The Optimization Services Project on COIN-OR

Robert Fourer, Jun Ma

Industrial Engineering & Management Sciences
Northwestern University

`[4er,maj]@iems.northwestern.edu`

Kipp Martin

Graduate School of Business
University of Chicago

`kmartin@gsb.uchicago.edu`

IFORS 2008

Johannesberg, South Africa — 13-18 July 2008 — MC17.3

“Optimization Services” (OS)

A framework for providing optimization tools

- XML-based
- Service-oriented
- Distributed
- Decentralized

A project for implementing such a framework

- Straightforward and ubiquitous access
- Powerful solvers

Using a robust service-oriented architecture

- Linking modeling languages,
  solvers, schedulers, data repositories
- Residing on different machines, in different locations,
  using different operating systems.

OS on the Internet

Home site: `www.optimizationservices.org`

- Overview, standards, publications, presentations, FAQs
- Contact information, downloads, licenses

Developer site: `www.coin-or.org/projects/OS.xml`

- Login, register, wiki, source repository, timeline, search

Newsgroup:

`groups.google.com/group/optimizationservices`

COIN mailing list:

`list.coin-or.org/mailman/listinfo/os`

*. . . **newsgroup and COIN mailing list***
are automatically cross-posted

OS Licenses, etc.

Written in multiple languages

- C/C++
- Java
- .NET

Released as open source code

- Under the Common Public License (“CPL”)

Available as a COIN-OR project

- Released this year
- More solvers being added
  - ∗ Bonmin most recently

OS Builds: Platforms

Unix

- Mac
- Linux

Windows

- Windows (MS Visual Studio)
- Cygwin (gcc)
- MSYS (gcc, cl.exe)

OS Builds: Integration

Core (OSCommon library)

Modeler side

- AMPL / `.nl`
- LINGO, What's Best (planned)
- MATLAB

Solver side

- COIN OSI
- AMPL/ASL
- Linear: CLP, CBC, CPLEX, Impact
- Nonlinear: IPOPT, LINDO, KNITRO, Bonmin
- CppAD (automatic differentiation)

. . . some still unstable

. . . looking for developers to provide others

OS Downloads

OSxL XML schemas (OSRepresentation library)

OSxL WSDL files (OSCommunication library)

. . . in a zipped file or individually

OS Downloads *(cont'd)*

Sources and builds on common platforms

- C/C++
  - readers/writers
  - client agent for contacting remote services
  - interfaces to solvers and modeling systems
  - automatic differentiation, etc.
- Java (to be put up)
  - same features as C/C++, plus
    Web Services, server, distributed systems.
- .NET (C#) (to be put up)
  - similar to Java but not as complete

OS Repository

Linear (netlib basic, infeasible, Kennington)

- Individual XML (**OSiL** format) files available now
- Zip files to come

Mixed integer (mainly from miplib 2003)

Nonlinear

- CUTE now, more to come

Stochastic

- Thanks to Gus Gassmann

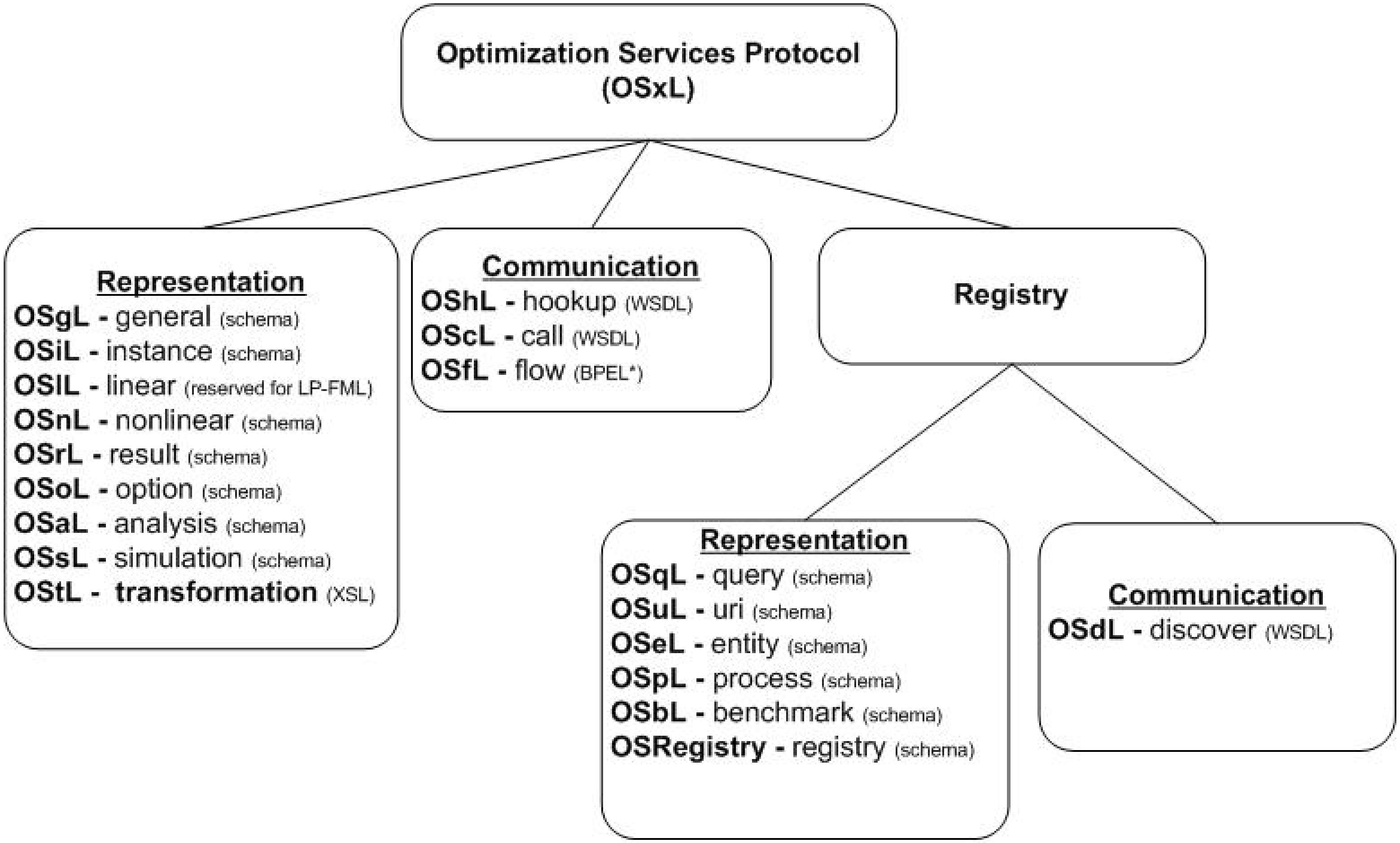
. . . all known documentation
(source, solution, description, type, etc.)

Standards

OS framework provides standards in 3 areas

- ➢ Optimization instance representation
- ➢ Optimization communication
  - ∗ accessing
  - ∗ interfacing
  - ∗ orchestration
- ➢ Optimization service registration and discovery

Standards



Optimization Services Protocol (OSxL)

Representation

OSgL - general (schema)
OSiL - instance (schema)
OSlL - linear (reserved for LP-FML)
OSnL - nonlinear (schema)
OSrL - result (schema)
OSoL - option (schema)
OSaL - analysis (schema)
OSsL - simulation (schema)
OStL - transformation (XSL)

Communication

OShL - hookup (WSDL)
OScL - call (WSDL)
OSfL - flow (BPEL*)

Registry

Representation

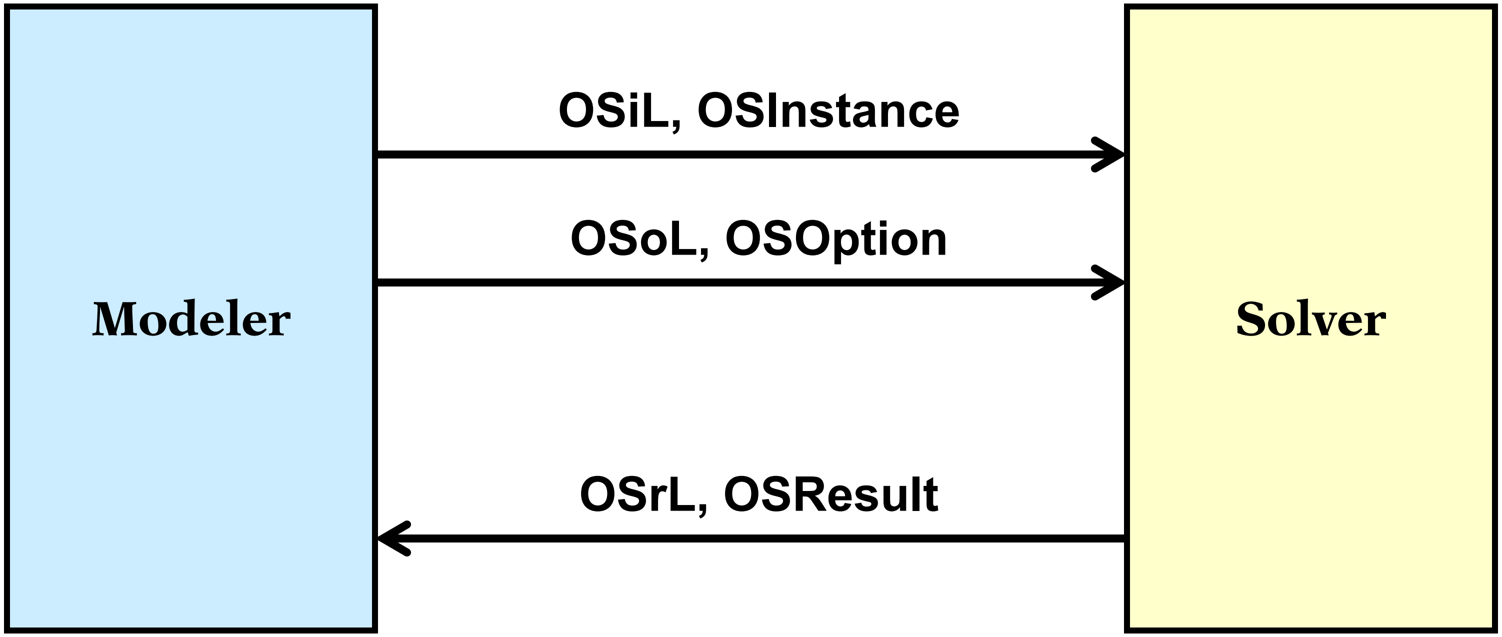
OSqL - query (schema)
OSuL - uri (schema)
OSeL - entity (schema)
OSpL - process (schema)
OSbL - benchmark (schema)
OSRegistry - registry (schema)

Communication

OSdL - discover (WSDL)

***OSmL:** a modeling language and NOT an Optimization Services Protocol
***Letters not currently used:** w, z
***BPEL:** Business Process Execution Language for flow orchestration.

Quick Overview



XML text files

- OSiL, OSoL, OSrL

In-memory data structures

- OSInstance, OSOption, OSResult

Motivation

XML Means “Tagged” Text Files . . .

Example: html for a popular home page

```
<html><head><meta http-equiv="content-type" content="text/html;
charset=UTF-8"><title>Google</title><style><!--
body,td,a,p,.h{font-family:arial,sans-serif;}
.h{font-size: 20px;}
.q{text-decoration:none; color:#0000cc;}
//-->
</style>
</head><body bgcolor=#ffffff text=#000000 link=#0000cc
vlink=#551a8b alink=#ff0000 onLoad=sf()><center><table border=0
cellspacing=0 cellpadding=0><tr><td><img src="/images/logo.gif"
width=276 height=110 alt="Google"></td></tr></table><br>
.......
<font size=-2>&copy;2003 Google - Searching 3,307,998,701 web
pages</font></p></center></body></html>
```

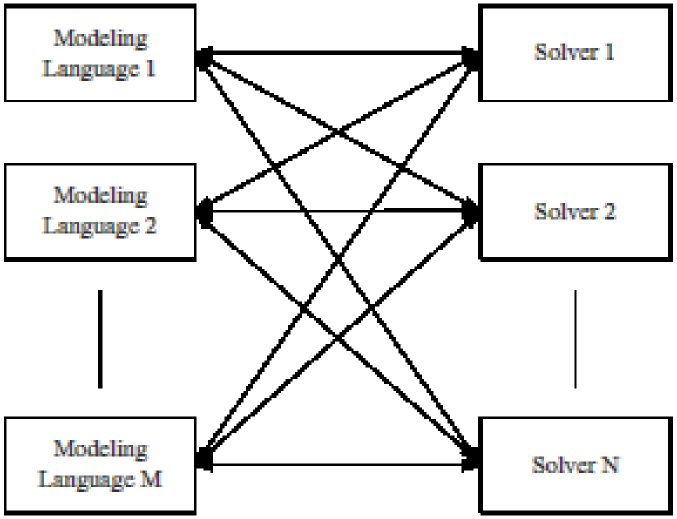
. . . a collection of XML tags is designed for a special purpose

. . . by use of a schema written itself in XML

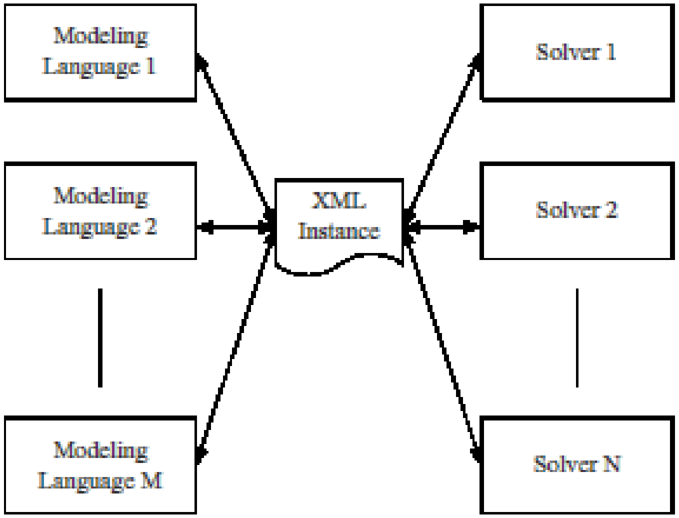
Motivation

Advantage of any standard

MN drivers without a



M + N drivers with a standard



Advantages of an XML Standard

Specifying it

- Unambiguous definition via a *schema*
- Provision for *keys* and *data typing*
- Well-defined expansion to new *name spaces*

Working with it

- Parsing and validation via standard *utilities*
- Amenability to *compression* and *encryption*
- Transformation and display via XSLT *style sheets*
- Compatibility with *web services*

OSiL: Optimization Problem Instances

Design goals

- Simple, clean, extensible, object-oriented

Standard problem types supported

- Linear
- Quadratic
- General nonlinear
- Mixed integer for any of above
- Multiple objective for any of above
- Complementarity

OSiL *(cont'd)*

Extensions (stable or near-stable)

- User-defined functions
- XML data (within the OSiL or remotely located)
- Data lookup (via XPath)
- Logical/combinatorial expressions and constraints
- Simulations (black-box functions)

OSiL *(cont'd)*

Prototypes

- Cone & semidefinite programming
- Stochastic
  - recourse, penalty-based, scenario (implicit or explicit)
  - risk measure/chance constrained
  - major univariate, multivariate, user-defined distributions
  - general linear transformation and ARMA processes
  - R. Fourer, H.I. Gassmann, J. Ma, and R.K. Martin,
    "An XML-Based Schema for Stochastic Programs."
    Forthcoming in *Annals of Operations Research*.

Text files

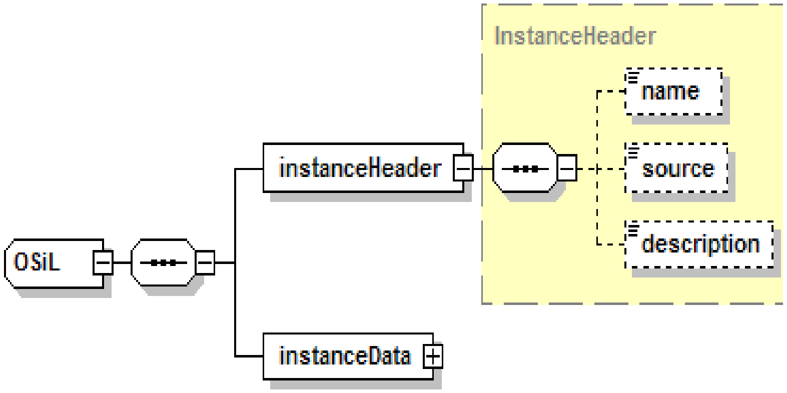
Text from the OSiL Schema

```
<xs:complexType name="Variables">
  <xs:sequence>
    <xs:element name="var" type="Variable" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="number" type="xs:positiveInteger" use="required"/>
</xs:complexType>
```

```
<xs:complexType name="Variable">
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="init" type="xs:string" use="optional"/>
  <xs:attribute name="type" use="optional" default="C">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="C"/>
      <xs:enumeration value="B"/>
      <xs:enumeration value="I"/>
      <xs:enumeration value="S"/>
    </xs:restriction>
  </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="lb" type="xs:double" use="optional" default="0"/>
  <xs:attribute name="ub" type="xs:double" use="optional" default="INF"/>
</xs:complexType>
```

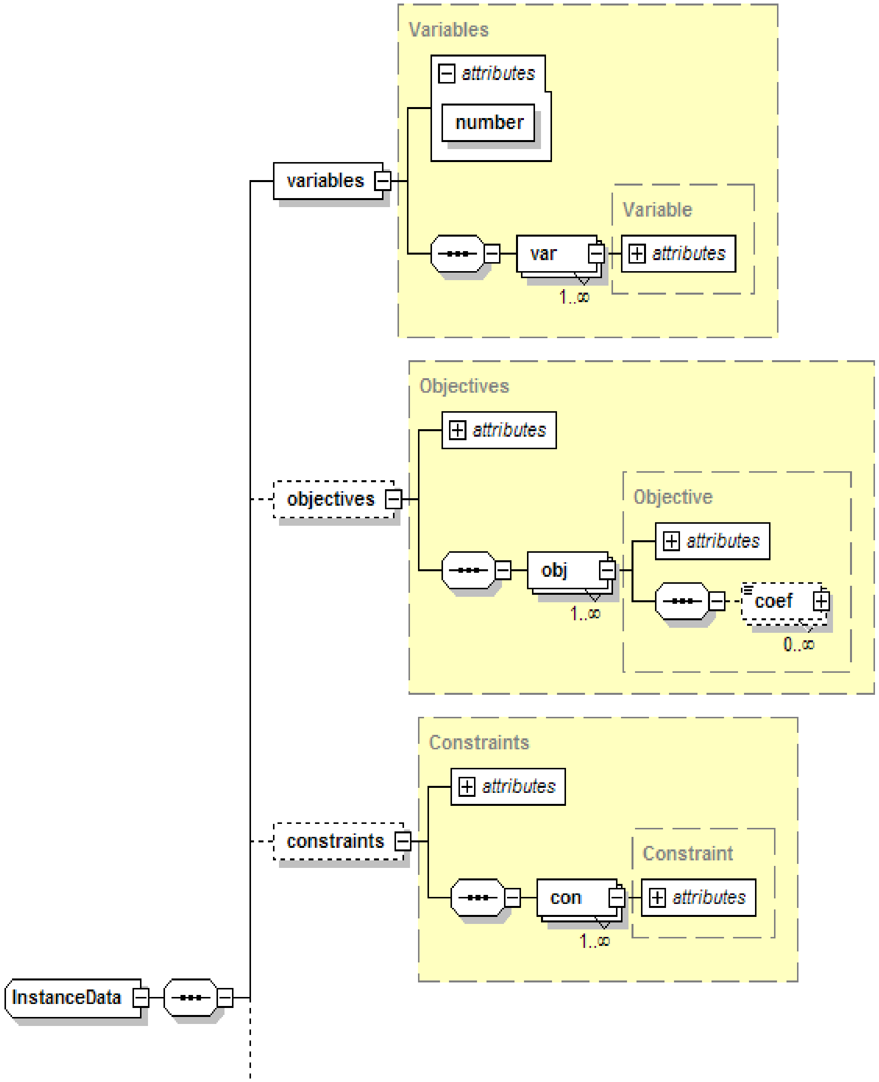
Text files

Diagram of the OSiL Schema



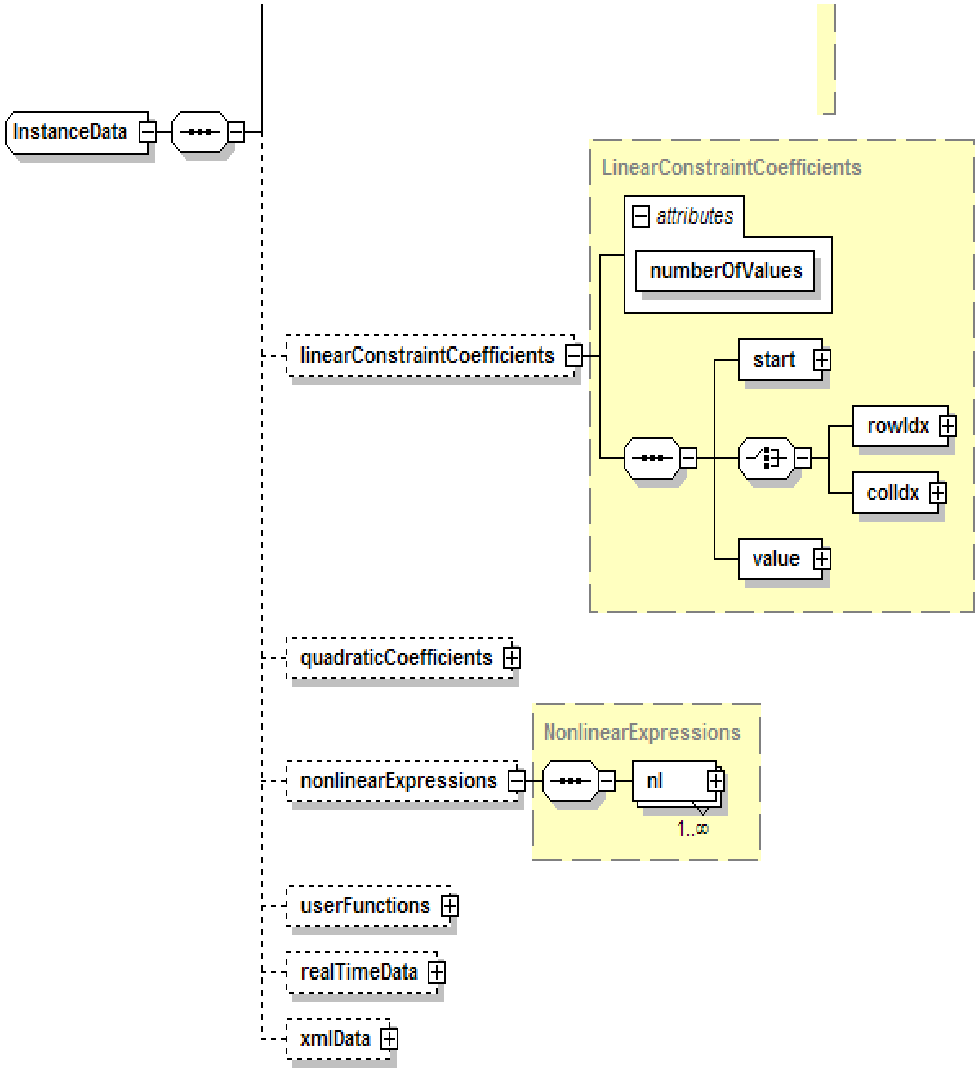
Text files

Details of OSiL's instanceData Element



Text files

Details of OSiL's instanceData Element



Text files

Example: A Problem Instance (in AMPL)

```
ampl: expand _var;

Coefficients of x[0]:
        Con1  1 + nonlinear
        Con2  7 + nonlinear
        Obj   0 + nonlinear

Coefficients of x[1]:
        Con1  0 + nonlinear
        Con2  5 + nonlinear
        Obj   9 + nonlinear

ampl: expand _obj;

minimize Obj:
        (1 - x[0])^2 + 100*(x[1] - x[0]^2)^2 + 9*x[1];

ampl: expand _con;

subject to Con1:
        10*x[0]^2 + 11*x[1]^2 + 3*x[0]*x[1] + x[0] <= 10;

subject to Con2:
        log(x[0]*x[1]) + 7*x[0] + 5*x[1] >= 10;
```

Text files

Example in OSiL

```
<instanceHeader>
   <name>Modified Rosenbrock</name>
   <source>Computing Journal3:175-184, 1960</source>
   <description>Rosenbrock problem with constraints</description>
</instanceHeader>

<variables number="2">
   <var lb="0" name="x0" type="C"/>
   <var lb="0" name="x1" type="C"/>
</variables>

<objectives number="1">
   <obj maxOrMin="min" name="minCost" numberOfObjCoef="1">
      <coef idx="1">9</coef>
   </obj>
</objectives>

<constraints number="2">
   <con ub="10.0"/>
   <con lb="10.0"/>
</constraints>
```

Text files

Example in OSiL *(continued)*

```
<linearConstraintCoefficients numberOfValues="3">
   <start>
      <el>0</el>
      <el>1</el>
      <el>3</el>
   </start>
   <rowIdx>
      <el>0</el>
      <el>1</el>
      <el>1</el>
   </rowIdx>
   <value>
      <el>1.0</el>
      <el>7.0</el>
      <el>5.0</el>
   </value>
</linearConstraintCoefficients>

<quadraticCoefficients numberOfQPTerms="3">
   <qpTerm idx="0" idxOne="0" idxTwo="0" coef="10"/>
   <qpTerm idx="0" idxOne="1" idxTwo="1" coef="11"/>
   <qpTerm idx="0" idxOne="0" idxTwo="1" coef="3"/>
</quadraticCoefficients>
```

Text files

Example in OSiL *(continued)*

```
<nl idx="-1">
   <plus>
      <power>
         <minus>
            <number type="real" value="1.0"/>
            <variable coef="1.0" idx="1"/>
         </minus>
         <number type="real" value="2.0"/>
      </power>
      <times>
         <power>
            <minus>
               <variable coef="1.0" idx="0"/>
               <power>
                  <variable coef="1.0" idx="1"/>
                  <number type="real" value="2.0"/>
               </power>
            </minus>
            <number type="real" value="2.0"/>
         </power>
         <number type="real" value="100"/>
      </times>
   </plus>
</nl>
```

Text files

Example in OSiL *(continued)*

```
<nl idx="1">
   <ln>
      <times>
         <variable idx="0"/>
         <variable idx="1"/>
      </times>
   </ln>
</nl>
```

OSrL: Optimization Problem Results

Counterpart to OSiL for solver output

- General results such as serviceURI, serviceName, instanceName, jobID, time
- Results related to the solution such as status (unbounded, globallyOptimal, etc.), substatus, message
- Results related to variables (activities), objectives (optimal levels), constraints (dual values)
- Service statistics such as currentState, availableDiskspace, availableMemory, currentJobCount, totalJobsSoFar, timeLastJobEnded, etc.
- Results related to individual jobs including state (waiting, running, killed, finished), userName, submitTime, startTime, endTime, duration, dependencies, scheduledStartTime, requiredDirectoriesAndFiles.

OSrL *(cont'd)*

Additional solution support

- Support for non-numeric solutions such as those returned from combinatorial or constraint programming solvers
- Support for multiple objectives
- Support for multiple solutions
- Integration of analysis results collected by the solver

OSoL: Optimization Options

Counterpart to OSiL for solver instructions

- General options including serviceURI, serviceName, instanceName, instanceLocation, jobID, license, userName, password, contact
- System options including minDiskSpace, minMemorySize, minCPUSpeed
- Service options including service type
- Job options including scheduledStartTime, dependencies. requiredDirectoriesAndFiles, directoriesToMake, directoriesToDelete, filesToCreate, filesToDelete, processesToKill, inputFilesToCopyFrom, inputFilesToCopyTo, etc.

Limited standardization of algorithmic options

- Currently only initial values

OSoL *(cont'd)*

Including support for:

- Various networking communication mechanisms
- Asynchronous communication (such as specifying an email address for notification at completion)
- Stateful communication (achieved mainly through the built-in mechanism of associating a network request with a unique jobID)
- Security such as authentication and licensing
- Retrieving separately uploaded information (when passing a large file as a string argument is inefficient)
- Extended or customized solver-specific or algorithm-specific options

Other XML Schema-Based Standards

Kept by the OS registry

- OSeL (entity, experimental): static information on optimization services (such as type, developer)
- OSpL (process, near stable): dynamic information on optimization services (such as jobs being solved)
- OSbL (benchmark, experimental): benchmark information on optimization services

For use by the discovery process

- OSqL (query, experimental): specification of the query format used to discover the optimization services in the OS registry
- OSuL (uri/url, experimental): specification of the discovery result (in uri or url) sent back by the OS registry

Other Schema-Based Standards *(cont'd)*

Formats and definitions

- OSsL (simulation, stable): format for input and output used by simulation services invoked via the Optimization Services to obtain function values
- OSgL (general, near stable): definitions of general elements and data types used by other OSxL schemas. Usually included in the beginning of another OSxL schema through the statement:
  `<xs:include schemaLocation="OSgL.xsd"/>`
- OSnL (nonlinear, stable): definitions (operators, operands, etc.) of the nonlinear, combinatorial, and other nodes used in other OSxL's, mainly OSiL

Other WSDL-Based Standards

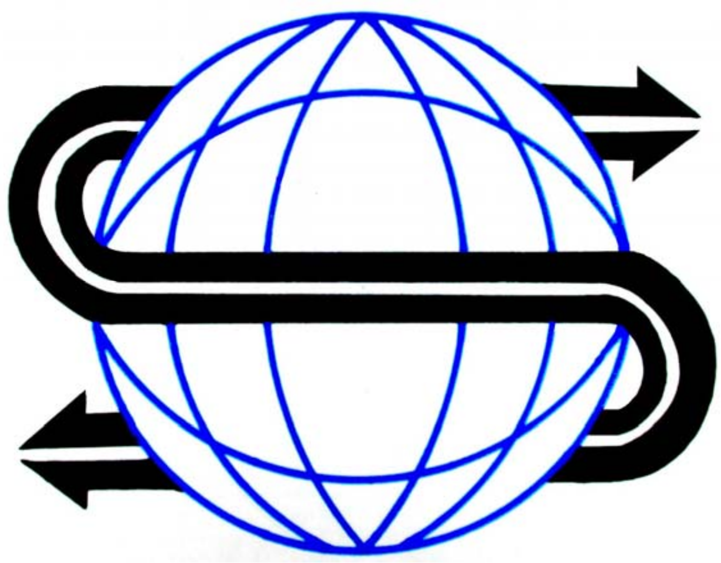
WSDL

- Web Service Definition Language

WSDLs for OS (stable)

- OShL (hook): for invoking solver/analyzer services
- OSdL (discover): for invoking optimization registry services to register and discover services
- OScL (call) for invoking simulation services, usually to obtain function values.

www.optimizationservices.org . . .



. . . Questions?