Is there a future for modeling systems?

Jan J. Bisschop
Paragon Decision Technology
Haarlem, the Netherlands
Background

- a customer raised the question
- ILOG to invest less in OPL?
- the IT department wants C++, and
- no dependence on small suppliers
Is there a future in modeling systems?
What is the future of modeling systems?

A personal vision
In the 70’s

- lengthy, low-level, error-prone programs
- programs written by computer experts
- programs difficult to change or transfer
- slow turn-around of submitted jobs
Thanks to

- visionaries at the World Bank
- the research climate
- the requests for decision support

GAMS was born
What did GAMS give us?

- high-level computerized model description
- technology also for domain experts
- ease of making changes
- ease of model transfer

Tremendous increased productivity
Productivity requirement remains, but ...

Now, 30 years later, the world of optimization has changed and critics of modeling tools make serious claims.
Claim # 1: current modeling tools play insignificant role

The major efforts are:
- data feed and cleaning
- graphical reporting
- client-server interaction
- web-enabling of applications
Claim # 2: current modeling tools are not professional

Instead, one should use

• C++ component libraries
• full C++ development support
• multi-developer source safe tools
• latest graphical design tools
Claim # 3: current modeling tools are often too slow

With C++ you can

• customize data structures
• profit from compiler improvements
• change to parallel implementations
Claim # 4: current modeling tools do not support re-use

Instead of “cut-copy-paste” operations you can use C++

- object classes
- derived classes
- component technology
Claim # 5: current modeling tools have poor algorithm linkage

Instead of black-box solvers, you can use
- various solvers with callbacks
- connect them at will
- introduce new interface functions
Claim # 6:
current modeling tools are likely to disappear

Instead of relying on fading technology you can use

• C++ generators with source code
• fully-trained professionals
• up-to-date technology
Summary of claims:

Current modeling tools:
• play insignificant role
• are not professional
• are often too slow
• do not support re-use
• have poor algorithm linkage
• are likely to disappear
Possible courses of action

- do not change fundamentally
- change to adjust to new niches
- change fundamentally

I like to address the latter
A complete development environment

Provide a technology with
• professional & up-to-date development tools
• a rich set of interfaces (C++, Java, .Net)
• extensive data connectivity
• etc.
Code generation and parallelization

For static data structures
- search for repetitive computations
- generate assembler code
- generate models in parallel
- evaluate functions/derivatives in parallel
Code generation (repetition)

A(i,j,t)
Object-oriented modeling

Object-oriented modeling paradigm:

• coarse-grain model objects
• use of OO design methods
• increased locality for modeler
• re-use of existing models/components
• dimension/scope increase through typed sets
Object-oriented modeling

OBJECT CLASS CrudeDistiller

crude distiller model

OBJECT CLASS OtherUnit

other unit model
Object-oriented modeling

OBJECT CLASS CrudeDistiller

OBJECT CLASS OtherUnit

These sets:

• are ordinary index sets
• provide access to identifiers in classes, for example cd.ThroughPut
Object-oriented modeling

OBJECT CLASS CrudeDistiller

**crude distiller model**

OBJECT CLASS OtherUnit

**other unit model**

SET CrudeDistillers OF TYPE CrudeDistiller

SET OtherUnits OF TYPE OtherUnit

OBJECT CLASS Plant

**crude distiller model**  **crude distiller model**  **...**

**other unit model**  **other unit model**  **other unit model**
Open solver environment for customized algorithms

Extend modeling systems to support
• multiple interacting math programs
• functions for model manipulation
• simultaneous access to multiple solvers
• extensive library of control functions
• support for solver callbacks
Open solver environment

Solver 1

Interface
- Exported Functions
- Callbacks

Generated Model(s)

Algorithm(s)

Modeling System

procedure Main;
   MP1.cb := P;
   repeat
      solve MP1;
      AddCuts(MP1);
   endrepeat;

procedure P:
   InitializeData;
   solve MP2;

Solver 2

Interface
- Exported Functions
- Callbacks

Generated Model(s)

Algorithm(s)
Perhaps merge companies?

Horizontal Package Solutions

Vertical Package Solutions

Generic modeling Tools

OR modeling Tools

Integrated Programming Environments & Middleware

Scientific / OR Software

Business Software

Custom Solutions / System Integration

Strategy

SAP
PeopleSoft
JD Edwards
Siebel
Clarify

PIMS
QuintiQ
ESCA
I2

Cognos
Business Objects
MicroStrategy
Sequent

ORTEC
AbOvo

Aspen,
SAS
ILOG

ORTEC
CQM

IBM, EDS, CAP, Atos/Origin, CMG/Logica, PinkRoccade, Ordina, ....

KPMG, PWC, ....

MCKinsey, BCG, Bain, Monitor, ....

Mathematica
MAPL
Mathcad
MatLab

AIMMS
AMPL
GAMS
LINDO
MPL

MS Excel

Visual C++ / C#
Visual Basic
Java
Delphi

Oracle
IBM
MicroSoft
Sun
HP

IBM, EDS, CAP, Atos/Origin, CMG/Logica, PinkRoccade, Ordina, ....

KPMG, PWC, ....

MCKinsey, BCG, Bain, Monitor, ....

IBM, EDS, CAP, Atos/Origin, CMG/Logica, PinkRoccade, Ordina, ....

KPMG, PWC, ....

MCKinsey, BCG, Bain, Monitor, ....
Is there a future for modeling systems?

For me, the answer is

YES,

but …