Microsoft
Solver Foundation

Microsoft Solver Foundation

What is Solver Foundation?

October 2009

Microsoft Solver Foundation Version 2.0

Table of Contents

Abstract

This document describes Microsoft Solver Foundation, a.Net-based system designed to contribute to an integrated business planning and optimization platform. Solver Foundation enables model-driven planning, scheduling, and optimization capabilities that can address business critical perspectives for process simulation, analysis, definition, deployment, automation, and full lifecycle management. Solver Foundation is a technology framework for providing these services to businesses and Microsoft technology partners by delivering a set of extensible solvers underneath a framework to aide in making better strategic and tactical business decisions using mathematical programming techniques. The Solver Foundation system is designed to run in both stand-alone mode, for example as an Add-in for Excel 2007, as well as part of a continuous planning/scheduling system with real-time constraints on performance and operations. It is designed to support its own embedding in business server and information worker systems and to run in single-node topologies. Finally, Solver Foundation provides a rich and expressive decision modeling infrastructure for higher-level modeling of business systems using either our Optimization Modeling Language (OML) or any CLS-compliant .Net languages (e.g., C#, F#, C++, IronPython, IronRuby, etc.).

The Two Worlds of Business Planning and Information Technology

Today, the leading optimization and planning company generates over $150M/year in direct revenue from its primary optimization engines (solver). In contrast, the top two downstream partners of these optimization technologies makes upwards of $3.5B/year in software revenue by embedding optimization technologies in MRP modules, planning tools, and supply-chain management solutions. The current optimization leader controls 80% of the commercial optimization marketplace, while partners and redistributors of optimization tools are realizing a much larger revenue stream that is healthy and growing. More importantly, it is revenue from mission and business critical systems, systems that are not easily commoditized and “open sourced” - the intrinsic value of the models and its capabilities is significant. Continuous planning and optimization is on a completely different level of design and engineering than a database API or forms package. Individually and in aggregate, the business planning and optimization marketplace constitutes a large revenue stream – with the majority of planning models being constructed within Excel. An economically ideal solution for partners would be to have an open framework that includes high quality solvers, and a solid model-driven integration platform. **Solver Foundation is that framework**.

We are seeing the natural logic of the “packaged software suite” trend play itself out. The value is in the suite and integration of tools, models, processes, and extensibility. Enterprise Resource Planning (ERP) was the beginning of this trend in the early 1990s. SAP went from being four ex-IBM consultants to being the largest enterprise software seller on the planet. The stakes and risks for 21st century businesses are very high and consolidation of planning, accountancy, and process control are vital to build the right perspective to make good business decisions. Increasingly, global competitiveness coupled with large financial outlays for information technology and integration of heterogeneous systems make IT investments risky affairs. Managing that IT risk and making strategic business decisions are top priorities. This is why SAP and other ERP vendors can make sales of large and complex systems – they can offer a full “360 Degree Platform” for full lifecycle of a business process. Not just business process execution and design, but the entire planning and tuning aspects through to repository management of all assets is provided in one unified command and control framework. This trend shows no sign of abating, and smart companies realize this. They understand that they need IT investments and technologies but are loathe taking chances. They make the safer ERP bet. Long term IT systems are patched and modified rather than replaced, in large part due to the “it works as is, don’t fix it” mindset as well as the presence of a lot of “invisible” homegrown software for planning and optimization.These applications, models, spreadsheets hold the business processes together. They also hold them in place to existing IT applications (read: ERP systems). Traditional IT systems have been mostly blind to the models and metrics of business planners and modelers. But this business “Dark Matter” is a large part of the strategic advantage for successful firms.

In Thomas Friedman’s **The World Is Flat**, the global, continuous optimization of supply chains for Dell (Computers) was presented as an example of the scale and sophistication required to maintain a leadership position in a world of decreasing margins and smarter competition. Dell’s advantage does not lay in industrial design, high-yield CPU manufacturing, custom chipsets, memory architecture or even operating systems - but in the ability to maximize the profit of its direct sales system and minimize the cost of supplied components and assembly of those systems. **Dell is in the business of supply chain management**. Its continuous planning and optimizing SCM system is its great advantage and one that it has successfully scaled to the global level. In another vertical area, the difference between a profitable airline company versus the majority of semi- to insolvent air transportation providers hinges upon the optimal allocation of pricing, routes, planes, and personnel. One thing is clear: the complex interdependencies of decision variables, constraints placed upon the businesses, and the need to achieve certain goals of profitability are too complex for “paper and pencil” and manual (ad hoc) methods. ERP systems have demonstrated the fallacy of “winging it”. Almost any company trying to do so today would quickly become insolvent. There is little to no recovery time. The wrong investments and decisions can completely capsize an enterprise. A failed ERP installation or botched DBMS migration can sideline a business for months or several business quarters, allowing competitors to seize the advantage and relegate the business to a footnote or Harvard Business School case study. Whether it is the wrong allocation of distributors, poor IT infrastructure choices, improper management of suppliers, or incorrect pricing of goods and services - the results can be fatal.

A business must balance strategic decision making with tactical investments and operational efficiencies. Today, the majority of information technology (IT) systems are operationally focused and non-strategic. They focus on efficiency not strategy. They do not help the business decision makers plan and optimize at a more global and strategic level. In essence, they are a cost of doing business and not a competitive advantage. The rise of open source software for many current IT systems, from application servers to database management systems, is a harbinger of the **platform composting** that is happening as a result of business pressures and the drive towards hyper-efficiencies of scale. The current crop of IT systems is designed to make the status quo more manageable. ERP systems, to their credit, provide the closest possible technology to help businesses plan their manufacturing, manage accounting, and schedule resource flows. All of this business process management is today expensive and complex to implement, but it is the only real game in town.

Dell's supply chain innovations and continuous planning systems required a systematic change in the global business process for sourcing components for Dell machines – based on price, yield quality, transportation costs, and demand levels. They require a new kind of business process formulation and improvement system. This kind of system is the focus of Solver Foundation. The major shortcoming of IT systems, models and tools is that they encompass their own internal mechanisms without reference to the dynamics of the businesses they are required to support. They lack **business context**. IT systems take on their own closed system dynamics, evolving at rates different from those required by the marketplace or by the enterprise's business models and networks. IT technology does provide dashboards, key performance indicators, business intelligence services, remote management, and semi-automation of value-producing business processes. However, the global optimization of the company's strategies and processes is difficult or impossible to ascertain with any predictability given these point-solutions. This ultimately leads to the **two-world syndrome** – the world of business planning and management by the front office staff and the semi-autonomous world of business IT management by MIS personnel drifting slowly away from the ever-changing and evolving business objectives. The rates of change in the front and back office are quite different today. Reducing or eliminating this discrepancy is key.

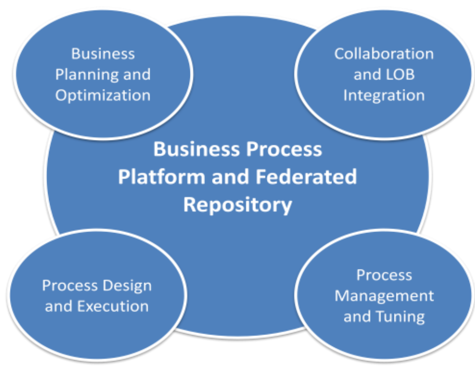


Figure A: **Business Process Platform and Federated Repository.** (Note separation of the front office-oriented system functions (planning, optimization, and collaboration) and the IT/back-office-oriented functions (process design, execution, management and tuning))

Presently, these are two worlds without a common language. Very few models, metaphors, or tools exist to span the gulf although many try. From a software perspective, the results have been desultory except in the ERP sector. But even that relative victory is pyrrhic. There is a compelling need for a 360-Degree business planning and optimization platform that can close the loop of decision making through to operations management and process improvement. Solver Foundation is one component, focusing on the modeling and support of key business decisions, variables, constraints, and objectives in one simple, uniform, and scalable system. It is a bridge from front to back office – mathematically rigorous and automatable within the IT world but designed for convenience and ease of use by business people in the front office. It succeeds by acknowledging that there are two worlds and languages in play and building a two way transformer that makes either side equal partners in the design, development, and operation of value-generative business processes and systems. Its integration with Microsoft Office (e.g., Excel, SharePoint) is an example of creating business models with familiar tools for front office personnel. Solver Foundation's sophisticated solving technologies make it possible to create feasible and optimal business plans with high degrees of accuracy and responsiveness. In a dynamic business world of increasing risk and interconnection, finding one's way to the right answers necessitates having the right kinds of tools and models.

Solver Foundation is designed to help avoid a part of this self-fulfilling dynamic. How? It provides a system of modeling and testing business assumptions and strategies in a scalable and continuous fashion. It is an aide to business people and IT. This is often referred to as operations research and mathematical programming in the literature. Solver Foundation frees the business planner to try out ideas, model their likely developments using stochastic techniques, and to better understand the effects of changes in their strategies and decisions on the final business outcomes. Solver Foundation is best thought of as a combination of the following.

- Combinatorial Optimization and Mathematical Solver Technologies
- Stochastic Modeling and Simulation Technologies
- Business Modeling Infrastructure with Higher-Level Declarative Language (OML)
- Interactive Tools for Modeling, Solving, and Results Analysis[1]
- Programming Model and Object Model for Construction of Solvers
- Embeddable Runtime Support for Hosting Solvers within Business Systems (LOBi, ERP, BPMS, BI, etc.)

This is a broad characterization. Solver Foundation is a modeling environment of a specific sort: it focuses attention on the key factors and outcomes of decisions. Solver Foundation is designed to support many kinds of common and business critical modeling activity (by non-IT personnel). These are referred to as **model domains**. The easiest analogy to this is classes of problems – how hard they are to solve, the kinds of values the solutions may take, and the constraints involved in the solution of the business problems. Solver Foundation supports all of the common domains that a business modeler and planner would require – including working with models that involve:

- Linear Programming (Primal Dual Simplex and Interior Point Techniques)
- (Mixed and Binary) Integer Programming (From Branch and Bound to Cutting Plane Techniques)
- Finite Domain Programming (From Constraint-Logic Programming to Adaptive Local Search Algorithms)
- Nonlinear Programming (From Convex (Quadratic) Programming to Non-Convex Techniques)
- Stochastic Programming (What if?-style Techniques with Markov Decision Procedures and Stochastic Variables)
- Dynamic Programming (Deterministic and Nondeterministic Techniques)

"Programming" in this case does not refer to IT programming. The term derives from military logistics and planning where a "program" denotes a process for allocation of goods, services, and personnel. We trust the usage will not prove challenging to disambiguate from the IT version of programming. It is best thought of as a combination of (a) modeling, (b) solving and (c) committing resources. This is a typical phase one model and is a major focus for Solver Foundation.

[1] Solver Foundation proposes to produce very little besides the Excel add-in and our CLI in terms of user interface tools. We expect that others will embed Solver Foundation technologies in their preferred user model and toolsets.

The Business Value of Planning and Optimization

The foregoing can be viewed by some as very abstract. It is hard to put a monetary figure on the value of good phase one modeling at times. We think that a little research can uncover obvious and compelling proof that phase-one modeling and technology investments is important. Based on Hillier and Liebermann's **Introduction to Operations Research** (7^{th} Edition), we can call out some well known "wins" described in the literature. The following table summarizes some of the higher profile operations research (OR) projects. Add to this the demonstrable successes of Wal-Mart, Ford Motors, Costco, and Dell in managing supply chains, warehousing logistics, and pricing based on applying OR techniques at scale.

Organization	Application	Year	Savings
IBM	Inventory Management and Supply Chain Optimization	1990	$20M/year and $250M less inventory
IBM	Dutch Railway Timetable Redesign	2008	$105M
Delta Airlines	Maximizing Profits of Assigning Airplane Types for 2500 Domestic Airline Routes	1994	$100M
Digital Equipment Corporation (HP)	Restructuring Global Supply Chain	1995	$800M
Hewlett-Packard	Redesign of Production Line for Printers (Buffers)	1998	$280M More Revenue
South African Defense Force	Redesign of Size and Shape of Armed Forces and Weapons Systems	1997	$1.1B
AT&T	Design of PC-based Business Customer Call Center System	1993	$750M
P.R.C.	Optimal Selection and Scheduling Of Energy Production and Sourcing	1995	$425M
Proctor and Gamble	Redesign of North American Production and Distribution System to Reduce Costs and Improve Time to Market	1997	$200M
Texaco, Inc.	Optimal Blending of Gasoline Ingredients to Meet Quality and Regulatory Requirements	1989	$30M

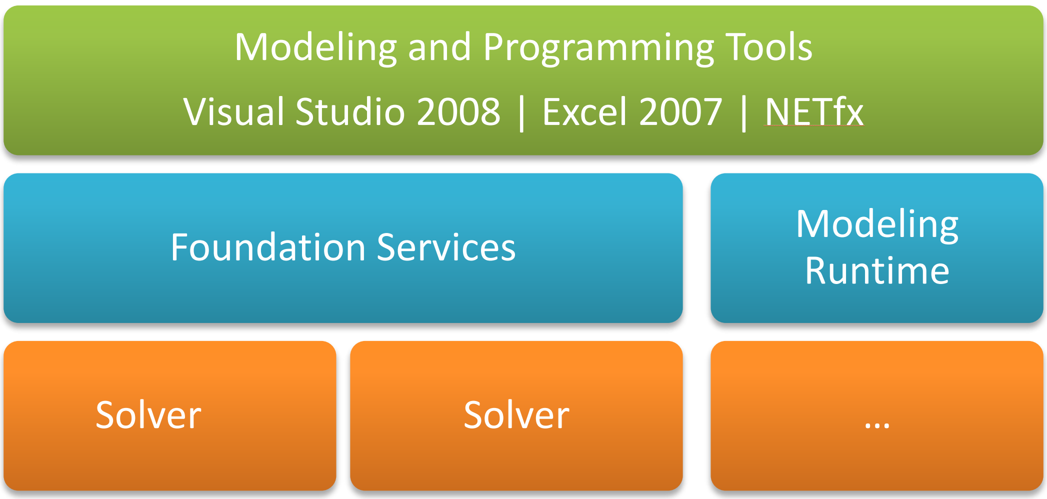
In most cases, the savings or increased revenues and profits is measured in hundreds of millions or occasionally in billions of dollars. This is not pocket change.

Not all planning and optimization use cases have these kinds of returns to show for the efforts involved. There are no silver bullets and we acknowledge this fact. Today, modeling for optimization and decision making is fraught with issues of complexity, cost, training, etc. That competitive enterprises staff up internal operations research departments should not come as any surprise though. As with the Delta Airlines example, sometimes the difference between insolvency and profit is the 1-3% "squeezed out" by proper use of mathematical programming and optimization techniques. People are willing to pay large sums of money for that 1-3% cushion.

The savings are evidenced in all important verticals – pharmaceuticals, transportation, telecommunications, retail, federal and military systems, production, manufacturing and energy management. Taken as a trend though, one does note that many of the most competitive business segments and industries make significant investments in operations research and planning in their strategic business processes. Tuning the right processes can have substantive bottom line impact.

Logical Architecture

The following diagram offers a simplified view of the architectural components of Solver Foundation. The system is best seen as being composed of three well-factored layers: (1) Tools and Surface Languages (C#, OML, and Excel Expressions), (2) Solver Foundation Services and Modeling Runtime and (3) Solver Plug-ins. Solver Foundation ships with several built-in solvers and a well-defined open API model for third party solver integration. All solvers can play on a level playing field and derive the benefits of the Solver Foundation Services, Modeling Runtime, deep NETfx support, etc.



The solvers plugged into the framework are implemented using a CLS-compliant .Net language. For this release, C#, F#, and IronPython are the preferred languages but any one is technically feasible. C# is preferred due to the integration of operator overloading (in-place operators for convenient constraint expression definition) as well as the support for LINQ integration. Readability and ease of use are important "soft factors" for adoption of modeling systems and Solver Foundation views C# as an excellent language for targeting IT developers.

Models and Solvers

The best way to understand modeling capabilities is to start with a simple model that introduces the key concepts. We note that the model we describe, NASA Project Budgeting, is technically a "toy" from an industrial perspective.[2] It is purely educational. Model sizes for typical models in industry run to the following sizes.

- Small "Strategic" Models (100s to 1000s of Decision Variables and Constraints) (Mostly used in **Strategic Modeling** and Prototypes)
- Medium Models (10,000s to 100,000s of Decision Variables and Constraints) (Most Models fit into this Category with 10,000s as a good mid-point)
- Large "Operational" Models (100,000s to 1M+ Decision Variables and Constraints) (Mostly used in Arbitrage, Pricing, etc.)
- Very Large Models (100s of millions of Decision variables and Constraints) (Complicated and Sophisticated Cases)

[2] Solver Foundation ships this and many other models in several formats: OML, C#, F#, IronPython, VB.NET, C++, etc.

A Solver Foundation (linear) model is a remarkably simple entity. It consists of the following components.

- **Parameters**. Defining the data that plugs into the model or more specifically the constants in the objective functions and coefficients in the constraints.
- **Decisions**. The 'outputs' of the solver – they are the results of the model being solved. Decisions take on values over numeric or symbolic ranges: reals, integers, finite domains (sets)[3]
- **Constraints**. These are the business constraints over the Decisions (Expressions involving variables)
- **Goals**. Defining the business goal or goals you are trying to accomplish. These are Objective Function(s) which are sought for by the business modeler, usually to minimize some cost)

The model is a representation of the key inputs and outputs in making a business decision. It abstracts away IT considerations like programming languages, forms packages, XML descriptors, and UX frameworks, and focuses on modeling the business process and the key aspects of the process that will generate value or consume resources. The trick and challenge is to take that insight and provide the ease of use and approachability that Microsoft is known for in IT and apply it to the new business arena of business planning and optimization.

The basic model definition, expressed in OML, for a Solver Foundation model looks like this.

```
MyModel := Model[Decisions[…], Goals[…], Constraints[…]];
```

The basic model definition, expressed in C#, for a Solver Foundation model looks like this.

```
Model model = context.CreateModel();
model.AddDecisions();
model.AddConstraints();
context.Solve();
```

Solver Foundation's use of operator overloading makes it natural for constraint expressions to be specified within C# without recourse to awkward prefix syntax or expressions-as-strings. A simple example would be:

```
model.AddConstraints(true,
      5 * X01 + 7 * X02 + 4 * X03 + 3 * X04 <= 14,
      X01 + X02 + X03 + X04 <= 2,
      X02 - X04 <= 0,
      X01 + X03 <= 1 );
```

The former corresponds directly to the **conceptual model** of most business planners and to the way that MBA/OR people are taught to use current optimization modeling systems. Models require specification of the domains of variables and the interrelationships of decision variables are modeled as constraints. The C# code sample is a natural representation for the C#/.Net developer – one only needs to concentrate on specifying the right relationships – the Solver Foundation system is well integrated with the C# and Visual Studio tool experience.

[3] Solver Foundation supports **arbitrary precision numerics** so that exact calculations and solutions are possible with its models.

The OML representation is more compact and oriented towards interactive model building and solving. Solver Foundation is designed to support direct OML definition and model evaluation, leaving choice of surface language up to the user. This flexibility allows Solver Foundation to cater to multiple constituencies.

Models are one of the key aspects of Solver Foundation; solvers are the other critical ingredient. In fact, Solver Foundation stands or falls on the quality of its solvers. The goal of a solver in this system is to ascertain the following about a model.

- Is the model feasible? (i.e., is there any possible solution to the specified constraints?)
- Is there an optimal solution to the model? (i.e., is there a best answer to the model's constraints and goals?)
- Is the model tractable? (i.e., is it possible to conveniently and practically analyze the results of the model's solution?)
- Is the model valid? (i.e., to what degree do the inferences drawn from the model and results hold true for the "real" system?)
- What is the impact of variable <x> to the solution? (This is referred to as **sensitivity analysis** and is very useful to business planners to understand the impact that certain "inputs" will have in the final "output" (goal))
- How accurate is the model? (i.e., given the kinds of numeric types being used, how susceptible to rounding error and precision issues are the solutions?)
- What if I make <x> optional? (i.e., given a set of constraints, can one arrive at a feasible solution if some of the constraints are "weakened"? Important for iterative modeling and planning)

There are several aspects to a successful business planning models – determining whether it is feasible, if an optimal solution exists, how sensitive the results are to variations in inputs, and finally the ease of iterating the model and trying out various combinations of constraints. This latter feature, support of "soft constraints", is a major boon to business modelers. Notable examples of usage of soft constraints include scheduling of NFL games and the development of group vacation planning software. In these cases, the ability to solve for a subset of the constraints is important. The planner is then able to adjust other constraints and continue refinement of the model. Feasibility checking, automatic model simplification, iteration, solver heuristic hints, partial solving with resumption and soft constraints are tangible and important benefits to industrial applications of planning and scheduling models.

It is important to note that the complexity of models and their solution is largely a function of the domains of the decision variables and the degree of the equations involved in the constraints (i.e. linearity vs. non-linearity). Models with real-valued variables and linear constraints are generally much easier to solve than those with integer-values and/or non-linear constraints. The disparity lies in the techniques used to "search" for a solution. Linear programming and model building has a long history of techniques – Simplex, Interior Point, etc. – with well defined runtime characteristics. Integer and non-linear programming solvers have to rely on different techniques – and many integer-based models cannot be optimally solved. This is more than acceptable to industrial practitioners – **95%+ are mostly interested in near-optimal or feasible solutions.** This fact makes model solving a more tractable exercise for large integer-models. Near optimality and basic feasibility are the two most prevalent outcomes or solution classes that industrial practice requires.

The following model and scenario is taken from Ronald Rardin's **Optimization in Operations Research**.[4] It deals with the capital/project budgeting for NASA Space Missions. The basic problem is: given a finite budget, how should one optimally choose which projects to fund that meet all of the requirements (monetary and other resource constraints) imposed and which deliver the maximal value to the agency. Each project has costs associated with it as well as interdependencies with other projects. These are modeled as constraints in the (linear) model. That is the business decision that needs to be made. The following table summarizes the key budgetary requirements for the missions, their ranked value, and the dependencies across missions.

Budget Requirements for NASA Space Programs

		Year Range and $B Needed							
Mission ID	**Name**	**2000-2004**	**2005-2009**	**2010-2014**	**2015-2019**	**2020-2024**	**Value**	**Not With**	**Depends**
1	Communications Satellite	6					200		
2	Orbital Microwave	2	3				3		
3	Io Lander	3	5				20		
4	Uranus Orbiter 2020					10	50	5	3
5	Uranus Orbiter 2010		5				70	4	3
6	Mercury Probe			8	8	4	20		3
7	Saturn Probe	1	8	1			5		3
8	Infrared Imaging				5		10	11	
9	Ground-based SETI	4	5				200	14	
10	Large Orbital Structures		8	4			150		
11	Color Imaging			2	7		18	8	2
12	Medical Technology	5	7				8		
13	Polar Orbital Platform		1	4	1	1	300		
14	Geosynchronous SETI		4	5	3	3	185	9	
Budget		10	12	14	14	14			

The goal is to make a feasible (and, if possible, optimal) choice. The constraints within this model include projects that cannot be, or must be, co-funded. Additionally, there is a different total budgetary cap per year. In OR literature, this class of problem is a multi-dimensional knapsack. Using Solver Foundation's interactive modeling language (OML), the following would constitute a well-formed model for the problem. The output desired by the problem is a "yes"/"no" (binary integer) for each potential mission/project NASA has planned over a time period. The key modeling activity is to map all of the salient aspects of the problem into a model and then solve it with the appropriate solver (in this case binary integer solver).

[4] Ronald Rardin, **Optimization in Operations Research**, pg.562-566

```
Model[

Decisions[
  Integers[0, 1],
  CommunicationsSatellite, OrbitalMicrowave, IoLander,
  UranusOrbiter2020, UranusOrbiter2010, MercuryProbe, SaturnProbe,
  InfraredImaging, GroundSeti, OrbitalStructure, ColorImaging,
  MedicalTechnology, PolarOrbital, GeosynchronousSeti
],

Goals[

  Maximize[
    200 * CommunicationsSatellite + 3 * OrbitalMicrowave + 20 * IoLander +
    50  * UranusOrbiter2020 + 70 * UranusOrbiter2010 + 20 * MercuryProbe +
    5   * SaturnProbe + 10 * InfraredImaging + 200 * GroundSeti +
    150 * OrbitalStructure + 18 * ColorImaging + 8 * MedicalTechnology +
    300 * PolarOrbital + 185 * GeosynchronousSeti
  ]],

Constraints[

  6 * CommunicationsSatellite + 2 * OrbitalMicrowave + 3 *
      IoLander + SaturnProbe + 4 * GroundSeti + 5 * MedicalTechnology <= 10,
  3 * OrbitalMicrowave + 5 * IoLander + 5 * UranusOrbiter2010 + 8 *
      SaturnProbe + 5 * GroundSeti + 8 * OrbitalStructure + 7 *
      MedicalTechnology + PolarOrbital + 4 * GeosynchronousSeti <= 12,
  8 * UranusOrbiter2010 + MercuryProbe + 4 * OrbitalStructure + 2 *
      ColorImaging + 4 * PolarOrbital + 5 * GeosynchronousSeti <= 14,
  8 * MercuryProbe + 5 * InfraredImaging + 7 * ColorImaging +
      PolarOrbital + 3 * GeosynchronousSeti <= 14,
  10 * UranusOrbiter2020 + 4 * MercuryProbe + PolarOrbital + 3 *
      GeosynchronousSeti <= 14,
  UranusOrbiter2020 + UranusOrbiter2010 <= 1,
  InfraredImaging + ColorImaging <= 1,
  GroundSeti + GeosynchronousSeti <= 1,
  UranusOrbiter2020 <= IoLander,
  UranusOrbiter2010 <= IoLander,
  MercuryProbe <= IoLander,
  SaturnProbe <= IoLander,
  ColorImaging <= OrbitalMicrowave
  ]

];
```

Solver Foundation would solve this problem with the following command:

```
Solve();
```

The result of solving the model would be, if the model proves to be feasible (i.e. having a possible solution), an assignment of values to the decision variables that meet the goal (objective function) specified in the model. In this case:

```
{-765, { CommunicationsSatellite->1,
      OrbitalMicrowave->0,
      IoLander->1,
      UranusOrbiter2020->1,
      UranusOrbiter2010->0,
      MercuryProbe->0,
      SaturnProbe->0,
      InfraredImaging->1,
      GroundSeti->0,
      OrbitalStructure->0,
      ColorImaging->0,
      MedicalTechnology->0,
      PolarOrbital->1,
      GeosynchronousSeti->1}
}
```

The projects assigned a “1” value in the output are those that would be funded by the NASA budgets over 2000-2024 given the constraints and objective specified for the business decision. This specific model is an example of a Binary Integer Programming (BIP) problem. In such problems, the goal is to make a yes/no decision about something. The first value in the result is the computed value of the objective function. As we can see, Solver Foundation models are **executable specifications**. Once specified, different solvers may be applied either manually or relying upon the Solver Foundation runtime to analyze and execute the appropriate solvers in either pipelined or parallel fashion. In the interactive OML Shell, the high-level Solve[] verb is used as an entry point to the solver infrastructure. The user also has the option of directly invoking a specific solver, using say Solve[] as the verb to invoke.

Solver Foundation solvers are designed to expose a rich and uniform interface. They are “pluggable” into an open specification that allows Microsoft as well as third party solver vendors to play in this arena. Solver Foundation would provide the integration framework to 3rd party solver vendors (e.g., CPLEX ®) into the rest of the NETfx/CLR system and permit the end user to choose, via declarative options, which solver and configuration to utilize.

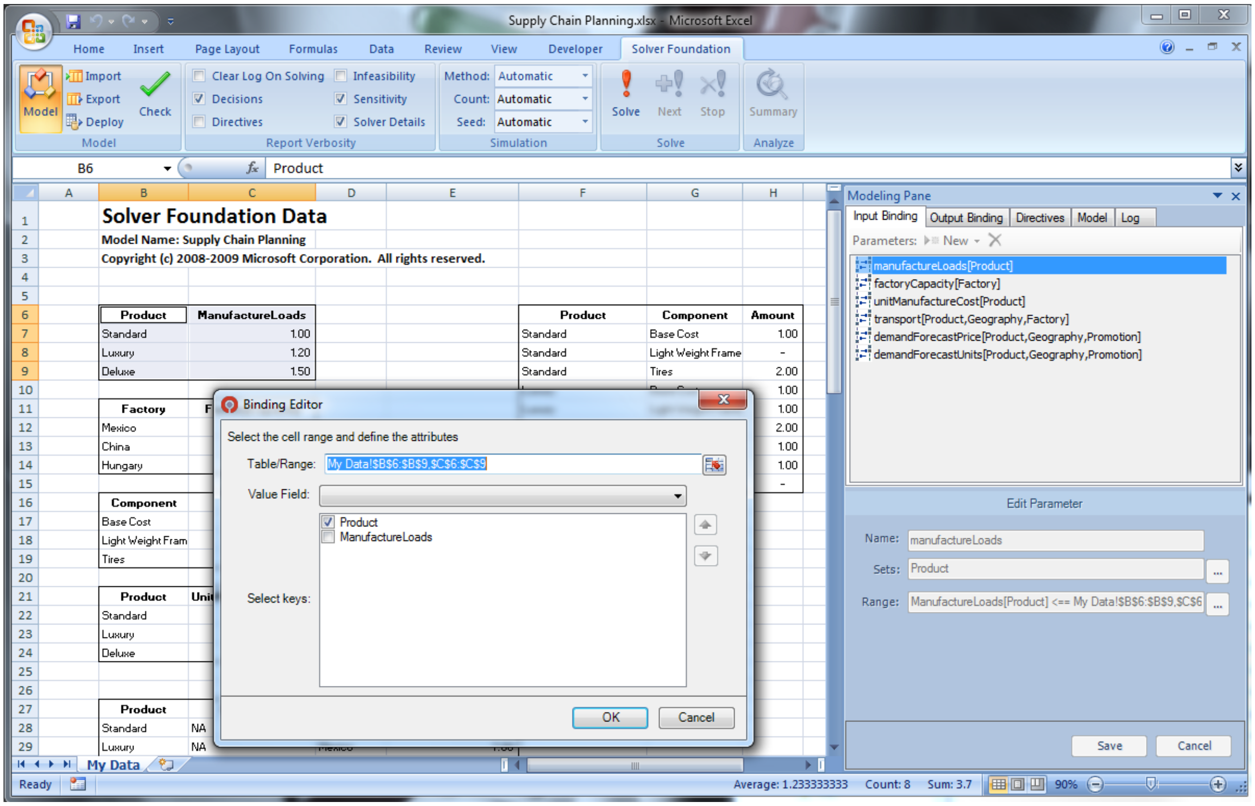
One of the key features developed for Solver Foundation is the ability to reflect on the structure and shape of models to determine, without end user explicit input, the likeliest and best solvers to utilize. In the example above, the runtime inferred the use of the BIP Solver from analyzing the "type" of variables and the constraints between them. Because the type was specified as integer constrained to zero or one, the runtime was able to infer that the model was a Binary Integer Program. This knowledge will have significant runtime performance implications in the final release of Solver Foundation. The model preprocessing and heuristic inference is a key stepping stone to the level of end user friendliness that operations research technologies need to deliver for the marketplace to really open up. Much of Solver Foundation research and design is targeted at the auto-sensing of model types and application of model transformations to deliver faster and more stable solutions.

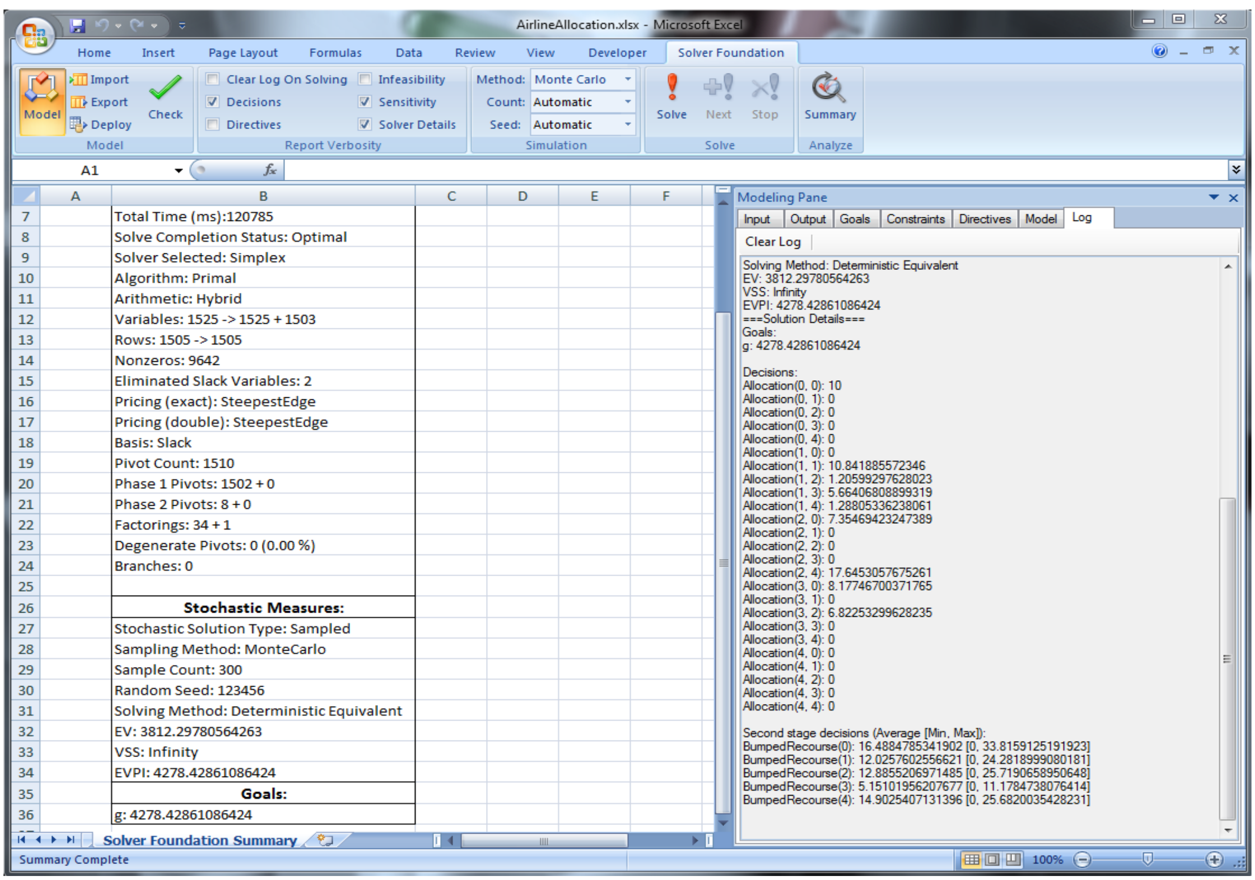
Solver Foundation models (expressed in OML within the Excel Add-in or in the nearly isomorphic C# object model) are designed for interoperation with current industry standards, including bi-directional transformation to and from (industry standard) MPS formats. These models form a baseline comparison point across solver systems and many large enterprises have significant internal investments in MPS models. Today, Solver Foundation can read and write those models. This makes the integration of Solver Foundation modeling tools and solvers a straightforward proposition for enterprises – they do not have to throw away their system of record linear models. Solver Foundation is also be releasing a specification of an XML interchange format, currently named OML, for transportation and integration of modern web services and lifecycle management systems. The same model transformation system will be used as in MPS. The transformation grammars are pluggable as well. This is partially based on the OML language and runtime which acts as an Interlingua model pipeline. Models in MPS format can be injected and solved, with results returning in OML, or C# object format. This interchange and interoperability makes the system quite flexible.

Excel and Office Business Intelligence Integration

Solver Foundation for Excel Add-in provides a unique opportunity to blend the power of Excel with Solver Foundations modeling and solving capabilities. This enhanced spreadsheet can then be shared with any other Excel2007 users via SharePoint Server, Exchange (email), etc. As an example, a Supply Chain problem is depicted below using the following constraints and goals:

a. $\sum_{p \in Products, a \in Areas, r \in Promotions} Plan[p,a,r,f] \leq capacity[f], \; for\ all\ f \in Factories$
b. $\sum_{r \in Promotions} Plan[p,a,r,f] = ShippingPlan[p,a,f], \; for\ all\ p \in Products, a \in Areas, f \in Factories$
c. $\sum_{f \in Factories} Plan[p,a,r,f] = SalesForecastUnits[p,a,r], \; for\ all\ p \in Products, a \in Areas, r \in Promotions$
d. $\sum_{f \in Factories} (unitCost[p] + transportationCost[p,a,f]) \times ShippingPlan[p,a,f] = TotalCost[p,a], \; for\ all\ p \in Products, a \in Areas$
e. $\sum_{r \in Promotions} SalesForecastUnits[p,a,r] \times demandForecastPrice[p,a,r] = TotalRevenue[p,a], \; for\ all\ p \in Products, a \in Areas$
f. $\sum_{p \in Products, a \in Areas} (TotalRevenue[p,a] - TotalCost[p,a]) = TotalProfit$
g. $SalesForecastUnits[p,a,r] \leq demandForecastUnits[p,a,r], \; for\ all\ p \in Products, a \in Areas, r \in Promotions$


Supply Chain Planning.xlsx - Microsoft Excel
Home Insert Page Layout Formulas Data Review View Developer Solver Foundation
Model Import Export Deploy Check
Clear Log On Solving Decisions Directives Infeasibility Sensitivity Solver Details
Report Verbosity
Method: Automatic Count: Automatic Seed: Automatic
Simulation
Solve Next Stop Summary
Analyze
B6 Product
Solver Foundation Data
Model Name: Supply Chain Planning
Copyright (c) 2008-2009 Microsoft Corporation. All rights reserved.
Product ManufactureLoads
Standard 1.00
Luxury 1.20
Deluxe 1.50
Product Component Amount
Standard Base Cost 1.00
Standard Light Weight Frame -
Standard Tires 2.00
Factory
Mexico
China
Hungary
Component
Base Cost
Light Weight Fram
Tires
Product
Standard
Luxury
Deluxe
Binding Editor
Select the cell range and define the attributes
Table/Range: My Data!$B$6:$B$9,$C$6:$C$9
Value Field:
Select keys:
Product
ManufactureLoads
OK Cancel
Modeling Pane
Input Binding Output Binding Directives Model Log
Parameters: New
manufactureLoads[Product]
factoryCapacity[Factory]
unitManufactureCost[Product]
transport[Product,Geography,Factory]
demandForecastPrice[Product,Geography,Promotion]
demandForecastUnits[Product,Geography,Promotion]
Edit Parameter
Name: manufactureLoads
Sets: Product
Range: ManufactureLoads[Product] <== My Data!$B$6:$B$9,$C$6
Save Cancel
My Data
Ready Average: 1.233333333 Count: 8 Sum: 3.7 90%


AirlineAllocation.xlsx - Microsoft Excel
Home Insert Page Layout Formulas Data Review View Developer Solver Foundation
Method: Monte Carlo Count: Automatic Seed: Automatic
A1
Total Time (ms):120785
Solve Completion Status: Optimal
Solver Selected: Simplex
Algorithm: Primal
Arithmetic: Hybrid
Variables: 1525 -> 1525 + 1503
Rows: 1505 -> 1505
Nonzeros: 9642
Eliminated Slack Variables: 2
Pricing (exact): SteepestEdge
Pricing (double): SteepestEdge
Basis: Slack
Pivot Count: 1510
Phase 1 Pivots: 1502 + 0
Phase 2 Pivots: 8 + 0
Factorings: 34 + 1
Degenerate Pivots: 0 (0.00 %)
Branches: 0
Stochastic Measures:
Stochastic Solution Type: Sampled
Sampling Method: MonteCarlo
Sample Count: 300
Random Seed: 123456
Solving Method: Deterministic Equivalent
EV: 3812.29780564263
VSS: Infinity
EVPI: 4278.42861086424
Goals:
g: 4278.42861086424
Modeling Pane
Input Output Goals Constraints Directives Model Log
Clear Log
Solving Method: Deterministic Equivalent
EV: 3812.29780564263
VSS: Infinity
EVPI: 4278.42861086424
===Solution Details===
Goals:
g: 4278.42861086424
Decisions:
Allocation(0, 0): 10
Allocation(0, 1): 0
Allocation(0, 2): 0
Allocation(0, 3): 0
Allocation(0, 4): 0
Allocation(1, 0): 0
Allocation(1, 1): 10.841885572346
Allocation(1, 2): 1.20599297628023
Allocation(1, 3): 5.66406808899319
Allocation(1, 4): 1.28805336238061
Allocation(2, 0): 7.35469423247389
Allocation(2, 1): 0
Allocation(2, 2): 0
Allocation(2, 3): 0
Allocation(2, 4): 17.6453057675261
Allocation(3, 0): 8.17746700371765
Allocation(3, 1): 0
Allocation(3, 2): 6.82253299628235
Allocation(3, 3): 0
Allocation(3, 4): 0
Allocation(4, 0): 0
Allocation(4, 1): 0
Allocation(4, 2): 0
Allocation(4, 3): 0
Allocation(4, 4): 0
Second stage decisions (Average [Min, Max]):
BumpedRecourse(0): 16.4884785341902 [0, 33.8159125191923]
BumpedRecourse(1): 12.0257602556621 [0, 24.2818999080181]
BumpedRecourse(2): 12.8855206971485 [0, 25.7190658950648]
BumpedRecourse(3): 5.15101956207677 [0, 11.1784738076414]
BumpedRecourse(4): 14.9025407131396 [0, 25.6820035428231]
Solver Foundation Summary
Summary Complete 100%

Descriptive and Prescriptive Models

Businesses require descriptive and prescriptive models to operate efficiently and strategically. One complements the other. Solver Foundation models are designed to be prescriptive. The majority of IT modeling systems to date, Unified Method being one example, are focused on describing the structure, composition and interaction of software systems. These are descriptive models, whether describing activity flows, web service interface contracts, composition of software modules through dependency graphs, or specializations of behavior in class diagrams. These are all vital to proper understanding of what a system is doing and how it is composed. Prescriptive models provide complementary and vital information – how systems should be configured to deliver on business constraints (e.g. "To make sure you can finish manufacturing the cars by end of month, set the main assembly line work buffers to <x>"), optimal values for parameters to extract desired results, etc. Stochastic modeling, trying to understand the future likely state of a system given current constraints, variables, and parameters is also a good example of using prescriptive models. Here, understanding the likely evolution of a system in time (forecasting the outcome), can help planners and decision makers make better decisions about allocating current resources (people, goods, money). Pricing models for currency, as used in arbitrage and derivatives trading, are examples of prescriptive models – in this case changing in **real-time** to reflect the trading price of currencies and derivatives in their respective markets. Many large-scale operational models are developed to permit continuous planning to deal with fluctuating business and market conditions. Solver Foundation is admittedly more focused on prescriptive models and infrastructure. This is well aligned with the prescriptive nature of business planning and strategic decision making. Solver Foundation models should, however, be easy to correlate to and interoperate with descriptive models produced by tools and systems downstream from the planning and optimization activities. Technologies like OML can play role in the extraction and sharing of model structures and solutions between Microsoft Office and Business Intelligence systems and backend enterprise systems.

Phases of Business Enterprise Modeling and Planning

Modeling and decision making take place in the dynamic context of real business problems. Before any Operations Research (OR) can be done, valid data (experience) must be harvested. The majority of OR work occurs within enterprises that have processes in place and are interested in tuning and making then competitive. Although planning is defined below as a phase one activity we acknowledge that the majority of these phase one activities presuppose real world experience and data collection about how the business under consideration is actually doing. Bad input data cannot be fixed by a model as models are data quality dependent. This highlights the interrelated, 360-degree nature of data and processes across all four of the pillars required for a complete BPMS Platform. Conceptually, modeling is best thought of as a closed-loop, three phase system. This is an intentional abstraction over the dozens of different classes and techniques of modeling in some use within businesses and IT departments globally. The phases include:

- Phase One: Business Planning, Modeling, and Simulation (What Should We Do? What If We Try This? How Important is <x> To the Outcome?)
- Phase Two: Business Technology Design and Implementation (How Should We Build It? Where Should We Deploy? How Much Do We Need?)
- Phase Three: Business Operations, Management and Tuning (What's Changed? Are We Meeting Expectations? What Should We Alter? What Did We Learn?)

Now, repeat Phase-One with input from Phase-Three…

This is ideally a closed loop. Feedback can be incorporated at the next “turn of the crank”. The tragedy of modern IT is that it largely forgets about the artifacts and processes involved in phase-one and instead concentrates on the phase two and three technologies. Phase-one is seen as a strictly exogenous to the “real” processes being constructed by IT. This is the classic trap that an IT perspective brings to the table. Again, the two-worlds of business and IT see different landscapes. Often, the solutions developed by short-circuiting phase-one get trapped in a local minimum or maximum. We view the entire loop of modeling phases as the 360-Degree Platform for business planning and optimization.

We hope you enjoyed this introduction to Solver Foundation, and recommend that you try the technology for yourselves. For more information, please visit www.solverfoundation.com, or for a free Express edition, please visit our community site at http://code.msdn.microsoft.com/solverfoundation.

-Solver Foundation Team