OPTIMIZATION SERVICES CONSTITUTION

HORAND GASSMANN, JUN MA, AND KIPP MARTIN

1. Separation of functionality. An instance representation should be just that, an instance representation. It should not be convoluted with solver options, solver results, etc.
2. Handle sparsity. Most real-world (especially linear programs) are very sparse. The instance format should be able to represent problems in a sparse format.
3. Extensible – we should be able to extend the core to new optimization types and remain backward compatible.
4. Be parser friendly – the model instance should be easy to parse. This is an advantage of XML, having both tags and data makes for easy parsing.
5. Fit with modern information technology infrastructures – this is an advantage of using XML, it is consistent with how data is being represented in middleware software
6. Have a natural in-memory representation – I am not sure how to word this properly. What I am driving at here are the rules we having governing how our OSiL, OSrL, and OSoL maps into in-memory objects. Each part of the OSiL *must* be mapped into an in-memory object following our rules.
7. Self contained with no ambiguity. The model instance should be completely defined and there should be no room for ambiguity. For example, in MPS format a max or min is not specified. This must be avoided.
8. Parsimonious. The representation should be as compact as possible while fully representing the instance in an unambiguous manner. There should be no unnecessary redundancy. For example, we should avoid the MPS feature that a column name is repeated for each nonzero in a row.
9. The design philosophy of OSiL is to honor the original model constructs as closely as possible, but at the instance level. It is critical to allow the solver to "internally" work with a condition specified at the model level rather than "forcing" the solver to deal reformulation of the condition. Examples of this include special ordered sets, and semi continuous variables.
10. The representation method should be conducive to error checking. This is why we define a schema and use XML as it facilitates error checking.

Date: May 14, 2012.