# Modelling and Solving Environments for MP: A Status Review and New Directions

*B. Dominguez-Ballesteros, G. Mitra, C. Lucas, N-S. Koutsoukis*

Dept. of Mathematical Sciences

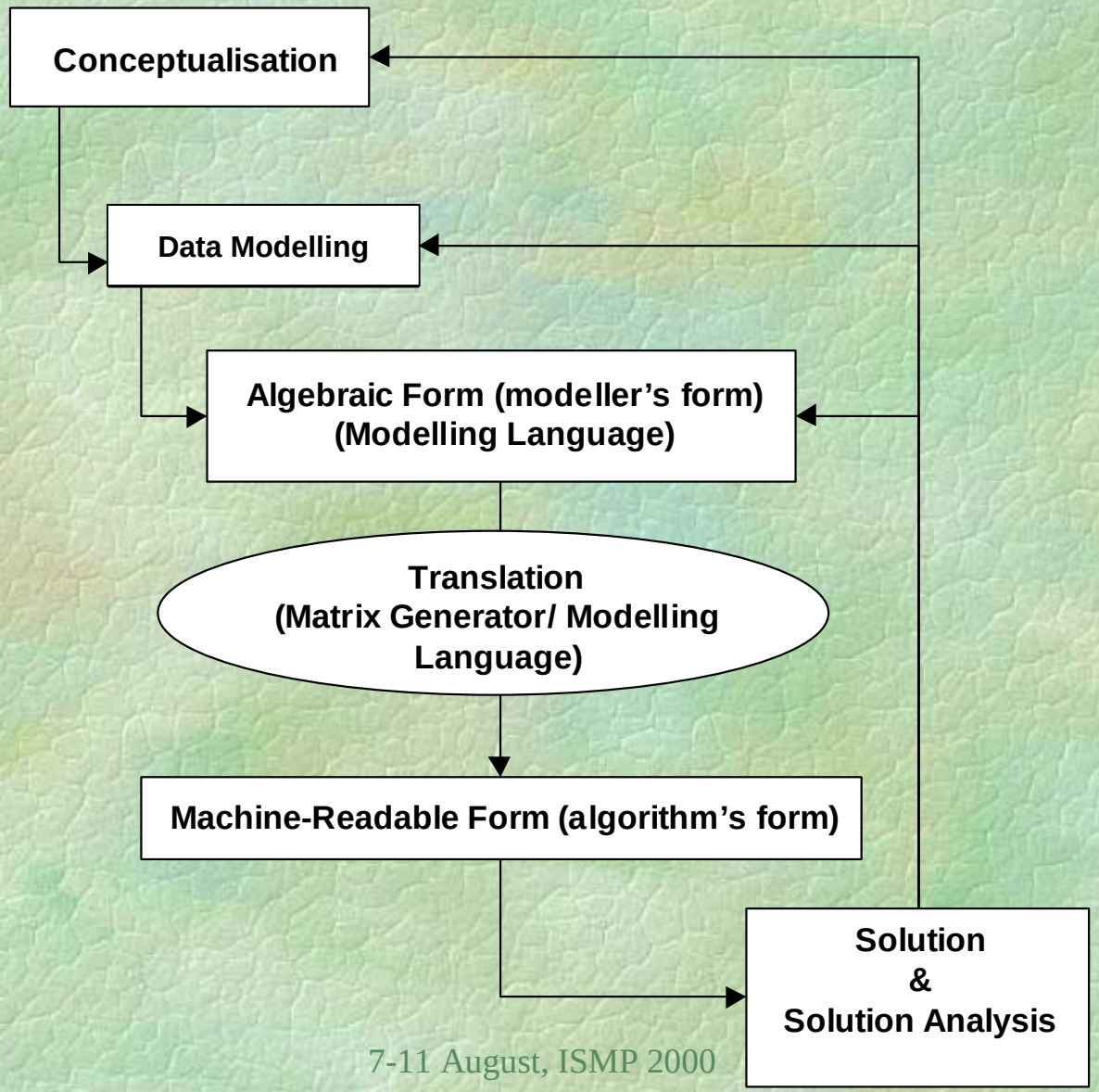
Brunel University

BRUNEL UNIVERSITY

# Outline

- Languages for MP Modelling
- Modelling Features and Extensions
  - Algebraic Features
  - Procedural Features
  - Logic
  - Modelling under Uncertainty
- Coupling of Modelling and Solving
- Analysis
- Positioning within the IS Architecture
- New Directions

# Recent View in the Process of Formulating LP Problems

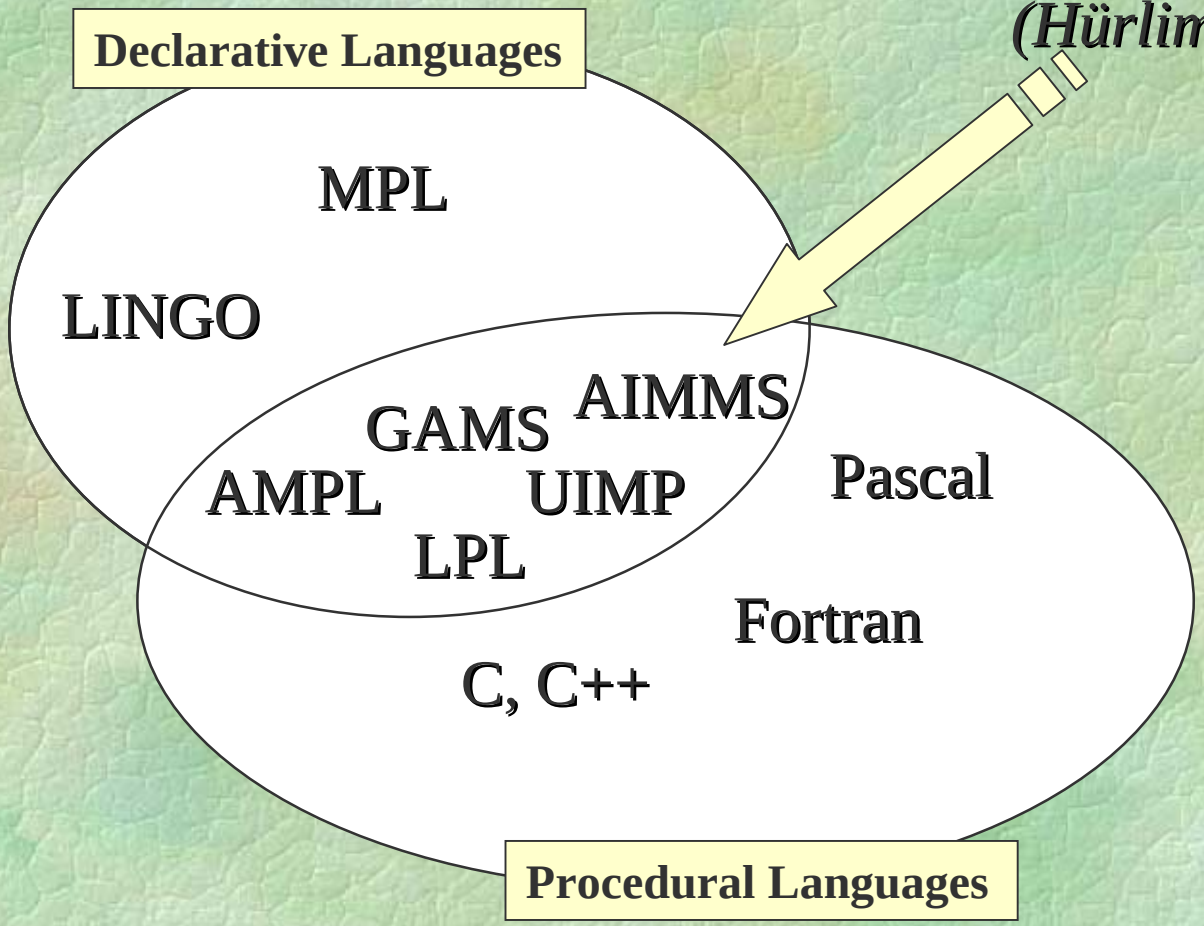





BRUNEL UNIVERSITY

# Languages for MP Modelling

*The new paradigm (Hürlimann 1998)*



# Modelling Features and Extensions

- Declarative Features
  - indices and sets

```
set RAWGAS ;

param OctaneRating {RAWGAS} ;
```

  - syntax of the linear form

*linear form* ::= [ { PLUS / MINUS } ] *linear term* [ { PLUS / MINUS } ... ]

*linear term* ::= { *simple linear term* / *compound linear term* }

# Modelling Features and Extensions

- Declarative Features

  - unit consistency

```
DATA

   RawMatCost           :=  3.00;          [$/lb]
   RawMatYield[perfume]  :=  (3, 4);       [oz/lb]
   Price[perfume, type]  := (7, 18,
                              6, 14);      [$/oz]
   ….
VARIABLES

   Production[perfume, type];              [oz]
   RawMat;                                 [lb]
    …

SUBJECT TO

   Production=RawMatYield * RawMat
```

$$[oz] = \frac{[oz]}{[lb]}[lb]$$

***LPL***

***AIMMS***

# Modelling Features and Extensions

- Procedural Features
  - modelling

    … *if-then-else ...*

  - solving

    … *solve, repeat, for, while, include ...*

    ***AMPL***

    ***AIMMS***

  - reporting

    … *display, show ...*

# Modelling Features and Extensions

- Logic

**Predicate calculus :**

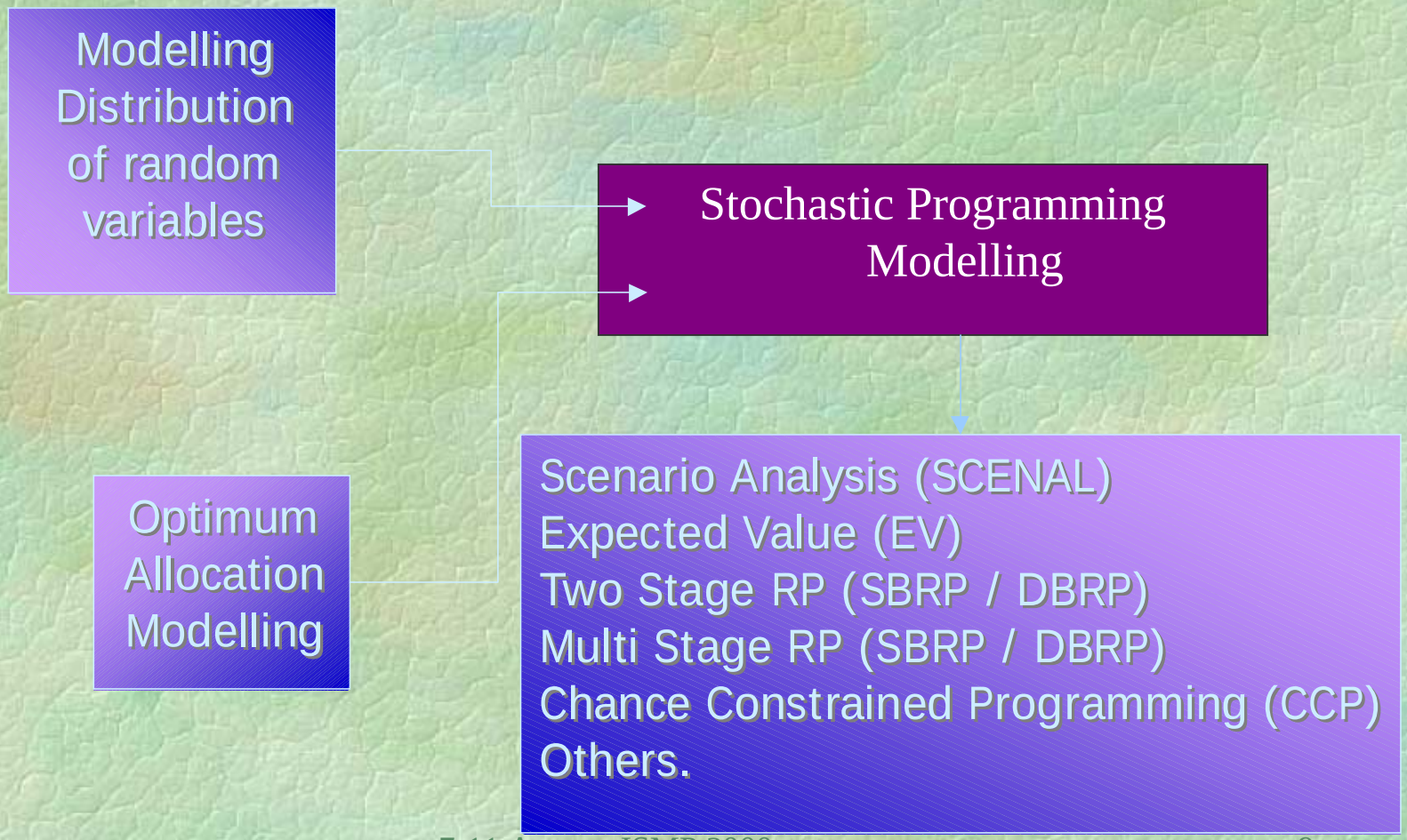
> *… and, or, at least, at most, all_different …*
>
> *… variables as indices …*

*ECLiPSe*
*OPL*
*AMPL*

**are combined with algebraic statements**

# Modelling Features and Extensions

## Optimization under uncertainty (Stochastic Programming SP)






BRUNEL UNIVERSITY

# Modelling Features and Extensions

- Keywords to connect models of randomness

| | |
|---|---|
| *… random data, time, scenarios, scenario tree,* | ***StochGen*** |
| *… stages, probabilities, chance constraints,…* | ***SMPL*** |
| | ***SAMPL*** |

# Coupling of Modelling and Solving

- Column Generation
- Rounding Heuristics, and Branch & Price Approaches
- Processing Scenarios in SP
  - Scenario Analysis
  - EVPI Test Sampling
  - Monte Carlo Sampling
- Logic Driven Solutions in Constraints
  - Inferencing
- Decompositions (Dantzig-Wolf, Lagrange, Benders')

# Column Generation

***Column Generation***

*Cutting stock*

- Use a (heuristic) procedure to enumerate (generate) a cutting pattern and introduce it in the LP/IP constraints.
- Determine the new cutting pattern by solving the knapsack problem. If no new column is found stop, else return to step 1.

***Column Evaluation***

- Generate all possible cutting patterns (columns).
- Select a subset of patterns for inclusion in the cutting stock optimisation problem.
- If not first run then use a filtering procedure to price out entering columns. Solve cutting stock optimisation problem.
- Evaluate stopping criteria, and stop or return to step 2.

# Rounding Heuristic

- To create fast MIP solutions:
  - Analyse solution of LP relaxation
  - Reuse the results as input data to fix variables
  - Establish some rounding criterion:

    Typically: the variables that are integer are excluded. The variables that are close to integer are rounded to the nearest discrete value.
  - Repeat the previous step until a discrete feasible solution is found.

# Branch & Price

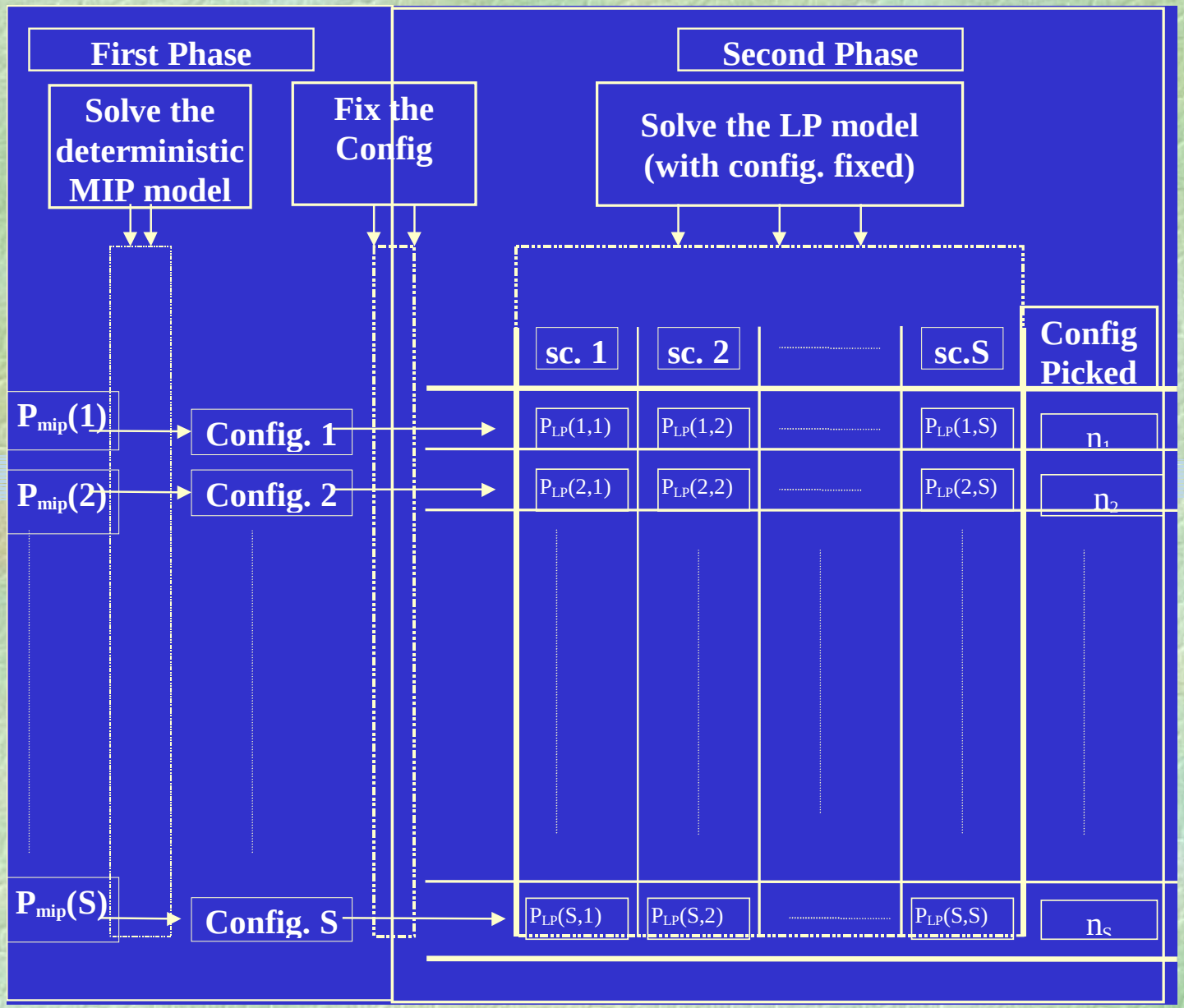
- Generate the LP relaxation with a subset of all the columns.
- Carry out a ‘price’ computation to identify columns to enter the basis.
- Solve the LP relaxation with the new columns.
- Branching when no columns price out to enter the basis, and the LP does not satisfy the integrality conditions.

# Processing Scenarios in SP

- Scenario analysis
- Sampling
  - EVPI sampling
  - Montecarlo sampling

# Processing Scenarios in SP

## Scenario analysis



# Processing Scenarios in SP

## Scenario analysis

- Solve the MIP for each scenario. The solution values of the integer variables are called "configs".
- We relax the MIPs and include the configs, forming | configs | x | Scenarios | relaxed LPs.
- For each scenario do

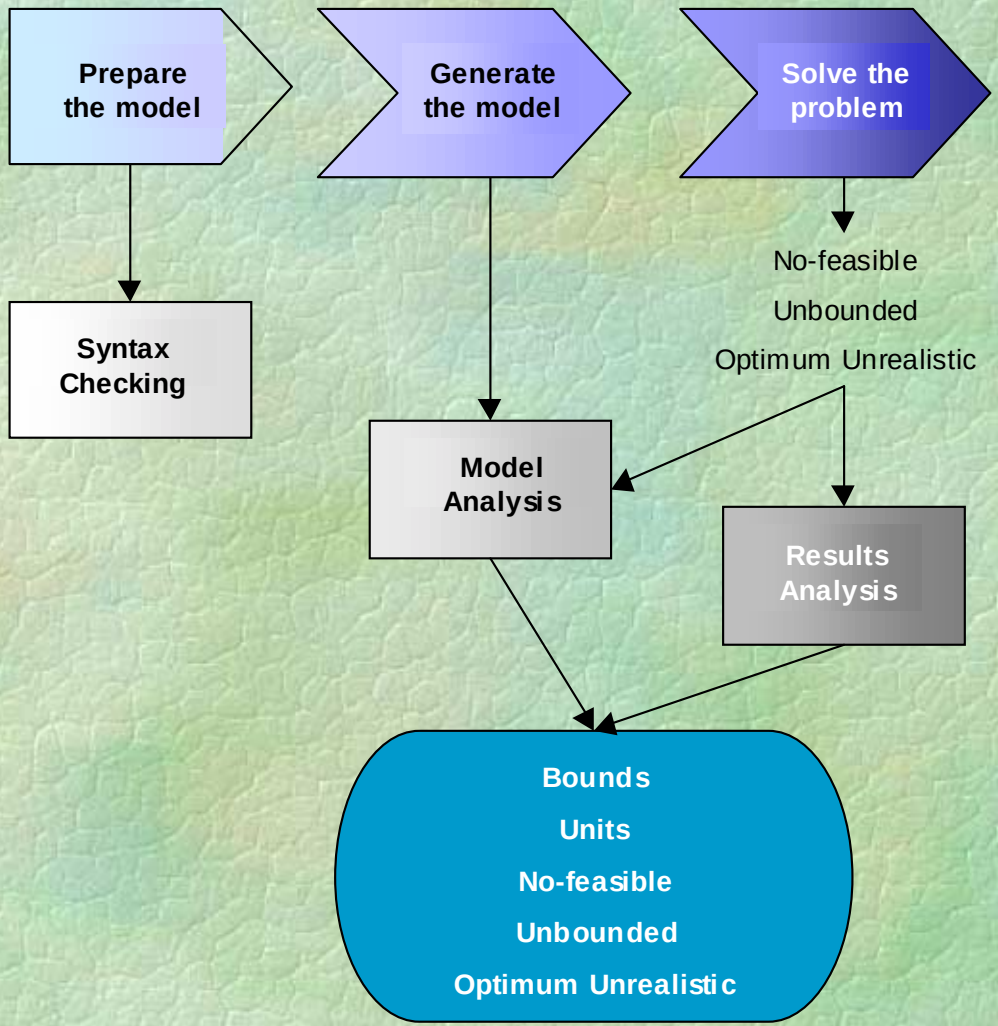
  For each config retrieved from the previous step do

  Evaluate $P_{LP}$ (config, scenario);

  Get the best $P_{LP}$ (config, scenario);
- For each config do

  Evaluate how good the config is: Count nº of scenarios that best performed with such config;
- It is very likely that the best config is the one that evaluated best in the previous step. Obviously, this depends on the subset of scenarios that has been chosen for such (evaluation) analysis.

# Analysis


Prepare the model
Generate the model
Solve the problem
Syntax Checking
No-feasible
Unbounded
Optimum Unrealistic
Model Analysis
Results Analysis
Bounds
Units
No-feasible
Unbounded
Optimum Unrealistic

# Model and Results Analysis

- **Model Analysis**

  Semantics of the Model

  Structure of the Matrix

  Structure Discovery to apply different solving algorithms

***GAMSCHK***

- **Results Analysis**

  Model Validation : Consistency …. Robustness

***ANALYZE***

# Positioning within the IS Architecture

- **The Information Value Chain**

  Data → Information → Knowledge

- **Data and Symbolic Modelling**

  Independence Model - Data

  Corporate Databases

  Tools available for Analysis of Data

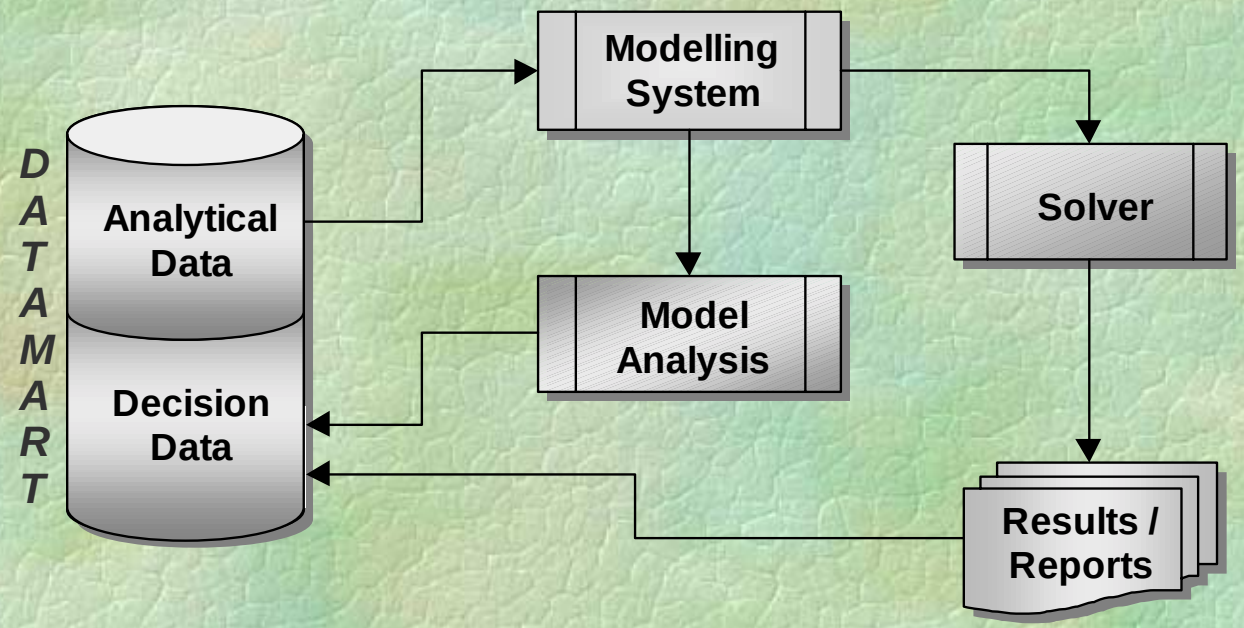
  Data Modelling …. Structure …. Symbolic Modelling

- **The Decision Database**

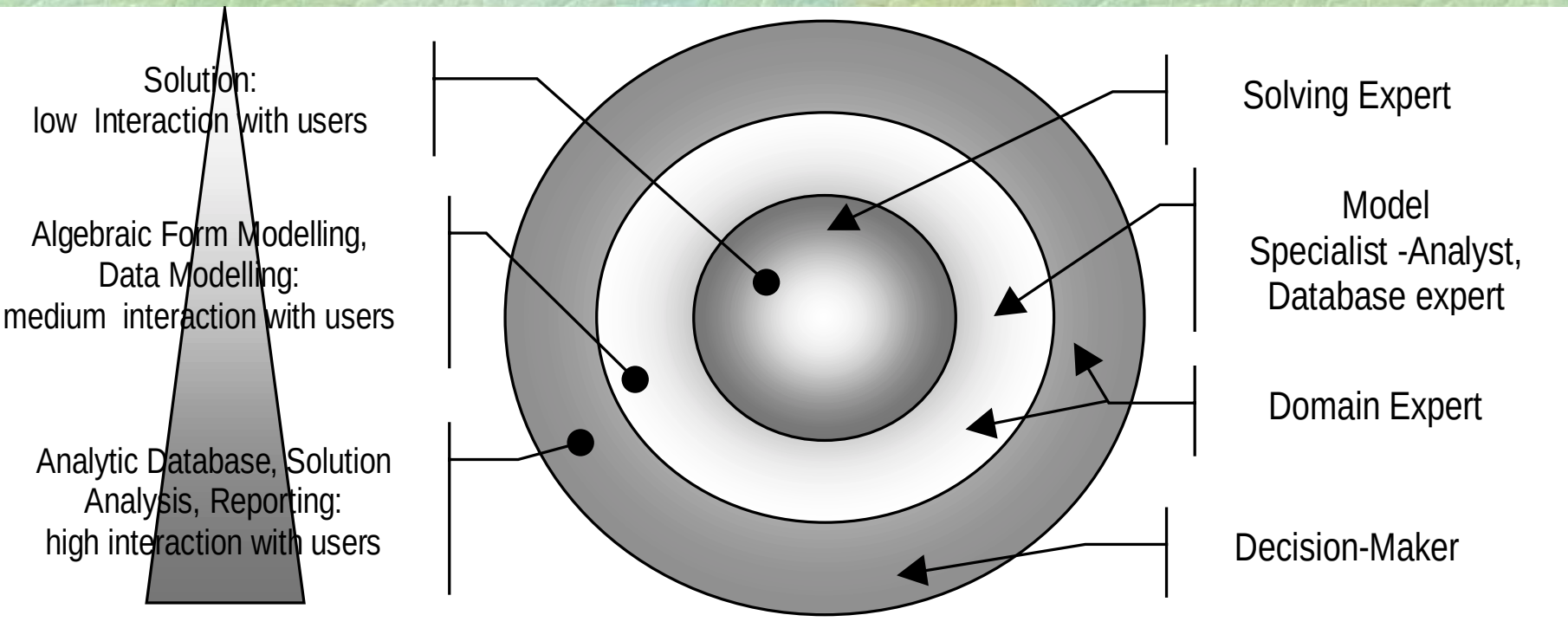
  Same Structure for the Results

# Decision data

*J.F. Shapiro*

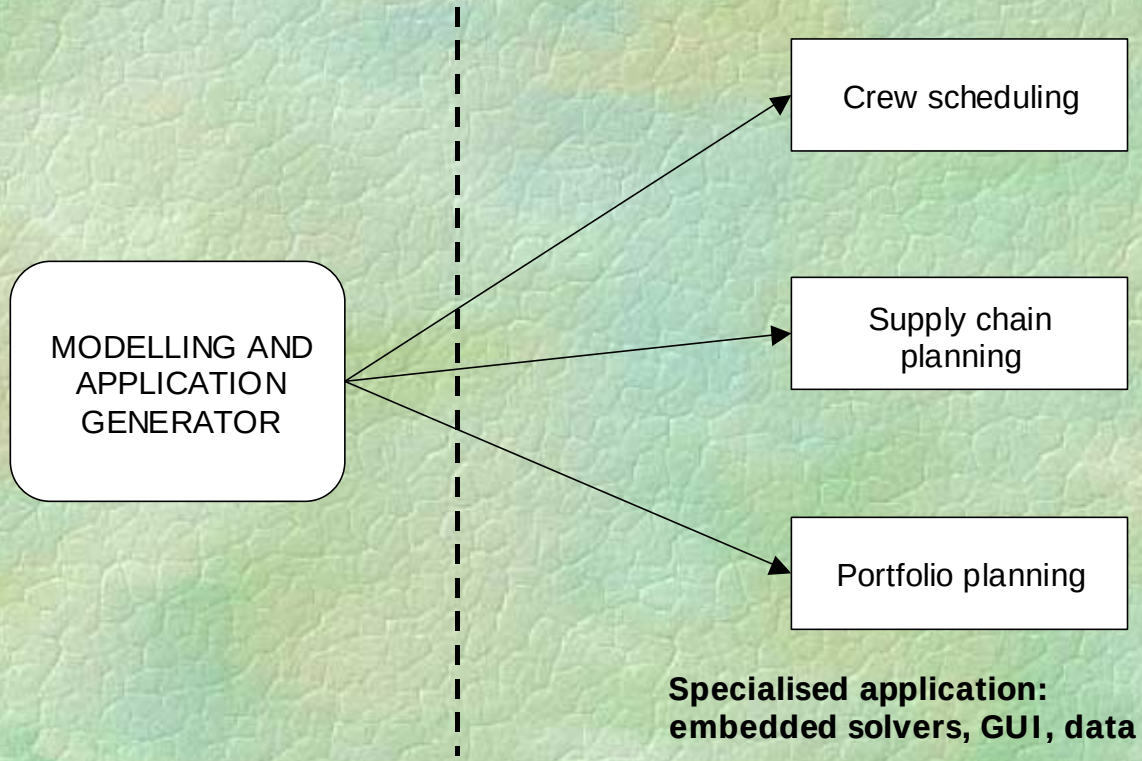


# Constituents and Level of End-User Interaction



# New Directions

- Application generation (optimization)



# New Directions

- Optimisation-based DSS Applications

  - OPL Studio

Optimisation Suite

Solver, Planner, Scheduler

Solver: LP …. LP Solver, CLP …. CLP Solver

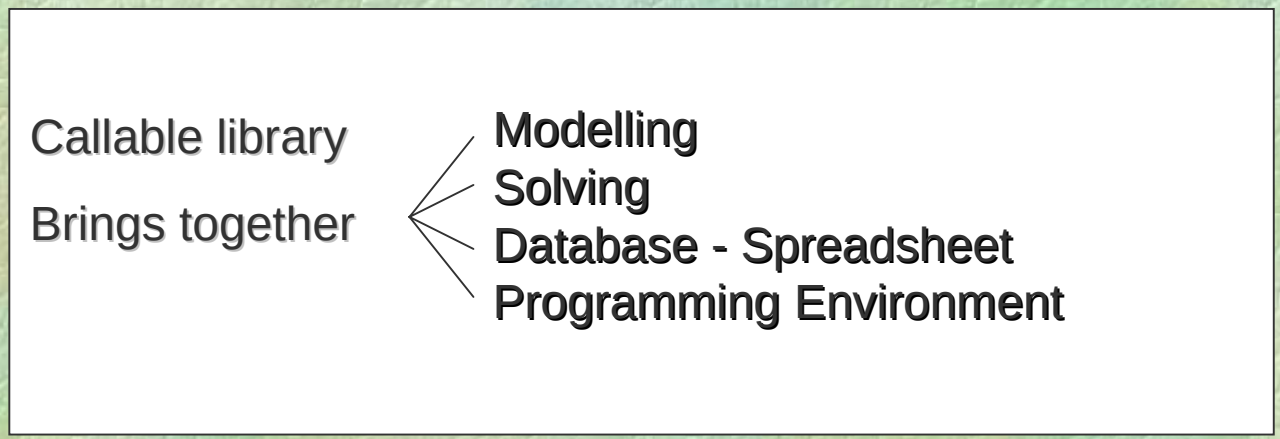
C++ code generation:

Model development …. Application deployment

# New Directions

- Optimisation-based DSS Applications

  - OptiMax



*...Thank you*