ORML: Optimization Reporting Markup Language

Obi C. Ezechukwu
Department of Computing
Imperial College London
Exhibition Road
London SW7 2BZ
United Kingdom

mailto:oce@doc.ic.ac.uk

Istvan Maros
Department of Computing
Imperial College London
Exhibition Road
London SW7 2BZ
United Kingdom

mailto:im@doc.ic.ac.uk

Departmental Technical Report 2003/13

ISSN 1469-4174

November 2003

Abstract

This paper presents a reporting mark-up language that enables the encoding of optimization reporting data in XML format. The language eases the process of integrating optimization software into decision support applications by shielding upstream applications from native representation formats. It also serves as an enabling technology for web services based optimization due to the fact that it provides a response encoding mechanism which is compatible with Internet standards, protocols and technologies.

Keywords: *XML, XSD, Optimization, Modelling, Web Services, Reporting*

It is often necessary to integrate optimization software into decision support applications, however due to the proprietary nature of the interfaces to, and representation formats used by such software, tight coupling is introduced in the process. Whereas this may not be a major issue in the case where the necessary pieces of software reside on the same machine, or local network, and are compatible at both the programming language and OS levels, it presents a challenge in distributed applications, or where incompatibilities are introduced by different programming language versions or hardware platforms.

A lot of research has been directed at resolving the issues relating to representation formats for optimization models, spurring technological advances such as the emergence of algebraic modelling systems. However little effort has been directed towards standardising solution output or reporting, largely because the problem is most relevant in web services environments. In standalone mode, algebraic systems deal with the task of interfacing to solution algorithms, thus shielding the end user from low-level I/O formats used by such algorithms. In the context of decision support applications, bridges can be implemented to handle the details of interfacing to optimization software. Whereas this work has to be repeated for every DSS application implementation and cumulatively is not a very efficient use of time, it is none the less possible.

However, there has been a lot of recent interest from members of the OR community in Internet computing technologies and architecture, with the particular aim of harnessing the strengths of the Internet for the benefit of the OR community, specifically the distribution of optimization software and functionality via the Internet. A number of projects have been initiated with the specific aim of achieving or facilitating this [Bhargava and Krishnan (1998), Fourer and Goux (2001)], all with varying degrees of success. Of particular interest is the possibility of distributing optimization functionality as Internet *'services'* using a web services methodology [Kristjansson (2001)].

The use of web services based architecture and technology for accessing, distributing and implementing optimization software would also have a very big impact on how decision support systems are implemented. Similar web services based approaches could be used for internal i.e. inter organisation implementations, as for the wider Internet community. Internet i.e. web services technology can be used alleviate problems associated with hardware, programming language, and operating system versions.

One of the basic requirements of the web services architecture is the ability to encode request and response information. As such, in order to achieve a situation where optimisation functionality can be accessed and distributed as web services, a means is required to encode request and response information, thus re-introducing the problem of model representation and at the same time bringing to light the problem of representing reporting i.e. response information.

AML [Ezechukwu and Maros (2003)] provides syntax for representing models and model data based on XML [11] thus alleviating the problem of request encoding. ORML (Optimization Reporting Markup Language) is a complimentary, standalone format that can be used for encoding reporting data. It is intended as an abstraction of existing formats, and it is based on XML, and thus is usable within an Internet computing environment. This paper explores the aims of the language, and provides a brief and introductory overview of its solution-description syntax using an example model.

1 Objectives

ORML is intended primarily as a reporting format with enough generality to cover the formats used by current optimization software. Although the level and format of information reported by optimization software varies from one system to the other, there are basic similarities that can be extracted in order to create a common syntax, and this is what ORML proposes to do. It is not concerned with the level of information requested nor how it is requested, as these are the responsibility of the encapsulating interface or subroutine, but rather how the information is encoded. It is designed to be loose and flexible enough to encapsulate as little or as much data as requested by the client software or user.

The aims driving the development of the language can be summarised in the following points:

i. **Abstraction of existing formats:** The primary aim of ORML is to present a common view of reporting information regardless of the underlying optimization software. It is also designed to be flexible enough to allow the inclusion of system and vendor specific data in reports without compromising the structure or integrity of the information. This could take the form of 'solver status' or 'solution status' information e.g. the number of iterations to reach a solution.
ii. **Vendor and Platform Independence:** ORML is not part of a modelling system nor does it require explicit support from any modelling system or vendor. It is also portable across hardware and operating system platforms, and programming languages.
iii. **Decoupling senders and receivers:** ORML breaks the coupling between client systems and optimization software caused by proprietary and non-portable formats. It essentially decouples the sender of the information from its receiver. As such, generic routines can be used in the client software for manipulating solution data regardless of the source of the data.
iv. **Compact syntax:** The ORML syntax is designed to be compact and easy to comprehend. It is also designed to provide an intuitive mapping to the matrix structure of optimization model instances and solution data, thus alleviating the task of learning the syntax.
v. **Extensibility:** By building on XML, ORML allows easy extension of its syntax. This is particularly useful if a user/organisation wants to extend ORML to map directly onto a specific modelling system, e.g. by the inclusion of additional fields in the reporting data.
vi. **Ease of use:** The abundance of XML processing API implies that it is reasonably cheap and easy to manipulate ORML format reports.

2 Example

This section presents an example model, which is a formulation of a simple multi-period production-planning problem. The objective of the model is to maximise the net revenue or profit from the production of a number of items, subject to meeting the demands of customers, and not exceeding the production capacity available.

This example is provided in order to illustrate the structure of an ORML solution report, i.e. to provide basic examples of majority of the types of elements that may occur in an ORML report. For the sake of brevity, the model is illustrated in conceptual form and the equivalent algebraic modelling language formulation is provided in MPL [8] syntax. The latter is actually generated by applying an XSL [13] stylesheet to AML model and data documents, however given the verbose nature of AML, and space restrictions, only the result of the translation is shown.

The conceptual model is provided in figure 2.1.The formulation contains three types of decision variables that can be explained as follows:

- *Store:* This is the amount of each product that is left in inventory at the end of each production period i.e. unsold stock
- *Sell:* The quantity of each product that is sold in each period
- *Produce:* The quantity of each product produced in a given time period

The model parameters i.e. data elements are as follows:

- *Price:* Sale price of each product
- *Demand:* Periodic product demand
- *ProductionCost:* The production cost per product
- *ProductionRate:* Production rate per product
- *WorkingDays:* The number of working days, or productive days in each period
- *StorageCost:* Per product storage cost
- *StorageCapacity:* Fixed storage capacity

It is obvious from the above statements that there are two sets, namely:

- *Product:* The set of products
- *Period:* The set of time periods

$$\textbf{MAX}\ GrossRevenues - TotalCosts \tag{1}$$

$$\textbf{Subject to:}\ \sum_i (Produce_{i,j} / ProductionRate_i) \le WorkingDays_j \tag{2}$$

$$\sum_{j,i} Store_{i,j} \le StorageCapacity \tag{3}$$

$$Produce_{i,j} + Store_{i,j-1} = Sell_{i,j} + Store_{i,j} \tag{4}$$

$$Sell_{i,j} \le Demand_{i,j} \tag{5}$$

$$i \in products, j \in period \tag{6}$$

$$GrossRevenues := \sum_{i,j} Price_i Sell_{i,j} \tag{7}$$

$$TotalProductionCosts := \sum_{i,j} ProductionCost_i Produce_{i,j} \tag{8}$$

$$TotalStorageCosts := \sum_{i,j} StorageCost_i Store_{i,j} \tag{9}$$

$$TotalCosts := TotalProductionCosts + TotalStorageCosts \tag{10}$$

Figure 2.1: Production Planning Model

The corresponding ORML solution document is provided in appendix A, and the equivalent MPL representation of the model is given by the listing:

```
  { Basic production planning model.  }
 TITLE
  ProductionPlanning;

 INDEX
  clothing  := (Trousers, Shirts);
  accessories  := (Socks);
  product  := (Trousers,Shirts,Socks);
  period  := (1, 2, 3, 4);

 DATA
  Price[product] := (39.50, 35.00, 5.99);
  Demand[product,period] := [Trousers, 1, 1300, Trousers, 2, 800, Trousers, 3, 6000, Trousers, 4, 3000,
        Shirts, 1, 1750, Shirts, 2, 1300, Shirts, 3, 4000, Shirts, 4, 8000, Socks, 1, 4000, Socks, 2, 3000,
        Socks, 3, 9000, Socks, 4, 15000];
  ProductionCost[product] := (20.00, 21.00, 6.00);
  ProductionRate[product] := (10, 9, 30);
  WorkingDays[period] := (20, 22, 20, 19);
  StorageCost[product] := (0.50, 0.50, 0.01);
  StorageCapacity := 800;

 DECISION VARIABLES
  Produce[product,period];
  Store[product,period];
  Sell[product,period];

 MACROS
  GrossRevenues = SUM(product,period:Price[product] * Sell[product,period]);
  TotalProductionCosts = SUM(product,period:ProductionCost[product] * Produce[product,period]);

  TotalStorageCosts = SUM(product,period:StorageCost[product] * Store[product,period]);
```

```
  TotalCosts = TotalProductionCosts + TotalStorageCosts;

MODEL
  MAX Profit = GrossRevenues - TotalCosts;

SUBJECT TO
  ProductionCapacity[period] : SUM(product:Produce[product,period] / ProductionRate[product]) <=
                                    WorkingDays[period];
  InventoryBalance[product,period] : Produce[product,period] + Store[product,period-1] =
                                    Sell[product,period] + Store[product,period];
  MaxStorageCapacity[period] : SUM(product:Store[product,period]) <= StorageCapacity;
  LimitSupply[product,period] : Sell[product,period] <= Demand[product,period];

END
```

In the implementation presented above, the value of the starting inventory is omitted because, the MPL modelling system assigns a default value of zero to it. However it is worth mentioning that this behaviour is not common among all modelling systems, and should not be assumed as the norm.

3 Syntax Description

The ORML solution file consists of four main nodes/elements namely: ***"<solutionSummary>"***, ***"<variablesSolutionReport>"***, ***"<constraintsSolutionReport>"***, and ***"<solveStatistics>"***. The *"<solutionSummary>"* node holds summary information about the solution. The *"<variablesSolutionReport>"* and *"<constraintsSolutionReport>"* nodes hold information on the post-solution values of variables and constraints. These two nodes are only applicable in cases where a solution is obtained for the model. The *"<solveStatistics>"* node is a loosely formatted node that allows solvers or optimization systems to provide additional information on the solution or cause of infeasibility. Most elements of the solution report are optional, thus allowing the calling application, interface or user to dictate the level of information that is included in the report. This however places most of the burden of content validation on the client application i.e. ensuring that the requested information is indeed included in the report.

The ORML solution document begins with the following node:

```
<orml:solutionReport modelId="ProductionPlanning" modelInstanceId="Illustration"
                        xmlns:orml="http://www.doc.ic.ac.uk/~oce/elsinore/2003/ORML">
```

The *'modelId'* and *'modelInstanceId'* attributes respectively provide a unique identifier for the model and the particular instance that the report relates. Both attributes are optional and the latter is more applicable to situations where the model is solved repeatedly with different data sets, e.g. when performing scenario analysis.

3.1 Solution Summary

The *"<solutionSummary>"* element as its name implies provides a brief summary of the solution. It is the only mandatory solution document element, simply because without it, there would be no means of ascertaining whether or not a solution was obtained, or the nature of the solution. It consists of two attributes: *'objectiveValue'* and *'solutionStatus'*. The *'objectiveValue'* attribute is optional because it is not applicable to constraint satisfaction problems or infeasible models. In the case where a model is infeasible, additional information on the cause(s) or nature of the infeasibility may be provided in the *'<solveStatistics>'* element which is described in section 3.4. The *'solutionStatus'* attribute provides information on the nature of the solution in the case where one is found, or indicates that the problem is not feasible. It can take any one of the following four values: STATUS_UNKNOWN, GLOBAL_OPTIMUM_FOUND, LOCAL_OPTIMUM_FOUND, and PROBLEM_INFEASIBLE. Although the collective range of states reported by various modelling systems is wider, it is necessary to limit the reported states to these four in order to ensure consistent support across all systems.

However, where required an optimization system may also report additional status information in the '*<solveStatistics>*' element.

The solution summary element for the given example is as follows:

<solutionSummary objectiveValue="15795.0" solutionStatus="GLOBAL_OPTIMUM_FOUND"/>

3.2 Variable Values Report

Information on variable values are held under the node "*<variablesSolutionReport>*", and values for each declared variable are encapsulated by the tag "*<variableValues>*" where the attribute '*variableId*' contains the identifier of the variable to which the values are related. The following listing provides the values of the "*Store*" variable from the example model:

```
<variableValues variableId="Store">
    <variableValue value="0" reducedCost="0">
        <subscript matrixLocation="1">Trousers</subscript>
        <subscript matrixLocation="1">1</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="0">
        <subscript matrixLocation="1">Trousers</subscript>
        <subscript matrixLocation="2">2</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="0">
        <subscript matrixLocation="1">Trousers</subscript>
        <subscript matrixLocation="3">3</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="-40">
        <subscript matrixLocation="1">Trousers</subscript>
        <subscript matrixLocation="4">4</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="-8">
        <subscript matrixLocation="2">Shirts</subscript>
        <subscript matrixLocation="1">1</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="0">
        <subscript matrixLocation="2">Shirts</subscript>
        <subscript matrixLocation="2">2</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="0">
        <subscript matrixLocation="2">Shirts</subscript>
        <subscript matrixLocation="3">3</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="-35">
        <subscript matrixLocation="2">Shirts</subscript>
        <subscript matrixLocation="4">4</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="-6">
        <subscript matrixLocation="3">Socks</subscript>
        <subscript matrixLocation="1">1</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="0">
        <subscript matrixLocation="3">Socks</subscript>
        <subscript matrixLocation="2">2</subscript>
    </variableValue>
    <variableValue value="0" reducedCost="0">
```

```
            <subscript matrixLocation="3">Socks</subscript>
            <subscript matrixLocation="3">3</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="-6">
            <subscript matrixLocation="3">Socks</subscript>
            <subscript matrixLocation="4">4</subscript>
        </variableValue>
    </variableValues>
```

Each value is encapsulated by a *"<variableValue>"* element, which has two optional attributes: *'value'* and *'reducedCost'* where the meaning of the former is self-explanatory and the latter represents the reduced cost of the variable.

In the case where a variable is indexed i.e. declared over one or more sets, and as such has more than one value, it is possible to relate each value to individual subscripts of the set(s), using the *"<subscript>"* tag. For example, the *'Store'* variable is indexed over the sets *'Product'* and *'Period'*, consequently each value of the variable relates to a unique pair of subscripts from the *'Product'* and *'Period'* sets respectively. In essence, the value of the *"<subscript>"* tag is an actual element of the set at the same index/indent in the variable declaration i.e. the i^{th} *"<subscript>"* tag is an element of the i^{th} set over which the variable is indexed. The optional *'matrixLocation'* attribute provides the co-ordinate of the value in the variable matrix. The use of the term *'matrix'* in this context does not refer to any low-level input format, but derives from the fact that an indexed variable as far as most modelling systems are concerned represents a family of variables as opposed to a single variable. Often at compilation time, the declaration is expanded across the elements of the indexing sets, thus the resulting structure is an n-dimensional structure, where 'n' is equivalent to the number of sets over which the variable is indexed. The combination of *"<subscript>"* elements obviously identifies a location within the variable matrix, and as such, no two values for any declared variable may contain the same collection of *"<subscript>"* elements.

From the solution listing, it is clear from the value of '0' assigned to the *'Store'* variables, that the punitive costs of storage should be avoided. This is because there is no constraint in the model to ensure that the demand for each product is met in all the time periods. However, the aim of the example model is to illustrate the syntax of ORML as opposed to solving a realistic or fully specified production planning problem.

3.3 Constraint Values Report

Constraint solution information is held under the *"<constraintsSolutionReport>"* top-level node. The data for each individual constraint declaration is encapsulated by the *"<constraintValues>"* tag, which has one mandatory attribute *'constraintId'* that links the solution report to the declared constraint i.e. its value is the identifier of the declared constraint. The following listing provides the solution report for the example constraint *"ProductionCapacity"*:

```
<constraintValues constraintId="ProductionCapacity">
        <constraintValue shadowPrice="0.0" activity="20.0" slackValue="0.0">
            <subscript matrixLocation="1">1</subscript>
        </constraintValue>
        <constraintValue shadowPrice="0.0" activity="22.0" slackValue="0.0">
            <subscript matrixLocation="2">2</subscript>
        </constraintValue>
        <constraintValue shadowPrice="0.0" activity="20.0" slackValue="0.0">
            <subscript matrixLocation="3">3</subscript>
        </constraintValue>
        <constraintValue shadowPrice="0.0" activity="19.0" slackValue="0.0">
            <subscript matrixLocation="4">4</subscript>
        </constraintValue>
```

</constraintValues>

Each constraint value is encapsulated by the *"<constraintValue>"* tag which has three optional attributes, namely: *'shadowPrice'*, *'activity'*, and *'slackValue'*. The meanings of the *'shadowPrice'* and *'slackValue'* attributes are self-explanatory, while the *'activity'* attribute encapsulates the value of the function which forms the right-hand side of the constraint i.e. the bounding function. In the case of the example model, all constraints excluding *'InventoryBalance'* are bounded by data vector i.e. parameter values, consequently the values of the *'activity'* attribute for these constraints are equivalent to the corresponding parameter values. For example, in the sample listing, the *'activity'* attribute values for the constraint *'ProductionCapacity'* evaluate to the values of the *'WorkingDays'* parameter, however in the case of the *'InventoryBalance'* constraint, the right hand side function consistently evaluates to '0' for the given data set, thus the *'activity'* attribute for the constraint's values are reported as '0'.

In the case where a constraint is indexed i.e. declared over one or more sets as in the given example, each *"<constraintValue>"* can be related to a subscript or element of each indexing set using the *"<subscript>"* tag. In essence, this associates each constraint value with a unique index in the constraint matrix. Once again, the term "matrix" does not refer to a low level format; rather it illustrates the manner in which modelling systems "expand" indexed constraints. In most modelling systems, when a constraint is indexed over one or more sets, the constraint merely identifies a family of related constraints rather than a single constraint. At compile time, each indexed constraint is expanded to an n-dimensional matrix structure where 'n' is the number of sets over which the constraint is indexed.

The *"<subscript>"* tags occur in the same sequence as the sets or indices in the constraint declaration i.e. the *"<subscript>"* node at position 'i' represents a valid element of the set specified as the i^{th} index of the corresponding constraint declaration. The *'matrixLocation'* attribute provides a matrix co-ordinate for the specified constraint value.

3.4 Solve Statistics

The *"<solveStatistics>"* top-level element is a relatively free-form element that allows the optimization system, solver or service to specify additional information relating to the solution or the cause(s) of infeasibility. This tag consists of sub *"<solveStatisticsEntry>"* tags, which consists of one or more *"<entryValue>"* tags. Each *"<solveStatisticsEntry>"* has an *"entryName"* attribute whose value serves as a key to the entry. In essence, the *"<solveStatisticsEntry>"* is a name, multi-value structure i.e. a name-value pair structure where each name can have multiple values.

The format and content of the *"<solveStatistics>"* element is not regulated, and as such, provides optimization systems or services the opportunity to include additional information in the solution report. This of course implies that the contents should not be relied upon by a receiving application except where published by and agreed with the vendor of the service or optimization system. A sample solve-statistics listing is given by the following:

```
<solveStatistics>
        <solveStatisticsEntry entryName="Iterations">
                <entryValue>4</entryValue>
        </solveStatisticsEntry>
</solveStatistics>
```

The above listing provides the number of iterations performed by the target solver until termination.

4 Conclusion and Further Research

This paper introduces an algebraic mark-up language that is designed primarily for the purpose of enabling distributed optimization, and aiding the integration of optimization software into decision support applications. It has demonstrated syntax for encoding response (reporting) information from

optimization systems, thus removing the dependency on proprietary and system specific formats, and reducing the coupling between client applications and optimization systems.

Future research on this language will focus on expanding the syntax to enable the encoding of pre and post optimality analysis information. The syntax will also be enhanced to enable the encoding of reporting data for additional classes of optimization models such as combinatorial and stochastic models.

5 Bibliography

1. *Optimization Reporting Markup Language*, http://www.doc.ic.ac.uk/~oce/elsinore/orml.htm
2. H. K. Bhargava and R. Krishnan, *"The World Wide Web: Opportunities for Operations Research and Management Science"*, INFORMS Journal on Computing, 10:4, pp. 359-383, 1998
3. Obi C. Ezechukwu and Istvan Maros. *"AML: Algebraic Markup Language"*, (forthcoming)
4. Obi C. Ezechukwu and Istvan Maros. *"OOF: Open Optimization Framework"*, DoC Departmental Technical Reports 2003/7, Imperial College, London, 2003
5. Obi C. Ezechukwu and Istvan Maros. *"OSCP: Optimization Service Connectivity Protocol"*, (forthcoming)
6. Robert Fourer, and Jean-Pierre Goux, *"Optimization as an Internet Resource"*, INTERFACES 31:2, pp. 130-155, 2001
7. Bjarni Kristjansson. *"Optimization Modeling in Distributed Applications: How New Technologies such as XML and SOAP allow OR to provide Web-based Services"*, http://www.maximal-usa.com/slides/Svna01Max/index.htm, 2001
8. *MPL Modeling System*. http://www.maximal-usa.com/mpl/mplbroc.html
9. *MPL OptiMax 2000 Component Library*, http://www.maximal-usa.com
10. *Simple Object Access Protocol (SOAP)*, http://www.w3.org/TR/SOAP/
11. *XML*, http://www.w3.org/XML/
12. *XML Schema*, http://www.w3.org/XML/Schema
13. *XSL*, http://www.w3.org/Style/XSL/

Appendix A. Multi-period Production Planning Model Solution

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<orml:solutionReport modelId="ProductionPlanning" modelInstanceId="Illustration"
                     xmlns:orml="http://www.doc.ic.ac.uk/~oce/elsinore/2003/ORML">

<solutionSummary objectiveValue="15795.0" solutionStatus="GLOBAL_OPTIMUM_FOUND"/>

    <variablesSolutionReport>
        <variableValues variableId="Produce">
            <variableValue value="200" reducedCost="0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="1">1</subscript>
            </variableValue>
            <variableValue value="220" reducedCost="0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="2">2</subscript>
            </variableValue>
            <variableValue value="200" reducedCost="0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="3">3</subscript>
            </variableValue>
            <variableValue value="190" reducedCost="0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="4">4</subscript>
            </variableValue>
            <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="1">1</subscript>
            </variableValue>
            <variableValue value="0" reducedCost="-7">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="2">2</subscript>
            </variableValue>
            <variableValue value="0" reducedCost="-7">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="3">3</subscript>
            </variableValue>
            <variableValue value="0" reducedCost="-7">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="4">4</subscript>
            </variableValue>
            <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="1">1</subscript>
            </variableValue><variableValue value="0" reducedCost="-6">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="2">2</subscript>
            </variableValue>
            <variableValue value="0" reducedCost="-6">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="3">3</subscript>
            </variableValue>
```

```
        <variableValue value="0" reducedCost="-6">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="4">4</subscript>
        </variableValue>
</variableValues>
<variableValues variableId="Store">
        <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="1">1</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="2">2</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="3">3</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="-40">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="4">4</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="-8">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="1">1</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="2">2</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="3">3</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="-35">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="4">4</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="-6">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="1">1</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="2">2</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="0">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="3">3</subscript>
        </variableValue>
        <variableValue value="0" reducedCost="-6">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="4">4</subscript>
        </variableValue>
</variableValues>
```

```
        <variableValues variableId="Sell">
                <variableValue value="200" reducedCost="0">
                        <subscript matrixLocation="1">Trousers</subscript>
                        <subscript matrixLocation="1">1</subscript>
                </variableValue>
                <variableValue value="220" reducedCost="0">
                        <subscript matrixLocation="1">Trousers</subscript>
                        <subscript matrixLocation="2">2</subscript>
                </variableValue>
                <variableValue value="200" reducedCost="0">
                        <subscript matrixLocation="1">Trousers</subscript>
                        <subscript matrixLocation="3">3</subscript>
                </variableValue>
                <variableValue value="190" reducedCost="0">
                        <subscript matrixLocation="1">Trousers</subscript>
                        <subscript matrixLocation="4">4</subscript>
                </variableValue>
                <variableValue value="0" reducedCost="-7">
                        <subscript matrixLocation="2">Shirts</subscript>
                        <subscript matrixLocation="1">1</subscript>
                </variableValue>
                <variableValue value="0" reducedCost="0">
                        <subscript matrixLocation="2">Shirts</subscript>
                        <subscript matrixLocation="2">2</subscript>
                </variableValue>
                <variableValue value="0" reducedCost="0">
                        <subscript matrixLocation="2">Shirts</subscript>
                        <subscript matrixLocation="3">3</subscript>
                </variableValue>
                <variableValue value="0" reducedCost="0">
                        <subscript matrixLocation="2">Shirts</subscript>
                        <subscript matrixLocation="4">4</subscript>
                </variableValue>
                <variableValue value="0" reducedCost="-6">
                        <subscript matrixLocation="3">Socks</subscript>
                        <subscript matrixLocation="1">1</subscript>
                </variableValue>
                <variableValue value="0" reducedCost="0">
                        <subscript matrixLocation="3">Socks</subscript>
                        <subscript matrixLocation="2">2</subscript>
                </variableValue>
                <variableValue value="0" reducedCost="0">
                        <subscript matrixLocation="3">Socks</subscript>
                        <subscript matrixLocation="3">3</subscript>
                </variableValue>
                <variableValue value="0" reducedCost="0">
                        <subscript matrixLocation="3">Socks</subscript>
                        <subscript matrixLocation="4">4</subscript>
                </variableValue>
        </variableValues>
</variablesSolutionReport>

<constraintsSolutionReport>
        <constraintValues constraintId="ProductionCapacity">
                <constraintValue shadowPrice="0.0" activity="20.0" slackValue="0.0">
```

```
                <subscript matrixLocation="1">1</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="22.0" slackValue="0.0">
                <subscript matrixLocation="2">2</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="20.0" slackValue="0.0">
                <subscript matrixLocation="3">3</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="19.0" slackValue="0.0">
                <subscript matrixLocation="4">4</subscript>
            </constraintValue>
        </constraintValues>
        <constraintValues constraintId="InventoryBalance">
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="1">1</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="2">2</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="3">3</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="4">4</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="1">1</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="2">2</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="3">3</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="4">4</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="1">1</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="2">2</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="3">Socks</subscript>
```

```
                <subscript matrixLocation="3">3</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="0.0" slackValue="0.0">
                <subscript matrixLocation="3">Socks</subscript>
                <subscript matrixLocation="4">4</subscript>
            </constraintValue>
        </constraintValues>
        <constraintValues constraintId="MaxStorageCapacity">
            <constraintValue shadowPrice="0.0" activity="800.0" slackValue="0.0">
                <subscript matrixLocation="1">1</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="800.0" slackValue="0.0">
                <subscript matrixLocation="2">2</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="800.0" slackValue="0.0">
                <subscript matrixLocation="3">3</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="800.0" slackValue="0.0">
                <subscript matrixLocation="4">4</subscript>
            </constraintValue>
        </constraintValues>
        <constraintValues constraintId="LimitSupply">
            <constraintValue shadowPrice="0.0" activity="1300.0" slackValue="0.0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="1">1</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="800.0" slackValue="0.0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="2">2</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="6000.0" slackValue="0.0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="3">3</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="3000.0" slackValue="0.0">
                <subscript matrixLocation="1">Trousers</subscript>
                <subscript matrixLocation="4">4</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="1750.0" slackValue="0.0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="1">1</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="1300.0" slackValue="0.0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="2">2</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="4000.0" slackValue="0.0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="3">3</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="8000.0" slackValue="0.0">
                <subscript matrixLocation="2">Shirts</subscript>
                <subscript matrixLocation="4">4</subscript>
            </constraintValue>
            <constraintValue shadowPrice="0.0" activity="4000.0" slackValue="0.0">
```

```
                    <subscript matrixLocation="3">Socks</subscript>
                    <subscript matrixLocation="1">1</subscript>
                </constraintValue>
                <constraintValue shadowPrice="0.0" activity="3000.0" slackValue="0.0">
                    <subscript matrixLocation="3">Socks</subscript>
                    <subscript matrixLocation="2">2</subscript>
                </constraintValue>
                <constraintValue shadowPrice="0.0" activity="9000.0" slackValue="0.0">
                    <subscript matrixLocation="3">Socks</subscript>
                    <subscript matrixLocation="3">3</subscript>
                </constraintValue>
                <constraintValue shadowPrice="0.0" activity="15000.0" slackValue="0.0">
                    <subscript matrixLocation="3">Socks</subscript>
                    <subscript matrixLocation="4">4</subscript>
                </constraintValue>
            </constraintValues>
        </constraintsSolutionReport>
</orml:solutionReport>
```