,j)) ;

supply(i) .. sum(j, x(i,j)) =l= a(i) ;
demand(j) .. sum(i, x(i,j)) =g= b(j) ;

Model modellP /cost, supply, demand/ ;
Solve modellP using lp minimizing z ;
Solution to LP model

Canning Plants (supply)  

Markets (demand)  

Seattle (350)  
San Diego (600)  
Topeka (285)  
Chicago (300)  
New York (325)  

Shipments (Number of cases)

Freight: $90 case / thousand miles  
Total cost: $154,935
Minimum shipment of 100 cases

- Shipment volume: $x$ (continuous variable)
- Discrete decision: $ship$ (binary variable, 0 or 1)

Add constraints:

$$x_{i,j} \geq 100 \cdot ship_{i,j} \quad \forall \ i, j \quad \text{(if } ship=1, \text{ then ship at least 100)}$$

$$x_{i,j} \leq 325 \cdot ship_{i,j} \quad \forall \ i, j \quad \text{(if } ship=0, \text{ then do not ship at all)}$$

$ship_{i,j} \in \{0,1\}$
Scalar mins minimum shipment volume (number of cases) /100/
   bigm big-M relaxation;

bigm = min(sm_max(i,a(i)), smax(j,b(j)));

Binary variables ship(i,j);

Equations
   minship(i,j) minimum shipment
   maxship(i,j) maximum shipment;

minship(i,j).. x(i,j) =g= mins*ship(i,j);
maxship(i,j).. x(i,j) =l= bigm*ship(i,j);

Model modelMIP /modelLP, minship, maxship/;

Solve modelMIP using mip minimizing z;
Solution to MIP model

Canning Plants (supply) shipments Markets (demand)

Seattle (350) 285 San Diego (600)

Topeka (285) 100 Chicago (300)

New York (325)

Freight: $90 case / thousand miles

Total cost: $155,835
Cost Savings (non-linear)

The cost per case decreases with an increasing shipment volume.

Replace:
\[
\min \sum_i \sum_j c_{ij} \cdot x_{ij} \quad (\text{Minimize total transportation cost})
\]

With
\[
\min \sum_i \sum_j c_{ij} \cdot x_{ij}^{\beta} \quad (\text{Minimize total transportation cost})
\]
GENERAL ALGEBRAIC MODELING SYSTEM

GAMS Algebra (MINLP Model)

Scalar beta    beta-factor / 0.976 /

Equations
  newcost    define new objective function;

newcost ..
  z =e=  sum((i,j), ship(i,j)*c(i,j)*(mins-mins**beta) + c(i,j)*x(i,j)**beta);

Model modelMINLP /modelMIP - cost + newcost/ ;

Solve modelMINLP using minlp minimizing z ;
Solution to MINLP model

Canning Plants (supply) → shipments → Markets (demand)

Seattle (350) → San Diego (600) → Topeka (285) → Chicago (300) → New York (325)

Freight: ~$90 case / thousand miles
Total cost: $142,752
Agenda Breakdown

Some Applications

Scenario Solving with GAMS
  - XYZ – Energy Company
  - ACSOM

Outlook: GAMS under ODM Enterprise
**Problem:** “small” number of independent scenarios...

- Through procedural language elements (e.g. loop statement)
- Works with any model type and solver
- Communication with solver through files or memory (in-core)
### Scenario Solver

**...10,000’s of independent scenarios...**
- GAMS Scenario Solver allows to generate model once and updates the algebraic model keeping the model “hot” inside the solver.
- Platform independent, works with all solvers
- Performance close to native solver API
- Example: Stochastic model, 66,320 linear problems

<table>
<thead>
<tr>
<th>Setting</th>
<th>Solve time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loop: Solvelink=%Solvelink.Chainscript (default)</td>
<td>7,204</td>
</tr>
<tr>
<td>Loop: Solvelink=%Solvelink.LoadLibrary%</td>
<td>2,481</td>
</tr>
<tr>
<td>GAMS Scenario Solver</td>
<td>392</td>
</tr>
<tr>
<td>Cplex Concert Technology</td>
<td>210</td>
</tr>
</tbody>
</table>

**Factor**

- 18.3
- 1.86
GAMS GRID Facility

Running scenarios in a **distributed** environment...

- Scalable: support large grids, but also works on desktops
- Platform independent, works with all solvers/model types
- Only minor changes to model required

1. Submission of jobs
2. “Grid Middleware”
   - Distribution of jobs
   - Job execution
3. Collection of solutions
4. Processing of results
Massive Parallel in the Cloud

XYZ- Energy Company

- Large scale MIP model
- Failures with Stochastic Optimization and Robust Optimization
- Scenario Analysis works!
- Task: Solve 1000 scenarios (MIPs, 1-2 hours) every week overnight
- Security issues: Obfuscation of (binary) GAMS files
- Post processing of results outside modeling system
Massive Parallel in the Cloud

- Deployment environment: „Cloud“ (Amazon EC2)
- A few hundred parallel instances (workers)
- Automated setup, including
  - Start instances
  - Prepare / Submit / Run GAMS jobs
  - Collect results
  - Stop instances
- Python
  - Boto: Python interface to Amazon webservice
  - Setup and control of instances (Workers)
  - GAMS OO API (Python): Control of GAMS jobs
Massive Parallel in the Cloud

Setup

- Preparation of obfuscated binary model file
- Preparation of data for every worker
- Mapping of results into original namespace

• Control of instances
• Submission of model and scenario data to workers
• Collection and merge of results

Configured AWS Instances
• Python
• GAMS
ACSOM

- Advanced Collaborative System Optimization Modeler
- Configuration of (expensive) military vehicles
- Project partners:
  - General Dynamics Land Systems (GDLS)
  - Industrial & Systems Engineering, Wayne State University
  - GAMS Development Corp
  - Amsterdam Optimization Group
The Whole-system Design Problem

21 SUBSYSTEMS

<table>
<thead>
<tr>
<th>Options per subsystem</th>
<th>Theoretically Possible Subsystem Combinations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2,092,152</td>
</tr>
<tr>
<td>3</td>
<td>10,460,353,203</td>
</tr>
</tbody>
</table>

AFES
Armor
Chassis Structure
Crew Station
Defensive Armament
Dismountable
ECS
Fuel
Hit Avoidance
Lighting
Mission Equipment (Shooter)
Mission Station
Mission Structure
NBC
Platform Electronics
Power Distribution & Mgmt
Propulsion
Signature Management
Suspension
Turret Structure
Water Management
Project **Scope**

Reimplementation of ACSOM 1.2 algorithm

Significant performance improvements

Use multiple cores
- For current setup (laptops)
- For high-performance machines
- Be prepared for advancing technologies (more parallelism)
# Scenario Solver and Parallel Combined

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Number of MIP models</th>
<th>Solve time</th>
<th>Rest of algorithm</th>
<th>Total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional GAMS loop (call solver as DLL)</td>
<td>100,000</td>
<td>1068 sec</td>
<td>169 sec</td>
<td>1237 sec</td>
</tr>
<tr>
<td>Scenario Solver</td>
<td>100,000</td>
<td>293 sec</td>
<td>166 sec</td>
<td>459 sec</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Number of MIP models</th>
<th>Worker Threads</th>
<th>Parallel sub-problem time</th>
<th>Rest of algorithm (serial)</th>
<th>Total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallel + Scenario Solver</td>
<td>100,000</td>
<td>4</td>
<td>116 sec</td>
<td>67 sec</td>
<td>183 sec</td>
</tr>
</tbody>
</table>

http://yetanothermathprogrammingconsultant.blogspot.de/2012/04/parallel-gams-jobs-2.html
ACSOM Project Summary

**New System**
- GAMS (complete algorithm; streamlined design)
- Multiple parallel instances of scenario solver
- Optimized communication with database

**Performance improvement**
(same hardware)
- From **hours** and even **days** for some instances
- To **minutes**, up to an **hour**
Scenario Solving with GAMS

Various options:

- Loop (declarative language elements)
- Scenario Solver
- Grid Facility
- Parallel Execution

Simple extensions of Model:

- Independence of Model and Platform
- Independence of Model and Solver
- Independence of Model and Interface
Agenda Breakdown

Some Applications

Scenario Solving with GAMS
- XYZ – Energy Company
- ACSOM

Outlook: GAMS under ODM Enterprise
GEM Energy Models

- Fully integrated U.S. coal/electricity model
- Runs at unit level: coal choices, cleanup decisions, dispatch
- User-friendly customized interfaces in popular applications (Excel, Access, etc. using GAMS data exchange utilities)
- Uses a dynamic transportation costing network (GTM™)
GEM Excel Interface
ODM System Architecture

ODM Enterprise Development
- Application UI Configuration
- Data Modeling
- Data Integration
- Reporting
- Application UI Customization

IT

ODM Enterprise Optimization Server & Engine

ODM Enterprise Data Server

ODM Enterprise Studio Planner & Client
- Custom GUI
- Batch process

Business Use

Development

Optimization Modeling, Tuning, Debugging

Embeds all CPLEX Optimization Studio
Replacing **OPL with GAMS**

Implementation:
- ODM Custom Task (GAMS Solve Task)
- GAMS OO Java – API (Data Exchange, Job Control)
GAMS Gas Transmission Model in ODM
GENERAL ALGEBRAIC MODELING SYSTEM

Summary

GAMS Design Principles

- Simple language
- Open architecture with independent layers

Striving for Innovation and Compatibility

- Models benefit from new developments
- Protect investments of users

Don’t lock users into a certain environment!
## Thank You

<table>
<thead>
<tr>
<th>USA</th>
<th>Europe</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAMS Development Corp.</td>
<td>GAMS Software GmbH</td>
</tr>
<tr>
<td>1217 Potomac Street, NW Washington, DC 20007</td>
<td>P.O. Box 40 59</td>
</tr>
<tr>
<td>USA</td>
<td>50216 Frechen, Germany</td>
</tr>
<tr>
<td>Phone: +1 202 342 0180</td>
<td>Phone: +49 221 949 9170</td>
</tr>
<tr>
<td>Fax: +1 202 342 0181</td>
<td>Fax: +49 221 949 9171</td>
</tr>
<tr>
<td><a href="mailto:sales@gams.com">sales@gams.com</a></td>
<td><a href="mailto:info@gams.de">info@gams.de</a></td>
</tr>
</tbody>
</table>

---

GAMS Development Corp. | GAMS Software GmbH | www.gams.com