Chapter 1

Overview of New Source Selection Functionality in the DLA Procurement Process

A Background

Between July 1996 and April 1999, ISI developed for DLA an innovative approach to managment of Finished Goods Inventorys, providing flexible, user-controlled decision making for automated sourcing of Long-Term Arrangements. The system comprised approximately 107,000 lines of code. That programming work was preceded by and supported by a great deal of conceptual design work.

Our code provides editors for business rules and contracts, support for DLA operations with a flexible underlying contract representation, and a general and extensible approach to providing sourcing services. In addition to the functionality, the design of the code reflects a great deal of thought to extensibility and maintainability. The implementation largely reflects those design concerns. The implementation is highly modular; it has been made fully compatible with clusters but is still usable with other approaches.

This chapter provides an overview of the system. The conceptual design and implementation is detailed in chapters following.

A.1 Work to produce the rule and contract editor

Comprehensive design

1) Supports a flexible range of contract types:

- few items, thousands of items
- catalog, highly managed
- group items hierarchically (like folders in PC), specify defaults
- conditional field values according to multiple dimensions (e.g., price conditional on quantity and region).

- Facility to make any field conditional
- Easily extensible to add new fields and new types of conditions

2) Comprehensive range of features to support user tasks

- Drafts vs. production
- Data versioning to support changes at period of performance boundaries or arbitrary times.
- Possible to enter today values that take effect at some point in the future.
- Novice vs. expert
- Descriptions of all fields and codes
- Dialogue structure designed to prevent errors

3) Authentication and workflow features

- Security to prevent unauthorized access to information
- Team and privilege-based mechanism to help organization manage vast number of contracts and data

4) Hooks for future

- Hook for adding notes to fields
- Hook to approve changes to contracts incrementally
- Hook to lock contract data to prevent unintended modification (locks for phases, items, fields)

5) Professional graphic design

- Early mockup to explore design ideas and collect feedback from users
- Professional design of dialogue and visual appearance

6) Support for geographies

- Define and modify geographies based on states and countries
- Geographies can be used to define rules or prices

7) Easy to use interface to specify rules

- Multiple types of rules
- Highly tuned interface to enter rules

Editor Implementation

Framework to implement simplified version of editor first, and extend as needed.

- Template-based generation of screens: enables very low cost modification of editor (e.g. add new fields, redefine layout)
- Web-based editor using modern browser facilities to make interactive pages (XML, Dynamic HTML)
- Significant functionality already available

A.2 1.2 Work to produce a comprehensive Contract Representation

1) Flexible representation scheme to support a wide range of contracts

- Easy to add new fields
- General mechanism to split field values according to the values of other fields. For example, price can be a single value, split by DVD/Stock, split by quantity, split by region, split by FMS or split for any combination. The representation allows an open ended set of fields with an open ended set of conditions to split the value.
- Hierarchical grouping mechanism to define defaults for fields, even when split according to many conditions
- Support for keeping multiple versions of field values to support scheduled changes to data (on period of performance boundaries or at arbitrary times)

2) Single master representation supporting editing, business rule processing, reports and data import

- Master representation in XML
- Editor works directly on the XML
- API provides convenient access to XML representation for rule processing
- Translator from XML to relational representation (under construction) supports automatic generation of tables to support reports

- Data import directly from XML

A.3 Work to produce the Sourcing Module

1) Modular implementation independent of communication infrastructure

- Works with clusters
- could work with any other data communication infrastructure

2) All decision making represented in rules

- Business rule interpreter core written in Java
- Clean APIs to data sources (contracts, requisition data, geographies, etc)
- Default behavior coded using rules (not accessible from GUI, nevertheless represented as rules so in the future can be accessed via rule editor). Long term goal is to enable CBUs to specify their own default rules
- User policy rules accessible via editor (same rule representation as default rules, but GUI designed for them)
- Interim implementation: some default rules are implemented in Java and "wrapped" as rules. This allowed us to use the generic rule interpreter for all decision making, and allowed us to encode complex behavior in Java. This scaffolding technique allows us to build all system functionality before we are ready to encode all behavior in the rule representation language.

3) Comprehensive documentation of all data flow in the sourcing module:

- Starts with list of all fields that need to be sent to the vendor and
- depot clusters.
- For each field trace all the computations back to other intermediate
- fields or to fields coming from the contract or the requisition (how many pages is this?)
- Detailed documentation of all issues for computing the values. If we are unsure about the logic for computing a field we have an issue in the document. If we are unsure whether we will have access to the data source for a field, we have an issue in the document.
- Document was created, but not kept completely up to date

4) Data sources

- API to access contract data from XML representation
- API to access contract data from PartNet
- Once data enters the sourcing module via any of its sources it is created the same. This means that it is easy to extend the sourcing module to make use of other data sources.

5) Implementation

- The basic interpreter is complete, although some enhancements may be needed for rules we have not coded.
- Significant number of rules have been implemented. (Note: we still have many issues unresolved because we're awaiting guidance: (i) some basic rules are not implemented (ii) some basic rules were implemented with our best guess for resolving an issue -- may be wrong.)

6) Comprehensive test harness and test GUI

- Provides visibility into decision making mechanisms of the sourcing engine
- Provides a GUI to enter requisition parameters
- Shows all sourcing possibilities considered for a requisition
- For each source, shows all computed attributes and intermediate results
- For each source, shows which rules applied and records their effect

A.4 Work on Clusterization

1) Cluster wrapping code makes our module play in the cluster society

2) Implementation of clusterization is complete for the forward channel (may need a few refinements).

3) Most of the work remains in the backward channel and the handling of exception conditions.

4) Our sourcing module is independent of cluster -- can be used as a module in a different architecture.

A.5 Other work

In addition to our work that is directly manifested in the system, we invested a great deal of hidden effort in oversight of NNY on GUI design and implementation. To get NNY to the point where they could understand the design, we developed an initial partial GUI that implemented 181 screens

via 12,000 lines of HTML. NNY produced their work by redesigning and reimplementing this initial version. For them to do so, we delivered requirements for, and individually reviewed, 187 separate screen design documents produced by NNY. We incorporated into our editor software 175 GIF images delivered by NNY, as well as assimilating 1,211 lines of JavaScript and 7,500 lines of HTML for example screens. Using those deliverables from NNY, we generated additional screens, and are in the process of writing XSL code that generates the corresponding HTML for any contract.

B Functional Requirements

The DEALMAKER sourcing agent is part of a larger society of agents that automates the processing of requisitions for DLA. The sourcing agent receives requisitions from other agents and determines the best source of supply for each requisition. Other agents send the orders to the vendors using the appropriate EDI transactions, handle the financial transactions with the customers and the vendors, and handle vendor responses.

The sourcing agent is implemented internally as a blackboard / production architecture organized around these three tiers of developers. The core is an interpreter of grammar-based rule templates and instances, which may be extended by programmers to add new low-level functionality. Following the constraints of the grammar, system administrators can create templates that define new classes of rules. Following those templates, buyers and contract managers can create new instances of these classes. Details and examples follow.

The sourcing agent processes requisitions in four stages:

1) *Generate proposals*. Using the item identification from the requisition, the system queries the database of contracts and an external source of vendor catalogs, such as PartNet, to find all contracts that supply the item. For each match, the system constructs a proposal object that represents the possibility of using that contract to supply the item. All attributes of the requisition and relevant attributes of the matched contract are copied onto each proposal.

2) *Elaborate proposals*. Compute and validate all attributes needed to compose the EDI transaction to submit an order to a vendor, including cost to DLA, marked-up price to DLA's customer, and expected delivery date. Rules implement DLA standard policy and contract-specific exceptions.

3) *Filter proposals*. Apply rules to verify that a proposal meets all the particular policies that have been established to use a particular contract for the particular requisition that needs to be filled. Filter rules annotate the proposal with recommendations to kill it or to use it in preference of other proposals.

4) *Rank proposals*. Apply rules that organize proposals into equivalence classes based on the requisition's priority code and desired delivery date. The rules successively split equivalence classes into smaller classes. The winning proposals are found by traversing the resulting tree of equivalence classes to find the first non-empty leaf class.

The processing for all steps except the generation of proposals is under the control of rules. The elaboration, filtering and even the ranking rules work on individual proposals. The system processes

one proposal at a time. The system applies the rules according to data-flow dependencies among rules. This assures that rules that test a computed attribute are attempted after the rule that computes the attribute. All rules whose condition is satisfied are applied, adding an annotation to the proposal. The annotations for the elaboration rules record computed attributes, the annotation for the filtering rules record the effect of the filter (e.g., kill) and the annotation for the sorting rules record the equivalence class to which the proposal is assigned.

In the DLA application, the elaboration, ranking and most filtering rules are ICP-global, and are expected to be handled by system administrators rather than by contract managers or buyers.

The elaboration phase is quite complex from a user perspective. Set-up is intended for a system admininstrator, and involves the specification of about 30 different rules. One example is the computation of the price that the customer pays DLA based on the price stated in the contract: depending on characteristics of the item, different price adjustment factors are applied. Another example is the computation of the number of items that must be ordered. This computation is complex because the units used in the requisition (e.g., each) may differ from the units in which a vendor supplies the item (e.g., dozen). In addition, items may be requested or sold in packs whose units do not match (e.g., box of ten dozen). In order to align the units, the rule may propose to increase the number of items to be ordered. The system features two price computation rules, one that computes the price for the number of items that the customer wants to order and one that computes the price for the number of items that need to be ordered to satisfy unit of issue and multi-pack requirements. Note that if the quantity was increased it may push the order to a different price tier, in which case the total order price will be computed using a different unit price. During the filtering phase, several filtering rules are applied to verify that the quantity to be ordered is valid. One is an ICP policy rule that specifies a threshold for price increases due to quantity increases. The others are rules that validate the price increase with respect to attributes of a requisition that specify whether the customer demands the exact quantity to be shipped and the threshold for price increases. If these validations fail, the corresponding rule places a "kill" annotation on the proposal.

The filtering phase includes many ICP-global filtering rules such as those described above.

There are three types of filtering rules under contract manager/buyer control. These rules give users extensive control for managing the contracts.

- *Exclusion*: these rules specify the conditions under which a requisition should be excluded from a contract or from specific items in a contract. For example, exclude all low priority, Army requisitions from contract X, but allow all requisitions from Fort Hood.

- *Must use*: these rules specify that certain requisitions must use a specific contract or set of contracts. For example, all low priority Army requisitions (for items supplied by contract X) must use contract X.

- *Serves only*: these rules specify that a contract only accepts requisitions with certain characteristics. For example, contract X serves only low priority Army requisitions.

DEALMAKER provides an easy-to-use interface to allow contract managers and buyers to specify these rules.

The ranking algorithms work by sorting proposals into different equivalence classes. The kill annotation defines the first equivalence class (i.e., killed vs not killed). Then, depending on the ranking algorithm selected, it either subdivides the not-killed proposals into those that meet the customer's required delivery date and those that don't or sorts the not-killed proposals by cost with those tying for cheapest being the top class afterward. Other rules divide the equivalence classes into finer grained classes. The ranking algorithm uses the attributes and annotations on the proposals.

Chapter 11

Finally, a random (or round-robin) selection of a winner is made from those proposals in the top equivalence class.

Contract Representation

C Background

In an operating DELTA system, the DELTA Contract Editor will be used to create, import, edit, and maintain contract data (and related data) for use by the DELTA Sourcing Engine, the PartNet interface, the Report Generation module, and other uses. These modules require a shared data model and a persistent data repository to intercommunicate. The DELTA Logical Data Model and Database provide these services.

D Functional Goals

D.1 Environment

The DELTA Logical Data Model and Database were designed to operate in an end-user production environment at one or more DLA sites. There are certain general functional requirements that most be met in this environment.

D.1.1 Concurrency

In an operating DELTA environment, there will be multiple users (contracting officers, buyers, etc.) accessing and maintaining contracts independently of each other. To this basic DLA functional requirement, ISI added the following more specific goals based upon our general experience in this domain, as well as specific knowledge of the DELTA environment:

D.1.1.1 Different Contracts

Users who are working on different contracts should be able to work without interfering with each other and without being required to perform special coordinating actions. The fact that the system supports multiple concurrent users should be transparent to the users in this circumstance.

D.1.1.2 Same Contract, Different Items

Users working on the same contract should be able to add and make concurrent changes to different items on the contract, or to the contract header and different items, without the risk of interfering with each other and without requiring special coordinating actions.

D.1.1.3 Same Contract, Same Item

It is not necessary to allow two or more users to edit concurrently the same line item or header of a single contract. When a conflict might occur, the users involved should be notified as soon as possible and the conflict should be resolved.

D.1.1.4 EDI Data

The DELTA Logical Data Model includes contract data fields that serve different purposes in DLA processing. In particular, the contract EDI data may be of interest only to specially designated EDI Specialists. Depending upon the operational workflow and workload at the DLA site, it may be desirable to allow the EDI Specialist to add, import, edit, etc. EDI parameters in parallel with other users of the same contract.

D.1.1.5 Incomplete Changes Don't Affect Operations

Users will want to be able to add, import, edit, etc. contract data (and related data) in parallel with Sourcing Engine operations on live requests. The Sourcing Engine must not act upon partial, inconsistent, or unapproved changes to the contract data (and related data). For example, a Buyer who is adding new line items to a highly managed contract might want to be able to split the data entry over two or more editing sessions, but might not want the changes to become visible until they all have been completed and reviewed.

The DELTA Logical Data Model and Database are required to provide support for these functional goals.

D.1.2 Performance

The applications accessing the DELTA Database (the Contract Editor, the Sourcing Engine, and the report and queries) each have particular data access patterns and performance requirements. The Contract Editor requires rapid access to the "header" of a contract, and a table of contents of the line items that are named in the contract (when appropriate). The Sourcing Engine requires rapid access to all lines, in all contracts, that supply a particular item. The DELTA Logical Data Model and Database must provide an appropriate implementation (including data object representation and database indexing) in order for the other DELTA applications to be able to meet their performance goals.

D.1.3 Reliability and Maintenance

The DELTA system should protect its data integrity from system and application malfunction (power failure, operator mistake, etc.) To this general requirement, ISI added the following more specific requirements:

D.1.3.1 Crash Resistance and Automated Recovery

The DELTA system will be a critical component of DLA logistics support. It needs to be up and running with minimum unplanned downtime. The system should be resistant to crashes, but if one should occur, the system should automatically recover to a consistent state and resume processing the production workflow. There are standard techniques, such as applying transaction-oriented relational database management systems, to address these requirements. The DELTA Logical Data Model and Database must be designed with these functions in mind.

D.1.3.2 Code Maintenance

The DELTA system will have several components operating in parallel on a common database. Such systems can be difficult to develop and debug in practice, because it is often difficult to determine which section of code was responsible for inserting an incorrect data record. The DELTA Logical Data Model and Database should incorporate features to facilitate software debugging and, when necessary, manual database recovery.

D.1.3.3 Data Representation Maintenance

The DELTA system is designed to evolve over time; adaptability to new business environments is one of its key overall design goals. The DELTA Logical Data Model and Database are required to grow and expand in pace with this dynamic environment. For example, it should be possible to add new data attributes without invalidating the existing contracts in the system.

D.1.4 Historical Data

There is a potential functional requirement in DELTA to retain (warehouse) superceded data for reporting, auditing, data mining, operation review, and/or system reliability analysis. For example, if a contract parameter, such as the minimum allowable delivery order dollar amount, is changed on a particular date, it may be required to retain a copy of the old value of the parameter and the date/time at which the change took place (and the identity of the user making the change). This information would be useful when verifying that the DELTA system properly processed prior requests, or when creating a report on the average number of changes to a contract after it is entered into the system, or for other purposes. In addition, the retention of historical data can be useful in system debugging and in setting up demonstrations. To the degree that this functionality is actually required, it is necessary to provide support for it in the design and implementation of the DELTA Logical Data Model and Database.

D.2 Interface

D.2.1 User Identification/Access Control

The DELTA Contract Editor (and other DELTA components) need to identify individual users, and in some cases perform user-specific actions. For example, the Contract Editor needs to know the identify of its user in order to record that the user is editing a particular contract. Certain contract attributes, such as the vendor EDI profile, may be restricted such that only certain types of user may change them. The DELTA Logical Data Model and Database must support the notion of user identification (until superceded by a higher-level DLA user identification system), and must support user-specific database access restrictions to the extent required by specific implementation requirements.

D.2.2 Export/PartNet

An operating DELTA system will support several classes of contracts. In some of these contracts, called highly-managed contracts, the DELTA Database will be the primary repository for item-specific information (item identification, unit pricing, etc.) for some or all of the items in the contract. Other DELTA components (Emall, PartNet) will require access to this data; the DELTA Database is required to support

an appropriate data export interface for them.

The performance and interface requirements of the external components may restrict the design used by the DELTA Database to implement the data export function. For example, PartNet expects that the item-specific data will conform to a certain general schema. Depending upon the implementation details of the DELTA Logical Data Model and Database , the data export interface to PartNet could be trivial or it could be complex.

D.2.3 Reporting/Queries

The DELTA Logical Data Model and Database must provide data objects to canned and ad-hoc reports. Typical reports might be queries such as "Which contracts are due to expire in the next 60 days?", "How many items are available through multiple DELTA contracts? How many are not?", or "How many new contracts have been added or modified in the last 7 days?"

D.2.4 Import

The operating DELTA system must support the import of initial data from existing DLA procurement systems, such as POPS and EPPI. Not only must the DELTA Logical Data Model and Database support the semantics of the POPS and EPPI contracts, in line with the requirement to support existing DLA ways of doing business, the DELTA Logical Data Model and Database were required to support a stable import representation for the POPS and EPPI data. This import representation was required to remain stable across a reasonable set of design changes (affecting the DELTA Logical Data Model and Database) regarding the details of contract data representation in support of the Contract Editor's and the Sourcing Engine's implementation and operation.

D.3 Domain

D.3.1 Legacy Compatibility

The DELTA system must be able to operate within the environment of the DLA's existing Long Term Arrangement contracts and requisition documents. For example, DELTA should be able to assume processing control over most or all POPS or EPPI contracts at a DLA center (ICP). DELTA must be able to support the pertinent semantics and business model of these existing contracts and requisitions in order to process them correctly. In turn, this requires that the DELTA Logical Data Model and Database must be able to support the required data representation and provide persistent storage so that the Contract Editor and Sourcing Engine can implement the required functionality.

D.3.1.1 MILSTRIP A0 Support

The DELTA system must support pertinent semantics of MILSTRIP A0 requests. This includes such things as identifying customers and customer groups by DoDAACs, supporting different types of expedited processing and delivery flags, ensuring that sufficient information is available to send the appropriate MILSTRIP responses to incoming requests, etc.

D.3.1.2 NSN and CAGE/PN Support

The legacy DLA systems were built around the notion that all stocked items and all recurrently ordered items would be identified by a National Stock Number (NSN). This approach has been augmented with the ability to support a more commercially-acceptable part identification scheme, such as manufacturer (or supplier) identification (CAGE code) and part number,

D.3.1.3 Emall Support

DELTA is also required to support the semantics of EMALL orders. This might require special support from the DELTA Logical Data Model and Database.

D.3.1.4 POPS/EPPI Support

DELTA is required to support contracts that were developed for use with the legacy POPS and EPPI systems.

D.3.2 New Functionality

Not only must DELTA support the DLA's existing ways of doing business, a primary goal of the DELTA effort is to create a system that can support new ways of doing business without reprogramming. The DELTA Logical Data Model and Database provide the necessary data structures and flexibility to support this requirement.

For example, quantity-based pricing is common in existing DLA LTA contracts. Contract pricing that depends upon the delivery region as well as quantity occurs less frequently, as the legacy order processing software (SAMMS) requires special programming and data entry to implement this pricing arrangement. In some cases the DLA has negotiated special LTA clauses that reimburse vendors for additional costs of expedited shipping and handling for urgent (high priority, or other criteria) orders. DLA personnel have indicated that it would be desirable if the DELTA system could incorporate varied pricing factors, such as geography, priority, and customer type, in a cleaner, more maintainable fashion than the legacy software provides. The DELTA Logical Data Model and Database must be able to adapt to and support new models of this sort.

D.3.3 Support for Time Representation

The Data Model and Repository must support several types of time-dependent contract data:

D.3.3.1 Periods of Performance and Standard Year Boundaries

There are time periods that are important to the structure and reporting requirements of DELTA contracts, in particular the Base Period (Base Year) and Option Periods (Option Years); together, these are called the contract's Periods of Performance. Calendar Years and Government Fiscal Years are also periods that are important to some aspects of contract administration. Certain data is supposed to change (or be reported) only at these period boundaries (but see (3), below). The Contract Editor must support these special time boundaries and their related attributes when creating and maintaining DELTA contracts. The DELTA Logical Data Model and Database must provide support for representing and storing data that is segmented by these periods.

D.3.3.2 Other Contract-specific Periods

There is data that may change (or be reported) periodically at other than a Period of Performance. For example, in some existing contracts the contract pricing may change on a monthly basis instead of an annual basis. The DELTA Logical Data Model and Database must provide support for representing and storing data that is segmented by these contract-specific periods.

D.3.3.3 Scheduled Changes

Certain contract data is subject to scheduled changes and updates. For DELTA, the most common example of this is price data in highly managed contracts, which may require data updates on an annual, monthly, or even daily basis. Other changes that may be scheduled in advance include adding or deleting items from a contract, and adding or deleting customers from a customer class. It is not always feasible to have operations staff (or even automatic data transfer programs) update the affected data at the instant that the change is supposed to take effect. Thus, DELTA should be able to load upcoming changes (e.g., new prices) in advance of their effective date, ensuring that the Sourcing Engine sees the new data at the appropriate time. The DELTA Logical Data Model and Database must provide appropriate support for implementing this functionality in DELTA.

D.3.3.4 Unusual Changes

In unusual cases, contract data might change at an arbitrary time, not related to one of the pre-established standard or contract-specific period boundaries. The potential reasons for such changes are varied: abrupt market changes, government or contractor reorganization, natural disasters, special conditions due to a state of war. As a consequence of these circumstances, prices, availability codes, shipping points (both source and destination), modes of shipment, and other contract attributes may need to change immediately. The DELTA Logical Data Model and Database should be flexible enough to support urgent changes of this nature without breaking other, conflicting functionality (such as retaining historical data, if required).

D.3.4 Contract Size

Although most existing DLA long term arrangement contracts are for one or a small number of items, some DLA Long Term Arrangement contracts are for catalogs of tens of thousands of items. The PartNet system will provide access to item-specific pricing information and other data for large contracts in the future, but in the short term there is the requirement to import data from large legacy contracts into the DELTA Database. The DELTA system, including the Contract Editor and the Sourcing Engine, needs to accommodate both large numbers of small contracts and small numbers of large contracts. The Logical Data Model and Database provide appropriate support such that the system's user interface and performance requirements in spite of varying contract size.

D.3.5 Contract Types

The DELTA system must be able to support a variety of contract models. In all contracts, the DELTA Logical Data Model and Database must store certain information about the contract as a whole. Other fields in the contracts, and in particular in the contract items, depend upon the type of the contract. The DELTA workflow for processing these contracts may also depend upon the contract type.

D.3.5.1 Highly-Managed Contracts

Highly-managed contracts are ones that are tightly negotiated and monitored by a DLA unit (CBU or equivalent). They may supply high-volume items for which a special price has been negotiated, or supply items the acquisition of which are being specially monitored and tracked, or they support a particular customer or program code with special order processing requirements. These contracts require that the DELTA Logical Data Model and Database support a very detailed representation of item prices, availability codes, and other attributes.

D.3.5.2 PartNet Catalog Contracts

In the DELTA architecture, the PartNet interface to vendor-maintained item information databases will provide access to vendor catalogs of items under LTA contract. The PartNet interface will, among other functions, will locate the vendors that source a requested item, and will provide pricing, availability, and other information about the item. This information will be combined with information in the DELTA Database to provide the complete set of information required by the DELTA Sourcing Engine. The DELTA Logical Data Model and Database supply the additional data needed to supplement that provided by PartNet.

The degree of catalog contract support provided by the DELTA Logical Data Model and Database may vary from contract to contract. For example, the initial PartNet system does not support regional pricing, but if a contract's regional price variations are defined to be a constant per-region adjustment factor, the DELTA Logical Data Model and Database could add the information needed to support regional pricing in this contract.

D.3.5.3 Prime Vendor Contracts

There are contracts for which little or no item-level information is available, possibly not even the item identifications themselves. The DELTA Sourcing Engine must connect requisitions with the contract and route them to the vendor based upon request attributes other than NSN or CAGE/PN. The MILSTRIP project code is a good candidate for routing requests to these contracts, but other attributes (requestor's DoDAAC, ship-to DoDAAC) are possible as well. Input received from DLA personnel indicates that it would be desirable for DELTA (and thus, the DELTA Logical Data Model and Database) to support this type of contract.

D.3.5.4 Future Contract Types

The DELTA system, and the DELTA Logical Data Model and Database, should provide rapid, flexible support for other contract types currently used in the DLA, and ones yet to be determined.

D.3.6 Pricing Models

As mentioned in earlier sections, pricing models are of particular importance to DELTA contracts. The DELTA Logical Data Model and Database must support certain initial pricing models to import existing DLA contracts, including standard unit pricing, quantity pricing, regional/quantity pricing, and customer/quantity pricing. Interviews with DLA personnel have indicated a desire for innovative pricing models, such as priority-based pricing. Other factors affecting pricing models include the frequency and type of changes to pricing that may occur, and whether pricing data is stored locally to DELTA, remotely (i.e., PartNet), or a combination of the two.

There are two types of prices of primary interest in DELTA: the price that the DLA charges its customers, and the price that the vendor charges the DLA. These are independent quantities, and separate pricing models could apply to each. For example, customer orders that are filled from depot stock are charged a standard unit price, while depot stock replenishment orders may be subject to quantity breaks in a contract price schedule. Alternatively, the customer and vendor price may be related by a formula, such as this example from DSCC LTAs: the vendor supplies a "list" unit price (possibly including quantity breaks), which is decreased or increased via a Price Adjustment Factor specified in the contract; this results in the unit price that the DLA will pay the vendor. A further increase, the Cost Recovery Rate, allows for DLA overhead costs in administering the contract; this results in the customer unit price.

In other existing DLA contracts, the customer pays a fixed price regardless of order priority, but the DLA pays the vendor a premium (through a separate fund) when high-priority, expedited processing is required. It would be useful if the DELTA system would allow the DLA to more directly support priority-based pricing, passing the additional charges directly through to the customer.

The DELTA Logical Data Model and Database should support both existing and as-yet undefined future pricing models as effectively as possible.

D.4 Usability

D.4.1 Common Values

Multiline contracts may contain values (example: FOB Point) that are the same for all items in the contract. In the more general model of DELTA contracts (as determined through examining existing contracts and data processing software, and through interviews with contracting specialists) there are occasional contracts for which these values may vary on a line-by-line (or subline) basis. Thus, these attributes are modeled at the line item basis in the DELTA Logical Data Model. However, in order to minimize user effort, the DELTA Contract Editor should minimize the amount of redundant data entry required for the most frequent types of contracts. The DELTA Logical Data Model and Database must support the ability to store common values in a single location, and exceptions on a line item basis. However, see below for the complications due to sublines.

D.4.2 Groups and Group Values

Some contracts contain named groups of relate items with certain shared attributes, with the values of the attributes varying from group to group in the contract. As above, the DELTA Contract Editor should be allow the user to enter any group-specific data once, rather than requiring data entry. The DELTA Logical Data Model and Database should support the function of associating items with groups, and of storing shared item attributes in association with groups.

D.4.3 Flexible Subline Criteria

In DLA LTA contracts, a single item may have multiple sets of prices, availability codes, and certain other attributes via "sublines". The attributes or conditions by which items are split into sublines may vary from contract to contract. Generally, these conditions relate directly to the parameters of a requisition (or similar request) being processed against the contract. The Data Model and Repository should support conditions such as DVD vs. Stock, FMS/MAP, specific

customer, and delivery region (CONUS/OCONUS). Other values were identified as desirable conditions for DLA LTA contracts. The DELTA Logical Data Model and Database should support the existing subline criteria, and should be structured to allow new subline criteria attributes to be added without perturbing existing contracts and without extensive reprogramming.

E Design and Architecture

Not due in this installment. To be written in upcoming installment.

F Design Rationale

Not due in this installment. To be written in upcoming installment.

Sourcing Module

G Background

To be written. This section will describe the concept behind the Business Rule Manager, explaining the approach by which selections are made by sequential applications of a number of different classes of rules. It will describe the classes of rules, the specific rules defined within each class, the inputs and outputs of the system.

H Functional Goals

To be written. The difficulty in developing this section has been that the functional goals are so closely intertwined with the specfics of rules (described in the following section) that it is difficult to talk about them separately. We are considering simply merging the two sections and adding more explanatory background material to what is now Section C.

I Design and Architecture

The current format underlying this document is a web site, rather than a text document. It is based on the ongoing records of issues that ISI has been maintaining throughout the project. Extensive work has been done to bring the material up to date. Additional work is needed to provide more explanatory material introducing each subsection.

In the final report, unless otherwise directed, ISI will turn over the web site for rehosting at a site of DLA's choosing. We recommend that as preferable, because the extensive cross-referencing is difficult to follow in a passive document. The alternative is to do an importation from the web site into Word, while maintaining the cross-referencing. We recommend against that, because it requires labor-intensive manual formatting with no real value added to DLA and the printed version is no easier to follow.

I.1.1 Sourcing Module Flow of Data

To find the root or latest version of these documents, visit ISI DELTA Sourcing Data. Text in this color is a show-stopper issue. Text in this color is an issue that should be resolved for a correct or efficient system. Text in this color represents an issue that was not resolved at the point where ISI stopped implementation.

I.1.1.1 Fields in Output Classes Produced by DELTA Sourcing

ALPINE defined these classes in the FGI cluster code.

I.1.1.1.1 Outputs Specific to Vendor Recipent Cluster

Java class name: VendorSupplySourceImpl extends SupplySourceImpl implements VendorSupplySource, NewVendorSupplySource

Property Name	Java Type
vendorContract	VendorContract
vendor	Vendor

Outputs Specific to Depot Recipent Cluster

Java class name: DepotSupplySourceImpl extends SupplySourceImpl implements DepotSupplySource, NewDepotSupplySource

Property Name	Java Type	Comments
TBD	TBD	Is in LDM

Outputs Specific to DRMS Recipent Cluster

Java class name: DRMSSupplySourceImpl extends SupplySourceImpl implements DRMSSupplySource, NewDRMSSupplySource

Property Name	Java Type	Comments
DTID	String	The DTID will be supplied to us by PartNet on DRMS queries. Typically, this will be a big-bucks item which is managed by a serial number of some sort in addition to the NSN. Conceptually, the DTID similar to a vendor part number or catalog code. On the other hand, it represents a *specific* instance or lot of returned material, which raises some questions which should be addressed at some time: 1) Suppose the customer requisitions, say, one tank by NSN, and DRMS has two of them on hand. Will PartNet give us two seperate responses, with seperate DTIDs? 1.1) If so, is there a specific procedure that we should follow for selecting the DTID to send back to DRMS with the requisition? Say, earliest? If so, is there a date field inside the DTID, (is there a four-character ordinal date as for a standatd MILSTRIP document number)? 1.2) If we get two customer requests for the same NSN, and PartNet gives us two DTIDs, should we allocate the first DTID to

		the first customer request, the second one to the second request, etc.? 2) Can a PartNet DRMS response to a quantity and unit-of-issue other than "1 EA" (presumably, in a single DTID)? Answer: YES 2.1) If so, is a customer allowed to requisition only part of that quantity? 2.1.1) If so, and if there is a stream of customer requests, is there any need to manage the "quantity on hand"? This raises the same technical issues as synchronizing with Depot stock levels, I suppose. 3) If DRMS responds with multiple DTIDs for the same NSN, and the customer request is for a quantity that is greated than the quantity avaialable in any single DTID's stock, but which could be satisfied by a combination of DTIDs, what do we do about it? [Pedro] Craig, this discussion of DTID seems to address issues beyond what is required from us in the short-term. I don't understand the issues completely, but I am afraid we may be addressing many more contingencies than we have to, causing Steve Bohn to give us a hard time. I do not understand why these are not short-term issues. It appears to me that we need to answer these questions in order to generate code that "correctly" sends requests to DRMS in January. [Pedro] DTID short term/long issue: I rather get the vendor and depot stuff working first. I consider the DTID issue longer term with respect to vendor/depot issues that we may still have.

Outputs Common to all Recipent Clusters (Vendor, Depot, DRMS)

Java class name: SupplySourceImpl implements SupplySource, NewSupplySource

Property Name	Java Type
CBUSourcingTimestamp	Date

Java class name: Vendor

Property Name	Java Type	Comes From	Comments
vendorName	String	org.org_nm	These should be compatible. Is character case an issue? The database is implemented to be case-retentive with efficient case-insensitive searches. So, we can easily supply an upper-case-only copy of the vendor name, if desired. Otherwise, we'll potentially supply a mixed-case name.
vendorAddress	MailingAddress	Use LTA.VENDOR_ID to lookup vendor records.	needs more work, see below: MailingAddress.

vendorCage	String X(40)	lta.vendor_id or lta.org_id, if the vendor is identified by CAGE code. The database may need to be straightened out about "lta.org_id"; in the Lockheed Martin model it may have been intended to provide a contract administrative link. Database format: VARCHAR2(5) or VARCHAR2(30)	Why is the requested width 40? A CAGE code ought to be exactly 5 characters wide (and the database could be changed to CHAR(5) to reflect this, I suppose, except for potential difficulties with foreign key compatibility).
adminLoc	String X(3)	lta_sub_line_item.admin_by VARCHAR2(3)	Note that Alpine has listed this as "vendor" data, but it is recorded in the Lockheed Martin logical data model (and our current database implementation) as contract-specific data at the *subline* level. I've often used this field as a "typical example" for discussion purposes, when exploring the underlying semantics of the Lockheed Martin logical data model. These ought to be compatible, modulo the issue of padding if any values are less than three characters wide. My impression is that these values ought to always be exactly three chars wide; perhaps the database format should be changed to CHAR(3)? On the other hand, this may be an organizational code for which we may want to impose validity checks and/or compute containment relationships, so perhaps this database field should key into the ORG table?

			Question: is there a handy list of admin_by values. [Pedro] Why is it important to note that ALPINE listed this as "vendor" data but LM has it in the LTA. I assume the data requirements are ultimately coming from Steve, and he already knows that. Does noting this change the answer in any way? If not, I suggest deleting this note and similar notes that appear below. Dan Cerys and I have discussed the issue of the "vendor" data, on the phone and in the latest Boston meeting. I have told him that the "vendor" data includes fields which were specific to the contract, rather than being constant for the vendor across all contracts. Dan asked me to identify the fields, so he could change his data structures appropriately.
paymentOffice	String X(2)	lta_sub_line_item.pay_office VARCHAR2(2)	Note that Alpine has listed this as "vendor" data, but it is recorded in the database as contract-specific data at the *subline* level. Note: these ought to be compatible, modulo the issue of padding.
tradingPartnerId	String X(11)	a new field is needed VARCHAR2(11)	I wonder whether this should be a vendor-specific field or a contract-specific field, or even a subline-specific field. Certainly, FAST saw cases where different products went to different vendor offices, or orders had to be routed to vendor offices based upon sales region. Unless we receive a functional requirement to the contrary, my recommended

			approach will be to create a per-contract field which is initialized, upon contract creation, from a per-vendor field.
stateCountryCode	String X(2) numeric for US state codes alpha for country codes	lta.state VARCHAR2(2)	I believe I have the list of country codes, but I don't have the numeric codes for US states handy. Also, the DELTA field name should be changed for clarity. [Pedro] We need the translation table for the LTA GUI.
socioEconomicCode [MOVED to VendorContract]]	String X(1)	lta.socio_econ_ind CHAR(1)	Note that Alpine has listed this as "vendor" data, but it is recorded in the data as contract-specific. Furthermore, in FAST we learned that a single company could qualify as a small business in some commodity/service areas and not in others. So, we'll store it in the LTA, but initialize it from a vendor-specific value at contract creation. Another problem: what should happen if the vendor's socia_econ_ind eligibility changes during the duration of the contract? Can such changes [usually] happen only at certain periods (calendar year, GFY, period of performance, etc.)? Note: the formats appear compatible. What are the values? (I might be able to pull a value set for this domain out of the FAST records.)

Java class name: VendorContract

Property Name	Java Type	Comes From	Comments
contractNumber	String X(13)	lta.lta_id VARCHAR2(13)	Note: the formats appear compatible. [LTA ID, PIIN]

releaseNumber	String X(30)	vendor_order.call_num VARCHAR2(4) last_coc.last_coc_num VARCHAR2(5)	Note: I know we've discussed in the past whether the our sourcing module should generate this number, or whether the vendor interface would generate it. I don't remember whether we reached any "final" agreement. Note: The requested formats are incompatible. The DELTA database is internally inconsistent, too. [Call or Serial Number]
customerPrice	double	customerUnitPrice	This is the fully-loaded unit price. We don't have this in the database, although we have discussed adding it (with concommitant date-related semantic changes to the price tiers). It is derived from lta_price_tier.unit_price using certain conversion factors (DLA Cost Recovery Rate, LTA Price Adjustment Factor) that were established in the Frankie Steward knowledge acquisition meeting. I'm also concerned about using a floating-point number for dollar currency. I'd like to explore using BigDecimal. Note: I recommend renaming this field to customerUnitPrice. [Pedro] I thought we had a way to compute the pricess? Isn't this what David is doing? If so, why do we need to raise it as an issue here? Because I was trying to create a document that described which logical data model/database fields were used to provide data for each output field. In this case, we don't have a database field that corresponds directly to the output field.
contractUnitPrice	double	orderUnitPrice	This is the DLA's unit price. We may or may not have this in the database. It might be lta_price_tier.unit price, depending upon how we want to handle the conversion factors described by Frankie Steward. I'm also concerned about using a floating-point number for dollar currency. I'd like to explore using BigDecimal. Note: I recommend renaming this to "purchaseUnitPrice".
totalValue	double	orderExtPrice	An extended price. Is this customerTotalValue or purchaseTotalValue (to use names parallel with the unit price names, above, as amended)? I'm also concerned about using a floating-point number for dollar currency. I'd like to explore using BigDecimal.
buyer	PhoneListing	???	It's possible that the buyer is "lta.person_id" (accompanied by "lta.assignment_id"), but I'm not certain about that. The current DELTA database doesn't support phone numbers, as explained below. We will incorporate support for the required functionality.
contractOfficer	PhoneListing	???	It's also possible that the contracting officer is "lta.person_id" (accompanied by

			"lta.assignlemt_id"). We definately don't have a place to store *both* the buyer and the contract officer, if they are different. We will incorporate support for the required functionality.
shipmentMethodOfPayment	String X(2)	This is a new field.	"PP" -prepaid; "PS" - seller to pay
FOBPoint	String X(2)	lta_sub_line_item.fob_pt CHAR(1)	"DE" - destination; "OR" - origin The data formats are incompatible. The Lockheed Martin documentation describes the present DELTA field as being derived from a MALT field. If it is not necessary to continue to support this exact functionality, we could widen the DELTA field to accomodate ALPINE's requested (X12-derived?) values.
payOfficeCode	String X(8)	lta_sub_line_item.pay_office VARCHAR2(2)	The formats are incompatible. As with "FOBPoint", we can alter the DELTA database to support ALPINE's requirement if the original functional requirement that lead to the present implementation is no longer operative.
CLIN	String X(4)	lta_line_item.lta_line_num VARCHAR2(4)	Note: does PartNet return CLINs and SubCLINs on vendor responses? If not, I propose to assume (for January) that PartNet LTAs are "catalog" style, with a single CLIN and SubCLIN for all items. Conceptually, a simple model is to imagine that the requested item identification, vendor's item identification, and availability codes all join the price tier. [Pedro] We better explain this to Steve in our discussion of catalog LTAs. Once the issue is resolved in that discussion, it becomes a non-issue here.
subCLIN	String X(2)	lta_sub_line_item.lta_sub_line_num VARCHAR2(2)	Note: see the CLIN notes, above.
vendorUnitOfIssue	String X(2)	lta_sub_line_item.purch_unit VARCHAR2(2)	Note: this DELTA identification is not 100% certain yet -- I need to review the Lockheed Martin documentation.
vendorPartNumber	String X(40)	lta_line_item.vendor_product VARCHAR2(24)	Note: The formats are not compatible. We can adjust the DELTA database to support ALPINE's requirement. There are still unresolved database issues inherited from the Lockheed Martin item_id/product_id strategy.
daysAR0	int int daysAR0;	lta_sub_line_item.sda lta_sub_line_item.ipg1_sda lta_sub_line_item.ipg2_sda lta_sub_line_item.ipg3_sda	It's not clear to me how the fields I cited above were intended to be used. Is the SDA field to be used when IPG-specific SDA's weren't provided? In any case, this doesn't quite match the model for catalog-based LTAs using a stocking code to determine the delivery days, as described by Frankie Steward. Note: The Lockheed Martin document is not precise on "ship" vs. "deliver" days. This particular issue required careful attention in FAST. [Pedro] I discussed this issue with Kitty already. The meaning should be "ship", that is why it is

			called ship_days_after. Isn't it? The underlying cluster data structures in the requests we receive appear to be days-until-receipt. The LTA fields appear to be days-until-shipment. Which is "daysARO": shipment or receipt? Also, as I said, Frankie talked about delivery days. [David] Needs analysis. If LTA gives ship days and we need receipt, then we need to estimate shipping days.
valueEngineering Clause	String X(1)	lta.val_eng_clause CHAR(1)	["Y" or "N"] Note: seems compatible.
fastPay	String X(1)	lta_sub_line_item.fast_pay CHAR(1)	[P, Y, N, Z, A, B]
discountPeriodCode	String X(2) default "00"	lta_payment_discnt.?	It is uncertain how to navigate in this table (there's no other reference to PAYMENT_DISCNT_NUM, but perhaps the date pair provides sufficient navigation), and I don't know how discountPeriodCode relates to DISCNT_START_DT, DISCNT_END_DT, and DISCNT_DAYS.
discountPeriodPercentageCode	String X(2)	?????	lta_payment_discnt.discnt_percent NUMBER(4,2) See the above comment on navigation in this table. Note: the requested formats are incompatible.
itemManagerORC	String X(2)	new field	
quantityVariance Code	String X(3) default "BOO"	lta_sub_line_item.qty_variance VARCHAR2(3)	Note: the formats are compatible. The Lockheed Martin logical data model document doesn't detail the codes. I recommend that we investigate the POPS codes, and determine what functional requirements, if any, are implied for the target set of contracts.
procurementMethodCode	String X(1)	lta_line_item.procmt_meth CHAR(1)	Note: the formats are compatible.
priceCompetition Code	String X(1)	lta.price_comp CHAR(1)	Note: the formats are compatible.
reportCode	String X(4)	lta.rpt_cntr_typ CHAR(1) lta.src_typ CHAR(1) lta.cntr_placmnt CHAR(1) lta.purch_procdr CHAR(1)	Concatenate
criticalityDesignatorCode	String X(1)	lta.scd_designator char(1)	Note: the formats are compatible.
remitToCode	String [a CAGE code]	new field.	
ACRN	String X(2)	new field (unless it's lta_sub_line_item.price_reason)	
socioEconomicCode	String		NEW, moved over from Vendor.java
markFor	String		NEW, dodaac code

discountPeriods	Vector		NEW
discountPeriodPercentages	Vector		NEW
vendorQuantity	int		NEW
shipmentMode	String		NEW, used in AS_ record
buyerDodaac	String		NEW, Steve said to add
accountableStationNum	String		NEW, Steve said to add, used in 850 [20]
transactionType	String		NEW: per Steve's request

Java class name: PhoneListing

Property Name	Java Type	Comes From	Comments
name	String		
phoneNumber	String		I don't think we have a place for this in the database at present! We discussed adding fields such as this in the later stages of the Lockheed Martin DELTA logical data model, but I don't think the proposals were published in report format. We will add fields to ISI's DELTA database to support the functional requirement to store and retrieve phone numbers for buyers and contract officers (and others as needed).

Java class name: MailingAddressImpl implements MailingAddress, NewMailingAddress

The Address table in the current DELTA Database is based upon a Lockheed Martin normalization that assumes a strict heirarchical relationship between city, state, zip and country. This isn't quite correct, since some cities have multiple zip codes, but some zip codes service multiple cities. We will change the Address table in the database to provide the requested functionality.

Property Name	Java Type	Comes From	Comments
addressLine1	String	???	
addressLine2	String	???	
addressLine3	String	???	
addressLine4	String	???	
addressLine5	String	???	
city	String	???	
state	String	???	
zip	String	???	
country	String = "USA"	???	

I.1.1.2 DELTA Sourcing Module: Proposal Annotations

•

To find the root or latest version of these documents, visit ISI DELTA Sourcing Data.
Please send comments and inquiries to DealMaker@isi.edu

Most "Comes From" entries consist of the database table name/link above a link to the field on that table. Annotations are documented here in several catagories:

- Attribute Created as Unique Identifier for Proposal
- Attributes Computed by Business Rules
- Attributes from LTA in DELTA Database
- Attributes from NSN Standard Item Data
- Attributes from MILSTRIP Requisition

I.1.1.2.1 Attribute Created as Unique Identifier for Proposal

Attribute Name	Java Type	Comes From	Comments
*** Proposal ID ***	String	ltaId + "-" + ltaLineNum + ltaSubLineNum + "-" + ltaPriceTierNum	Concatenation of primary key fields. The proposal ID field is a debugging aid; it should appear at the top of the Annotation display due to the asterisks. The proposal ID is unique to each LTA-based proposal.

I.1.1.2.2 Attributes Computed by Business Rules.

The rules computing these values are "build-in" in the current implementation. While they are constructed with a GUI tool, that tool is not suitable for Item Managers and Buyers (however, the three kinds of rules in the Contract Editor are suitable for these users). These rules are not contract-specific. The design assumption is that they are set up by the ICP or CBU (or on their behalf by system administrators in lieu of completing the GUI for editing and analysing these kinds of rules). Clicking on the "Comes From" link takes you to the rule that sets the value. All values examined by rules are available from attributes in this document (except for computed attributes that have not yet been computed for that proposal).

Attribute Name	Java Type	Comes From	Comments
order Unit Price	float	Compute LTA Line Item Price order Unit Price	
order Ext Price	float	Compute LTA Line Item Price order Ext Price	
customer Unit Price	float	Compute LTA Line Item Price customer Unit Price	
customer Ext Price	float	Compute LTA Line Item Price customer Ext Price	
order Qty	int	Compute LTA Line Item Quantity order Qty	
order UI	String	Compute LTA Line Item Quantity order UI	
excess in order Qty	int	Compute LTA Line Item Quantity excess in order Qty	Overbuy due to Multiple Order Quantity.
vendor Qty	int	Compute LTA Line Item UI vendor Qty	
vendor UI	String	Compute LTA Line Item UI vendor UI	
excess in vendor Qty	float	Compute LTA Line Item UI excess in vendor Qty	Overbuy due to unit conversion
excess in NSN UI Qty	float	Compute LTA Line Item UI excess in NSN UI Qty	Overbuy due to unit conversion

RDD absent	int	Compute LTA Line Item Delivery Lead Time RDD absent	Not sure how to determine this from Cluster input
computed sda	int	Compute LTA Line Item Delivery Lead Time computed sda	Need means to add on ship time to get delivery days
meets RDD	int	Compute LTA Line Item Delivery Lead Time meets RDD	Need means to add on ship time to get delivery days
delivery days ARO	int	Compute LTA Line Item Delivery Lead Time delivery days ARO	Need means to add on ship time to get delivery days
delivery days ARO less than 30	int	Compute LTA Line Item Delivery Lead Time delivery days ARO less than 30	Need means to add on ship time to get delivery days
selected Ranking Criteria	Object	Set Line Item Ranking Criteria	Interpretable sorting criteria

I.1.1.2.3 Attributes from LTA in DELTA Database.

These values are fetched from the XML representing the LTA when first requested and are cached locally for quick reference thereafter. Note that the lookup in the XML uses the values of other attributes to select the appropriate alternative stored in the LTA. For example, method-of-support (in particular) is used to choose between stock buy and direct vendor delivery values for quantity-variance/overship-percentage (and many other values). The name of the XML field is available by following the comes from link. The comments here do no reflect features implemented as part of XML support and so many are resolved in the current implementation. In our implementation, this table is obsolete because the rules access pseudo-attributes that are really just links to the XML field (using the file selectors.xml to map from the pseudo-attribute name to the access path in the XML including branches controlled by variable values such as method-of-support).

Attribute Name	Java Type	Comes From	Comments
ltaId	String	**LTA_ID**	Primary key part 1/4
ltaLineNum	String	**LTA_LINE_NUM**	Primary key part 2/4
ltaSubLineNum	String	**LTA_SUB_LINE_NUM**	Primary key part 3/4
ltaPriceTierNum	String	**LTA_PRICE_TIER_NUM**	Primary key part 4/4
LTA Price Adjustment Factor	float	NOT CURRENTLY SOURCED Defaulted to 0.85 [available in XML]	The Price Adjustment Factor converts the lta unit price to the Purchase Unit Price Per Item. It the same for all DVD items in a catalog. Other methods of support (ie, restock orders) may have a different Price Adjustment Factor. The Price Adjustment Factors may change on Period of Performance boundaries. The actual value may be positive, negative, or zero. Currently, it is a direct multiplier, however, it could be represented as a percentage change, eg, +8.25 implies a multiplier of 1.0825, -7.33 implies a multiplier of 0.9266. The Price Adjustment Factor for depot LTAs is 0.

DLA Cost Recovery Rate	float	NOT CURRENTLY SOURCED Defaulted to 1.0825 [available in XML as entered on a per-contract basis and subject to time-phased update]	The Cost Recovery Rate converts the Purchase Unit Price price to the Customer Unit Price Per Item. It is the same for all items in a catalog. It may change on a federal fiscal year boundary. The Cost Recovery Rate does not apply to depot restocking orders. I don't know whether this is supposed to be a straight multipler or a percentage change.
LTA_AWD_DT	Date	**LTA** **LTA_AWD_DT**	
LTA_PERF_START_DT	Date	**LTA** **LTA_PERF_START_DT**	
LTA_EXPIRE_DT	Date	**LTA** **LTA_EXPIRE_DT**	
LTA_MIN_DOL	float	**LTA** **LTA_MIN_DOL**	
LTA_MAX_DOL	float	**LTA** **LTA_MAX_DOL**	
LTA_DO_MIN_DOL	float	**LTA** **LTA_DO_MIN_DOL**	
LTA_DO_MAX_DOL	float	**LTA** **LTA_DO_MAX_DOL**	
CNTR_PLACMNT	String	**LTA** **CNTR_PLACMNT**	
ML_OK	String	**LTA** **ML_OK**	
VAL_ENG_CLAUSE	String	**LTA** **VAL_ENG_CLAUSE**	
LTA_REQMT_PROJ_ID	String	**LTA** **LTA_REQMT_PROJ_ID**	
LTA.VENDOR_ID	String	**LTA** **VENDOR_ID**	
CNTR_ACT	String	**LTA** **CNTR_ACT**	
CNTR_VEH_TYP	String	**LTA** **CNTR_VEH_TYP**	
FY	String	**LTA** **FY**	
GSA_FSS_CNTR_ID	String	**LTA** **GSA_FSS_CNTR_ID**	
TRANS_LABEL_IND	String	**LTA** **TRANS_LABEL_IND**	
NEW_NEGN_AUTHY	String	**LTA** **NEW_NEGN_AUTHY**	
PRICE_COMP	String	**LTA** **PRICE_COMP**	
QUOTE	String	**LTA** **QUOTE**	
RPT_CNTR_TYP	String	**LTA** **RPT_CNTR_TYP**	

SCD_DESIGNATOR	String	**LTA SCD_DESIGNATOR**	
STATE	String	**LTA STATE**	
LTA.DODAAC	String	**LTA DODAAC**	
SERIAL_NUM	String	**LTA SERIAL_NUM**	
PURCH_PROCDR	String	**LTA PURCH_PROCDR**	
LTA_STAT	String	**LTA LTA_STAT**	
SOCIO_ECON_IND	String	**LTA SOCIO_ECON_IND**	
SRC_TYP	String	**LTA SRC_TYP**	
LI.ORG_ID	String	**LTA_LINE_ITEM ORG_ID**	
LI.VENDOR_ID	String	**LTA_LINE_ITEM VENDOR_ID**	
ITEM_ID	String	**LTA_LINE_ITEM ITEM_ID**	
VENDOR_PRODUCT	String	**LTA_LINE_ITEM VENDOR_PRODUCT**	
VENDOR_PRODUCT _CODE_TYP_IND	String	**LTA_LINE_ITEM VENDOR_PRODUCT_CODE_TYP_IND**	
PROCMT_METH	String	**LTA_LINE_ITEM PROCMT_METH**	
LINE_MIN_QTY	int	**LTA_LINE_ITEM LINE_MIN_QTY**	
LINE_MIN_DOL	float	**LTA_LINE_ITEM LINE_MIN_DOL**	
LINE_MAX_QTY	int	**LTA_LINE_ITEM LINE_MAX_QTY**	
LINE_MAX_DOL	float	**LTA_LINE_ITEM LINE_MAX_DOL**	
LINE_DO_MIN_QTY	int	**LTA_LINE_ITEM LINE_DO_MIN_QTY**	
LINE_DO_MIN_DOL	float	**LTA_LINE_ITEM LINE_DO_MIN_DOL**	
LINE_DO_MAX_QTY	int	**LTA_LINE_ITEM LINE_DO_MAX_QTY**	
LINE_DO_MAX_DOL	float	**LTA_LINE_ITEM LINE_DO_MAX_DOL**	
LINE_QTY_TO_DT	int	**LTA_LINE_ITEM LINE_QTY_TO_DT**	
LINE_DOL_TO_DT	Date	**LTA_LINE_ITEM LINE_DOL_TO_DT**	
WARRANTY	String	**LTA_LINE_ITEM WARRANTY**	

TRADE_DISCNT_PERCENT	float	**LTA_LINE_ITEM TRADE_DISCNT_PERCENT**	
PURCH_UNIT	String	**LTA_SUB_LINE_ITEM PURCH_UNIT**	
QTY_VARIANCE	String	**LTA_SUB_LINE_ITEM QTY_VARIANCE**	
REVIEWER	String	**LTA_SUB_LINE_ITEM REVIEWER**	
SHIPMENT_MODE	String	**LTA_SUB_LINE_ITEM SHIPMENT_MODE**	
TYP_ANALYSIS	String	**LTA_SUB_LINE_ITEM TYP_ANALYSIS**	
IPG1_SDA	int	**LTA_SUB_LINE_ITEM IPG1_SDA**	
IPG2_SDA	int	**LTA_SUB_LINE_ITEM IPG2_SDA**	
IPG3_SDA	int	**LTA_SUB_LINE_ITEM IPG3_SDA**	
MULTIPLE_ORDER_QTY	int	**LTA_SUB_LINE_ITEM MULTIPLE_ORDER_QTY**	
SDA	int	**LTA_SUB_LINE_ITEM SDA**	
VENDOR_DELIV_DAYS	int	**LTA_SUB_LINE_ITEM VENDOR_DELIV_DAYS**	
SRP	String	**LTA_SUB_LINE_ITEM SRP**	
PRICE_REASON	String	**LTA_SUB_LINE_ITEM PRICE_REASON**	
ACTIVATION_DT	Date	**LTA_SUB_LINE_ITEM ACTIVATION_DT**	
SUB_LINE_MIN_QTY	int	**LTA_SUB_LINE_ITEM SUB_LINE_MIN_QTY**	
SUB_LINE_MIN_DOL	float	**LTA_SUB_LINE_ITEM SUB_LINE_MIN_DOL**	
SUB_LINE_MAX_QTY	int	**LTA_SUB_LINE_ITEM SUB_LINE_MAX_QTY**	
SUB_LINE_MAX_DOL	float	**LTA_SUB_LINE_ITEM SUB_LINE_MAX_DOL**	
SUB_LINE_DO_MIN_QTY	int	**LTA_SUB_LINE_ITEM SUB_LINE_DO_MIN_QTY**	
SUB_LINE_DO_MIN_DOL	float	**LTA_SUB_LINE_ITEM SUB_LINE_DO_MIN_DOL**	
SUB_LINE_DO_MAX_QTY	int	**LTA_SUB_LINE_ITEM SUB_LINE_DO_MAX_QTY**	
SUB_LINE_DO_MAX_DOL	float	**LTA_SUB_LINE_ITEM SUB_LINE_DO_MAX_DOL**	
SUB_LINE_QTY_TO_DT	int	**LTA_SUB_LINE_ITEM SUB_LINE_QTY_TO_DT**	
SUB_LINE_DOL_TO_DT	float	**LTA_SUB_LINE_ITEM SUB_LINE_DOL_TO_DT**	
METH_OF_SPT	String	**LTA_SUB_LINE_ITEM**	

		METH_OF_SPT	
ADMIN_BY	String	LTA_SUB_LINE_ITEM ADMIN_BY	
CONVERSION_FACTOR	float	LTA_SUB_LINE_ITEM CONVERSION_FACTOR	
EDI_IND	String	LTA_SUB_LINE_ITEM EDI_IND	
FAST_PAY	String	LTA_SUB_LINE_ITEM FAST_PAY	
FOB_PT	String	LTA_SUB_LINE_ITEM FOB_PT	
PRINT_IND	String	LTA_SUB_LINE_ITEM PRINT_IND	
OTHER_COST	String	LTA_SUB_LINE_ITEM OTHER_COST	
PAY_OFFICE	String	LTA_SUB_LINE_ITEM PAY_OFFICE	
POA	String	LTA_SUB_LINE_ITEM POA	
POI	String	LTA_SUB_LINE_ITEM POI	
PTIER_LOCATION_ID	String	LTA_PRICE_TIER LOCATION_ID	
PTIER_START_DT	Date	LTA_PRICE_TIER PTIER_START_DT	
PTIER_END_DT	Date	LTA_PRICE_TIER PTIER_END_DT	
BREAK_PT_LOW_QTY	int	LTA_PRICE_TIER BREAK_PT_LOW_QTY	
BREAK_PT_HI_QTY	int	LTA_PRICE_TIER BREAK_PT_HI_QTY	
LTA_UNIT_PRICE	int	LTA_PRICE_TIER UNIT_PRICE	

I.1.1.2.4 Attributes from NSN Standard Item Data.

These values are common to all proposals created to fulfill this requisition. These are currently stored as part of the LTA pending availability of a live data feed (possibly via PartNet's response for Depot Stock).

Attribute Name	Java Type	Comes From	Comments
NSN standard Unit Price in cents	int	DLSC_CATALOG_ITEM NSN_STD_UNIT_PRICE	PartNet was a timely source for this data
NSN standard Unit of Issue	String	DLSC_CATALOG_ITEM NSN_STD_UI	PartNet was a timely source for this data
NSN standard Min Order Qty	int	DLSC_CATALOG_ITEM NSN_STD_MIN_ORD_QTY	PartNet was a timely source for this data
NSN standard Multiple Order Quantity	int	DLSC_CATALOG_ITEM NSN_STD_MULT_ORD_QTY	PartNet was a timely source for this data

I.1.1.2.5 Attributes from MILSTRIP Requisition.

These values are common to all proposals created to fulfill this requisition.

Attribute Name	Java	Comes From	Comments

	Type		
customerName	String	Comes from the string returned by the method getServiceID() on the DodaacLocation returned by the requisition's getRequisitioner() method.	
customer address object	Object	Returned by the requisition's getRequisitioner() method.	This is currently the same as shipping address object in the implementation.
requisitionPriority	int	Comes from the int returned by the requisition's getPriority() method.	
request delivery in days	int	Find shipBy from the first Schedule Element (of type QuantityRangeScheduleElement) using getEndDate(). Convert this to days. Note: there's considerable confusion between ship-by, ship-before, and requested delivery date. We need an analysis of the code and customer requirements.	
request quantity	int	Find quantity from the first Schedule Element (of type QuantityRangeScheduleElement) using getEndQuantity().	
request Part (not currently an Annotation)	Object	part = taskgetDirectObject().getAsset()	
request NSN (not currently an Annotation)	String	part.getTypeIdentificationProperty(). getTypeIdentification(). substring(4,itemCode.length())	
request unit of issue	String	part.getPackageProperty().getUnitOfIssue()	
request project code (not currently an annotation)	String	getProjectCode() method on the requisition.	
request depot stock buy	boolean	NOT CURRENTLY SOURCED defaults to "NO".	
shipping address object	Object	Returned by the method getShippingAddress() on the DodaacLocation returned by the requisition's getShipToDODAAC() method.	
ZipCode	String	The zip attribute on the MailingAddress returned by the method getShippingAddress() on the DodaacLocation returned by the requisition's getShipToDODAAC() method.	
State	String	The state attribute on the MailingAddress returned by the method getShippingAddress()	

		on the DodaacLocation returned by the requisition's getShipToDODAAC() method.	
Country	String	The country attribute on the MailingAddress returned by the method getShippingAddress() on the DodaacLocation returned by the requisition's getShipToDODAAC() method.	

I.1.2 Sourcing Module Rule Template Summary

I.1.2.1 Background

This document summarizes the Rule Templates for the Source Selection Module (cluster plug-in) of the Finished Goods Inventory system. Details may be found in Template Details Background. Outstanding, future, and past issues are recorded in DELTA Rule Template Issues.
For a more detailed background explanation and for details on each rule template , see Template Details Background. Each Rule Template Name is a link to details about that rule. Rule Actions may add or modify an attribute of the Proposal or may add to the set of filtering/ranking annotations on the Proposal (e.g., Kill). The Phase indicates the relationship of the function of the rule template and the order in which it must execute with respect to other rules.
The "Status; Implementation/Testing Status; Due Date" column gives a general indication of what is happening with each rule template. First, there is an general indication which is one of "Future", "In progress", and later "Testing", "Integrating", and "Ready". This is followed by two phrases indicating separately the status of implementation and testing. Finally, the due date indicates when we expect a rule template to be ready for external testing and integration (i.e., ISI has completed implementation and done in-house testing).

I.1.2.2 Summary Round-up

Rule Templates	• 49
In progress	38
Coded	11
Defined	25
Needs Analysis	2 (+1)
Future	11 (+1)

I.1.2.3 *Proposal Generation -- Finding Substitution Parts; Looking for Depot Stock, LTAs, and Catalogs*

Status	Due	Rule Template Name	Implementation	Testing Status
Future	TBD	Approved-Substitute-OK Request AAC (Implicit)	Defined, needs data source	Needs test case data
Future*	TBD	No-Substitution Request AAC (Implicit)	Defined	Needs test case data
In progress	11/15/98	Lookup NSN in LTAs (Implicit)	Coded [CHANGE TO USE CACHE INPUT]	Needs test case data
In progress	12/21/98	Lookup NSN in Depot Stock via PartNet (Implicit)	Defined	Needs test case data
In progress	12/21/98	Lookup NSN in PartNet (Implicit)	Defined	Needs test case data
In progress	03/31/99	Lookup CAGE&PN in LTA (Implicit)	Defined	Needs test case data
In progress	03/31/99	Lookup CAGE&PN in Depot Stock via PartNet (Implicit)	Defined	Needs test case data
In progress	04/30/99	Lookup CAGE&PN in PartNet (Implicit)	Defined	Needs test case data
In progress	TBD	Propose Price Break (Implicit)	Defined, mostly coded	Needs test case data

I.1.2.4 *LTA -- Customer/Project Code Matchup*

Status	Due	Rule Template Name	Implementation	Testing Status
In progress	9/30/98	LTA Excludes Customers	Coded	Needs test case data
In progress	9/30/98	LTA Limits Customers	Coded	Needs test case data

In progress	9/30/98	LTARestricts [and limits] Customer(s)	Coded	Needs test case data
In progress	12/21/98	LTA Valid for Project Code	Defined	Needs test case data

I.1.2.5 *LTA Capabilities*

Status	Due	Rule Template Name	Implementation	Testing Status
In progress	04/30/99	LTA Does Not Honor No-Substitution Request AAC	Defined	Needs test case data
Future	TBD	LTA Does Not Support No-Backorder Request AAC	Defined	Needs test case data
In progress	TBD	LTA Does Not Support Approved-Substitute-OK Request AAC	Analysis in progress	Needs test case data
Future	TBD	LTA Does Not Support Ship-Newest-Stock Request AAC	Analysis needed	Needs test case data
In progress	04/30/99	LTA Does Not Support Must-Meet-Airborne-Personnel-Safety-Requirements Request AAC	Defined	Needs test case data
In progress	TBD	LTA Does Not Meet CONUS-Source-Only Request AAC	Defined	Needs test case data
In progress	03/31/99	LTA Does Not Support FMS	Defined	Needs test case data (Future enhanceme likely; Needs analys Due TBD)
Future	TBD	LTA Does Not Support FMS (Enhanced)	Needs analysis	Needs test case data
In progress	11/15/98	LTA Ships To Destination/Region	Defined	Needs test case data
In progress	9/30/98	LTA Header DO Total Minimum Dollars	Defined	Needs test case data
In progress	04/30/99	LTA Header DO Total Minimum Dollars (bump up qty)	Defined	Needs test case data
In progress	9/30/98	LTA Header DO Total Maximum Dollars	Defined	Needs test case data
Delete	9/30/98	LTA Header DO Line Item Minimum Dollars	Defined	Needs test case data
Delete	9/30/98	LTA Header DO Line Item Maximum Dollars	Defined	Needs test case data

In progress	11/15/98	LTA Contract in Effective Date Range	Defined	Needs test case data
Future	TBD	LTA Lifetime Minimum Dollars	Needs Analysis	Needs test case data
In progress	03/31/99	LTA Lifetime Maximum Dollars	Defined	Needs test case data

I.1.2.6 *LTA Line Item Capabilities*

Status	Due	Rule Template Name	Implementation	Testing Statu
Future	TBD	LTA Line Item Ships To Destination/Region	Defined	Needs test cas
In progress	9/30/98	LTA Line Item Supports Unit of Issue	Coded	Needs test cas
In Progress	9/30/98	LTA Line Item Convert to Unit Of Purchase	Coded	Needs test cas
In progress	9/30/98	Check minimum dollars on DO Line Item	Defined	Needs test cas
In progress	04/30/99	Check minimum dollars on DO Line Item (bump up qty)	Defined	Needs test cas
In progress	9/30/98	LTA Line Item DO Line Item Maximum Dollars	Defined	Needs test cas
In progress	9/30/98	Check minimum quantity on DO Line Item	Defined	Needs test cas
In progress	04/30/99	Check minimum quantity on DO Line Item (bump up qty)	Defined	Needs test cas
In progress	11/15/98	LTA Line Item DO Line Item Maximum Quantity	Defined	Needs test cas
In progress	9/30/98	Compute LTA Line Item Delivery Lead Time	Coded	Needs test cas
In progress	9/30/98	Compute LTA Line Item UI	Coded	Needs test cas
In progress	9/30/98	Compute LTA Line Item Quantity	Coded	Needs test cas
In progress	9/30/98	Compute LTA Line Item Price	Coded	Needs test cas
Future	TBD	LTA Line Item DO Rejected For Requisition	Needs ALP Vendor Cluster API and ISI Design	Needs test cas

I.1.2.7 *Proposal Adjustments, Priority, and Ranking*

Status	Due	Rule Template Name	Implementatio	Testing Status

			n	
Future	TBD	Add Depot Backorder Lead Time	Needs Depot Data API and ISI Design	Needs test case data
In progress	04/30/99	Current Inventory Fails No-Backorder Request AAC	Defined	Needs test case data
Future	TBD	Inform Vendor No-Backorder Request AAC	Needs Vendor Cluster API and ISI Design	Needs test case data
In progress	03/31/99	Request AAC Allows Up-To-5-Dollars-More	Defined	Needs test case data
In progress	03/31/99	Allow Dollars More	Defined	Needs test case data
In progress	TBD	Proposal Does Not Meet Require-for-Free Request AAC	Analysis in progress	Needs test case data
In progress	11/15/98	Lead Time Meets Priority	Coded (incorp. RDD)	Needs test case data
In progress	9/30/98	Price/Priority/LeadTime Rating	Coded	Needs test case data

I.1.3 Rule Template Details

I.1.3.1 Background

This document serves to enumerate the functionality of rules in the DELTA Source Selection Module (Cluster Plug-in). For a summary listing of rules and their current status and due date, see DELTA rule Template Summary. Current, future, and past issues are recorded in DELTA Rule Template Issues.
This document indicates for each Rule Template its Phase and Action. The Phase indicates the relationship of the function of the rule template and the order in which it must execute with respect to other rules. Currently, there are the phases "Elaborate", "Filter", and "Rank". However, more work is needed to further refine rule ordering. Already, some elaboration and filter rules must run out of simple order. It is also possible for ad-hoc conditions on a rule instance to introduce an

ordering dependency that is not present based on the rule template. Thus, we may require Rule Analysis to compute and store a partial ordering over all rule instances to be used at run time. Rule Actions may add or modify an attribute of the Proposal or may add to the set of filtering/ranking annotations on the Proposal (e.g., Kill).
The current plan calls for each rule to be conceptually applied to each proposal. Fortunately, most specialized rules will not be applicable. The applicable specialized rules will be found by looking them up according to LTA ID, Customer, or other significant attribute (found on the Proposal). General rules, such as those defined at the ICP level for all LTAs, will be applied to every proposal (all general rules for a modest set of proposals should take much less compute time than, say, a database query; to be verified).
In this document, Rules are grouped according to when they run and then their key data element(s), currently one of:: Proposal Generation; LTA and Customer/Project Matchup; LTA Constraints; LTA Line Item Constraints; and Proposal Ranking (elaborations computed for the Proposal to support ranking). A Proposal represents the matchup of a Line Item on an LTA to a Requistion and has named attributes added for additional information. There may be zero or many proposals generated for a requisition. The process of creating proposals implements the generic implicit rule **LTA Supplies Item** -- that is, every proposal is assured to contain a source of supply that has some indication that the part will be available. The supplier may be found by searching in the DELTA LTA database or in PartNet for the requisition's NSN or for its CAGE and Part Number (CAGE&PN). Once proposals have been generated, they may be elaborated (have attributes added or modified). One attribute is the set of annotations that control a final filtering and ranking phase. For example, a "Kill" annotation prevents using that proposal for a Deliver Order (DO). The annotation "Fastest" causes this proposal to be ranked between proposals with longer and shorter delivery times (within an equivalence grouping determined by higher-precedence ranking factors). If all proposals are killed, then the requisition will be sent out of the Sourcing module for alternate processing (SAMMS or manual).
In the future, this concept of a proposal may become a LineItemProposal. Then, an OrderProposal would represent combinations of LineItemProposals to (1) allow rolling together MILSTRIP orders that are identical except for the item and quantity and (2) to support DODAAC-fund/ship-to shopping cart orders from E-MALL. Significant additional analysis and design is necessary to verify this representation's viability given the computational complexity of optimizing the possible combinations of Line Items and LTAs/Vendors.
Rules are intended both to provide the most common functionality for filtering and ranking proposals and also to support exceptional case processing (without resorting to programming). Rule templates specify certain parameters and implicitly access certain database values. In addition, rules may have "ad-hoc" conditions controlling when they apply. The ad-hoc conditions appropriate to each rule template remain to be analysed.
A rule template (built by programmers) gives the non-programmer (Item Manager, Contracting Officer, Buyer) the ability to create rules that control the filtering and ranking of proposals. This document enumerates the rule templates planned for the

first release of the DELTA Source Selection Plug-in. The planned research for the third year of effort includes the ability for non-programmers to create or modify rule templates to some degree.
Each rule template mandates that instances of the rule include specified fields such as LTA or Customer. Upon creating a rule, these fields will be available for fill-in. In addition, additional selected fields may be added as conditions controlling when the rule will be applicable to a proposal. These condition fields may include the Ship-to region and Priority. These optional condition fields are not yet listed for each rule template. The goal is to provide flexibility without unmanageable complexity. The non-programmers involved, for example, Item Managers and Contracting Officers, must be able to understand the impact of the rules they specify. They must have a processing model in mind for this. The conditions on rules -- especially very similar rules -- must be clear in their minds as to when the rule will apply. This greatly complicates the design of the ad-hoc conditions. One case to consider is when a small number of exceptional Line Items violate the general condition that applies to many more other Line Items on an LTA. Even more confusing might be when several kinds of exceptions need to apply over different subsets of the Line Items, say limitation to named customerss and named customer exclusions. The user interface (GUI) on the LTA/Rule Editor needs to be accessible to other people to understand these rules once they have been entered.

I.1.3.2 ICP/CBU Defaults and Enforcements

This category of rules represents a future capability for establishing default rules and for managerially controlling the behavior of a rules created be members of an organization. Defaults are effective when no explicit override is present. An example default is to use the normal lead time computation unless another rule applies that specifies an alternative. Enforcements limit the behavior of explicit rules. An example enforcement is to prevent shipment to OCONUS to non-military customers for a particular NSN (say, powerful encryption software). ICP/CBU Defaults and Enforcements require the rule administration facility that is not scheduled to be built for some time -- therefore, further specification in this area is to be accomplished later. For each rule template, this document provides an initial indication of the Default and Enforcement rules that might be needed or desired. [It remains to schedule implementation of these rule templates and create an implementation and testing tracking mechanism (which will probably be manual).]
Return to Template Index Page

I.1.3.3 *Proposal Generation -- Finding Substitution Parts; Looking for Depot Stock, LTAs, and Catalogs*

"Implicit" means these rules are to be implemented in code and will not appear in the LTA/Rule Editor. They are documented here for completeness.

Approved-Substitute-OK Request AAC (Implicit)

Description: If substitutions are allowed, then generate proposals for them. Also, we may wish to add preference toward a "supercedes" substitute.

Phase: Generation
Action: Create proposal

No-Substitution Request AAC (Implicit)

Description: Deals with the processing of the Acquisition Advice (AAC). If substitutions are allowed, a data source for approved substitutes may be queried to generate proposal(s) with the substitute part identifier(s). The resulting proposals will receive the same processing as those for the original part identifier (elaboration, filtering, and ranking). The No-Substitution AAC prevents Sourcing from searching for substitution parts (and is not needed until substitution is implemented). The rule LTA Does Not Honor No-Substitution Request AAC kills proposals if the vendor cannot receive or ignores this AAC option.
Phase: Generation
Action: Create proposal
Test case 1: requisition(...NSN with substitutions for which supplier(s) exist, AAC allows substitutions)
show substitute NSNs appear in proposals
Test case 2: requisition(...NSN with substitutions for which supplier(s) exist, AAC does not allow substitutions)
show substitute NSNs do not appear in proposals
Return to Template Index Page

Lookup NSN in LTAs (Implicit)

Description: When the incoming request includes an indication of an LTA-supplied part, generate proposal(s) and annotate with data fields.
Phase: Generation
Action: Create proposal

Lookup CAGE&PN in LTA (Implicit)

Description: When the incoming request includes an indication of an LTA-supplied part, generate proposal(s) and annotate with data fields.
Phase: Generation
Action: Create proposal
The DELTA LTA database is queried to generate proposal(s) that pair the requisition with a potential source of supply.
Test case 1: requisition(...NSN supplied by a DELTA LTA)
show proposal(s) generated for DELTA LTA
Test case 2: requisition(...CAGE&PN supplied by a DELTA LTA)
show proposal(s) generated for DELTA LTA
Return to Template Index Page

Lookup NSN in Depot Stock via PartNet (Implicit)

Description: When the incoming request includes an indication of a Depot-supplied part, query Partnet, generate proposal(s), and annotate with data fields.
Phase: Generation
Action: Create proposal

Test case: requisition(...NSN supplied by Depot)
show proposal generated for Depot

Lookup CAGE&PN in Depot Stock via PartNet (Implicit)

Description: When the incoming request includes an indication of a Depot-supplied part, query Partnet, generate proposal(s), and annotate with data fields.
Phase: Generation
Action: Create proposal
Test case: requisition(...CAGE&PN supplied by Depot)
show proposal generated for Depot

Lookup NSN in PartNet (Implicit)

Description: When the incoming request includes an indication of a Vendor-supplied part, query Partnet, generate proposal(s), and annotate with data fields.
Phase: Generation
Action: Create proposal
Test case: requisition(...NSN supplied by a PartNet vendor)
show proposal(s) generated

Lookup CAGE&PN in PartNet (Implicit)

Description: When the incoming request includes an indication of a Vendor-supplied part, query Partnet, generate proposal(s), and annotate with data fields.
Phase: Generation
Action: Create proposal
Test case: requisition(...CAGE&PN supplied by a PartNet vendor)
show proposal(s) generated
Return to Template Index Page

Propose Price Break (Implicit)

Description: It is possible that buying a slightly higher quantity will result in a lower total price. By creating a proposal with a Quantity putting it into the next price tier, this possibility will be filtered and ranked alongside other options. The change is price will be tracked to support Dollars-More handling.
Phase: Generation
Action: Create proposal
Test case 1: requisition(...NSN supplied by some LTA with price breaks, quantity just below a price break)
show additional proposal generated for buying at the price break
Return to Template Index Page

LTA Excludes Customers[1]

Exclusion rules are also definable in the Contract Editor. Such rules apply only to proposals for using that contract. See the Contract Editor screens to understand what proposal annotations may be tested in the conditions (LHS) part of those exclusion rules.

Description: Explicitly name customers that may not place orders against this LTA.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Proposal.LTA; Requisition.Customer
ICP/CBU Default: none
ICP/CBU Enforcement: none
Sample rule LTA SPD00111111 excludes Customers Army and Navy when Priority is IPG1
Test case 1 requisition(customer=Fort H, item supplied by LTA SPD00111111, priority=1)
show proposal for SPD00111111 killed
Test case 2 requisition(customer=Air force, item supplied by LTA SPD00111111, priority=1)
show rule does not apply
Return to Template Index Page

LTA Limits Customers[2]

"Serves only" rules are also definable in the Contract Editor. Such rules apply only to proposals for using that contract. See the Contract Editor screens to understand what proposal annotations may be tested in the conditions (LHS) part of those rules.

Description: Explicitly name customers that may place orders against this LTA.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Proposal.LTA; Requisition.Customer
ICP/CBU Default: none
ICP/CBU Enforcement: none
Sample rule: Limit LTA SPD020097D9387 -- General Hardware to service customer(s) Army
Test case 1: requisition(customer=Fort H, item supplied by LTA SPD020097D9387)
show rule does not kill proposal for this LTA
Test case 2: requisition(customer=Wright-Patterson AFB, item supplied by LTA SPD020097D9387)
show rule kills proposal for this LTA
Return to Template Index Page

[1] *The documentation of this rule is not fully up to date.*
[2] *The documentation of this rule is not fully up to date.*

LTA Restricts [and limits] Customer(s)[3]

"Must use" rules are also definable in the Contract Editor. Such rules apply only to proposals for using that contract. See the Contract Editor screens to understand what proposal annotations may be tested in the conditions (LHS) part of those rules.
Description: This LTA must be used by explicitly named customer(s) for the parts offered by this LTA. Optionally, the LTA may be limited to these named customers.
Phase: Rank [&Filter]
Action: Prefer [& Kill] [or Must-Use for end-user rules specified in the Contract Editor]
Attributes Modified:
Attributes Accessed: Proposal.LTA; Requisition.Customer
ICP/CBU Default: none
ICP/CBU Enforcement: none
Sample rule 1: LTA SP020094D0006 -- CTS CORPORATION is prime for Fort Huachuca
Sample rule 2: LTA SP020094D0006 -- CTS CORPORATION is prime for and limited to Fort Huachuca
Test case 1: requisition(customer=Fort Huachuca, item supplied by LTA SPD020094D0006)
rule 1, show rule does not kill proposal for this LTA
rule 2, show rule does not kill proposal for this LTA
Test case 2: requisition(customer=Fort Bragg, item supplied by LTA SPD020097D9387)
rule 1, show rule does not apply
rule 2, show rule kills proposal for this LTA
Return to Template Index Page

LTA Valid for Project Code

The current system supplies this rule in the form of allowing Project Code to be a condition on rules defined in the Contract Editor.
Description: Project codes serve a primary purpose of taging transactions for analysis "by project". However, they may also be used to achieve special routing and handling of requests and requested items. For example, certain Navy DODAACs and project codes are funneled to special contracts.
Phase: Filter
Action: Prefer
Attributes Modified:
Attributes Accessed: Proposal.LTA; Reqisition.ProjectCode
ICP/CBU Default: none (no database fields currently defined to support this)
ICP/CBU Enforcement: none
Sample rule: LTA SP020094D0007 -- CTS CORPORATION is valid for and limited to Project Code COPAD.

[3] *The documentation of this rule is not fully up to date.*

Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007, no project code)
show rule kills proposal for this LTA
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007, project code COPAD)
show rule does not kill proposal
Test case 3: requisition(customer=Navy, item supplied by LTA SP020094D0007, project code XXXXX)
show rule kills proposal for this LTA
Return to Template Index Page

I.1.3.5 LTA Capabilities

Many LTA-specific rules are only specifiable in the Contract Editor as Exclusion or Serves-only rules with appropos conditions.

LTA Does Not Honor No-Substitution Request AAC

Description: In certain commodities (example: drugs) vendors might perform item substitution regardless of a "do not substitute" flag in an EDI requisition. In these cases, the requisition needs to be processed manually by the vendor. Because DELTA cannot arrange for this, proposals to such companies are killed when the AAC calls for no substitutions. We expect this rule to go unused because proposals are based on contracts or current catalogs (and we expect PartNet to inform the Sourcing Module if the item is discontinued at that vendor [-- does this need its own rule?]). The rule No-Substitution Request AAC (Implicit) prevents the generation of proposals with substitutions in them.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.AAC
ICP/CBU Default: none
ICP/CBU Enforcement: none
Sample rule: LTA SP020094D0007 -- CTS CORPORATION does not honor "no substitution" requests.
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007, AAC does not imply "do not substitute")
show rule does not kill proposal
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007, AAC implies "do not substitute")
show rule kills proposal for this LTA
Return to Template Index Page

LTA Does Not Support No-Backorder Request AAC

Description: The No-Backorder AAC option indicates a desire to fill the requisition from current stock. This rule template enables the prevention of orders to a vendor when this AAC option is present. The rules Current Inventory Fails No-Backorder

Request AAC and Inform Vendor No-Backorder Request AAC handle the No-Backorder AAC option generically, so flagging a particular LTA using this rule is optional.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.AAC
ICP/CBU Default: none
ICP/CBU Enforcement: none
Sample rule: LTA SP020094D0007 -- CTS CORPORATION does not support "no backorder" requests.
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007, AAC does not imply "no backorder")
show rule does not kill proposal
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007, AAC implies "no backorder")
show rule kills proposal for this LTA (unless it is decided to rely on informing the Vendor Cluster to monitor for a backorder response from vendor)
Return to Template Index Page

LTA Does Not Support Ship-Newest-Stock Request AAC

Description: Further analysis needed. This is a rule template dealing with the processing of an Acquisition Advice (AAC). This may be an outmoded AAC code from the days when DLA stockpiled inventories.
Phase: Filter
Action: Kill
Return to Template Index Page

LTA Does Not Support Must-Meet-Airborne-Personnel-Safety-Requirements Request AAC

Description: For NSN requisitions and contractual LTAs that enumerate their Line Items with NSNs, the part supplied is expected to implicitly meet this requirement. In the other cases of finding parts by CAGE&PN or in a catalog (even a catalog-based DELTA LTA), meeting this requirement cannot be determined by the sourcing module, so the proposal is killed (causing SAMMS or manual handling of the requisition if all proposals are killed).
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.AAC
ICP/CBU Default: This rule is expected to be an ICP default rule
ICP/CBU Enforcement: none
Sample rule: LTA SP020094D0007 -- CTS CORPORATION does not support Must-Meet-Airborne-Personnel-Safety-Requirements Request AAC

Test case 1: requisition(customer=Navy, AAC not Must-Meet-Airborne-Personnel-Safety-Requirements, item supplied by LTA SP020094D0007)
show rule does not kill proposal for this LTA
Test case 2: requisition(customer=Navy, AAC is Must-Meet-Airborne-Personnel-Safety-Requirements, item by NSN supplied by LTA SP020094D0007)
show rule does not kill proposal for this LTA
Test case 3: requisition(customer=Navy, AAC is Must-Meet-Airborne-Personnel-Safety-Requirements, item by CAGE&PN supplied by LTA SP020094D0007)
show rule kills proposal for this LTA
Return to Template Index Page

LTA Does Not Meet CONUS-Source-Only Request AAC

Description: An Acquisition Advice (AAC) option is to require ordering from a CONUS source only. This rule verifies that the place of performance on the LTA meets this requirement. Most or all LTAs will be within the 50 states plus Puerto Rico. (ISI: assess implication)
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.AAC; Proposal.LTA.PlaceOfPerformance (ISI: verify this field is in database; include in LTA Editor)
ICP/CBU Default: This rule is expected to be an ICP default rule
ICP/CBU Enforcement: none
Sample rule: LTAs with PlaceOfPerformance outside CONUS do not meet CONUS-Source-Only Request AAC
Test case 1: requisition(..., AAC not CONUS-Source-Only, item supplied by LTA SP020094D0007)
show rule does not kill proposal for this LTA (or kill any other proposal)
Test case 2: requisition(..., AAC is CONUS-Source-Only, item supplied by LTA SP020094D0007)
show rule kills proposal for this LTA only when the LTA PlaceOfPerformance is not CONUS
Return to Template Index Page

LTA Does Not Support FMS

Description: More analysis is needed to properly implement FMS orders in FGI, so at this time all such requests will be kicked out for SAMMS or manual handling by killing all proposals with this rule. FMS/MAP requisitions may require special processing, such as MAPAD lookup for freight forwarding information. Unless the vendor guarantees FMS/MAP processing, FMS/MAP requisitions should be processed manually.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.RequestorDODAAC

ICP/CBU Default: No LTA supports FMS (i.e., kill all proposals if FMS -- for now, FGI will not order for FMS)
ICP/CBU Enforcement: none
Sample rule: No LTA Supports FMS (the ICP Default until further analysis is done).
Test case 1: requisition(..., not FMS,...)
show rule does not kill proposal
Test case 2: requisition(..., FMS, ...)
show rule kills proposal for this LTA (and all other proposals as well)
Return to Template Index Page

LTA Ships To Destination/Region

Description: Certain LTAs may be limited to delivering to (or not to) certain regions. Although this can be expressed in the subline item-specific pricing information, it may be desirable, as a matter of policy, or of contractual obligation, to limit a particular LTA's delivery area. DEALMAKER's database currently supports a tabular representation of non-overlapping geographical regions based on states and zip codes. An editor allowing non-programmers to create and modify multiple named custom regional divisions of the continental U.S. is planned.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.ShipToDODAAC
ICP/CBU Default: none (handled by default at the LTA Line Item level by rule LTA Line Item Ships To Destination/Region)
ICP/CBU Enforcement: none
Sample rule: LTA SP020094D0007 -- CTS CORPORATION does not ship to "Heavy Equipment" region "SouthWest".
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007, shipto=Seattle WA)
show rule does not kill proposal
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007, shipto=San Diego CA)
show rule kills proposal for this LTA
Return to Template Index Page

LTA Header DO Total Minimum Dollars

Description: If a proposal does not meet this minimum, increase the quantity so that it does and track the increase for Dollars-More handling. Until we handle multi-line-item orders, there is no decision as to which item to increase.
Phase: Elaborate
Action: Boost Qty & Dollars-more

LTA Header DO Total Maximum Dollars

Description: An LTA can have constraints on the total purchase price on a Delivery Order (DO). These are recorded in database fields with these rules for enforcement

(or exceptions). For now, DEALMAKER is assuming single line item DOs to match the limitation of MILSTRIP. If the minimum is not met, then there is an increase in Proposal.Quantity allowable in certain cases (such as the appropriate AAC code).
Phase: Filter
Action: Kill
Attributes Modified: Proposal.QuantityIncrement
Attributes Accessed: Proposal.LTA.TotalMinDODollars / Proposal.LTA.TotalMaxDODollars
ICP/CBU Default: This rule is likely to be an ICP default using LTA Line Item database attributes
ICP/CBU Enforcement: potential to have a per-order limit for the entire ICP or CBU
Sample rule 1: LTA SPD0200094D008 -- General Plumbing requires DO Total Minimum of $100.00
Sample rule 2: LTA SPD0200094D008 -- General Plumbing requires DO Total Maximum of $500,000.00
Test case 1: requisition(..., item supplied by SPD0200094D008, quantity 100)
show that rule 1 increases the quantity if the item cost is under $1 to meet $100 minimum by adding to the Proposal.QuantityIncrement
show that rule 2 kills the proposal if the (Quantity + QuantityIncrement) multiplied by the item cost exceeds $500,000
Return to Template Index Page

LTA Header DO Line Item Minimum Dollars

Description: If a proposal does not meet this minimum, increase the quantity so that it does and track the increase for Dollars-More handling.
Phase: Elaborate
Action: Boost Qty & Dollars-more

LTA Header DO Line Item Maximum Dollars

Description: An LTA can have constraints on the purchase price for the given quantity of every Line Item on Delivery Order (DO). These are recorded in database fields with these rules for enforcement (or exceptions). If the minimum is not met, then there is an increase in Proposal.Quantity allowable in certain cases (such as the appropriate AAC code).
Phase: Filter
Action: Kill
Attributes Modified: Proposal.QuantityIncrement
Attributes Accessed: Proposal.LTA.LineItemMinDODollars / Proposal.LTA.LineItemMaxDODollars
ICP/CBU Default: This rule is likely to be an ICP default using LTA Line Item database attributes
ICP/CBU Enforcement: none
Sample rule 1: LTA SPD0200094D008 -- General Plumbing requires every DO Line Item Minimum of $100.00

Sample rule 2: LTA SPD0200094D008 -- General Plumbing requires every DO Line Item Maximum of $500,000.00
Test case 1: requisition(..., item supplied by SPD0200094D008, quantity 100)
show that rule 1 increases the quantity if the item cost is under $1 to meet $100 minimum by adding to the Proposal.QuantityIncrement
show that rule 2 kills the proposal if the (Quantity + QuantityIncrement) multiplied by the item cost exceeds $500,000
Return to Template Index Page

LTA Contract in Effective Date Range

Description: The LTA is in a active period of performance for this requisition.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Proposal.LTA.ExpirationDate; Proposal.LTA.StartDate
ICP/CBU Default: this rule is likely to be implemented as an ICP default rule only that looks at the LTA database for dates
ICP/CBU Enforcement: none
Sample rule: LTA SPD0200094D008 -- General Plumbing is not effective after December 31, 1999
Test case 1: requisition(customer=Navy, item supplied by LTA SPD020094D0008, RDD=December 10, 1999)
show rule does not kill proposal for this LTA
Test case 2: requisition(customer=Navy, item supplied by LTA SPD020094D0008, RDD=December 10, 2000)
show rule kills proposal for this LTA
Return to Template Index Page

LTA Lifetime Minimum Dollars

Description: (Future; needs much more analysis.) An LTA can specify an amount of business in dollars that the vendor is assured to receive. If this amount is not spent during the lifetime of the LTA, the government is libel to the vendor for the balance of the minimum. This rule template represents the need for DELTA to analyse the needs of the ICPs to automate a policy for directing orders so as to meet minimums on all suppliers before spending over minimum with some suppliers. Note that the customer is going to pay the NSN standard unit price and so DLA will want to be sure that it optimizes the payout of unspent minimums against less competitive prices. It may be desirable to select other sources anyway, so this feeds into the ranking algorithm rather than forcing a selection (and multiple sources may be below minimum). The amount of boost may be proportional to the time remaining to meet the minimum and the dollars left to meet the minimum. The ranking may need to consider how non-competitive this LTA is compared to other proposals and compare this with the other factors.
Phase: none
Action: n/a

Attributes Modified: Proposal.BoostRanking (ISI: consider alternative implementations)
Attributes Accessed: Proposal.LTA.LifetimeMinimumDollars; Sentinels[LTA].LifetimeDollarsSpent
ICP/CBU Default: This rule is likely to be an ICP default using LTA Line Item database attributes
ICP/CBU Enforcement: none
Sample rule 1: ICP default rule to boost chances for any LTA not meeting its lifetime minimum
Test case 1: requisition(..., item supplied by an LTA not meeting lifetime minimum) show that rule 1 sets Proposal.BoostRanking for the proposal for the LTA not meeting lifetime minimum (to add weight to selecting this LTA)
Return to Template Index Page

LTA Lifetime Maximum Dollars

Description: An LTA can have a limit on the amount of business in dollars that the vendor is contracted to provide. While an ICP default rule will implement ICP policy, it may be allowed to override this rule for particular LTAs under specified circumstances, such as Priority in IPG1. (Note from ISI: The mechanism for ICP/CBU default rules being overridden has not been designed as of 7/10/98.) (ISI: The upkeep of the Sentinels[LTA].LifetimeDollarsSpent has not been designed.)
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Proposal.LTA.LifetimeMaximumDollars; Sentinels[LTA].LifetimeDollarsSpent
ICP/CBU Default: This rule is likely to be an ICP default using LTA database attributes and possibly requisition priority
ICP/CBU Enforcement: none
Sample rule 1: ICP default rule to kill all proposals that would put the LTA over its lifetime maximum
Test case 1: requisition(..., item supplied by LTA where this order puts it over lifetime maximum)
show that rule 1 kills the proposal for this LTA if the (Quantity + QuantityIncrement) multiplied by the item puts the LTA over its lifetime maximum
Return to Template Index Page

I.1.3.6 LTA Line Item Capabilities

LTA Line Item Ships To Destination/Region

Description: (Future; not needed at line item level at this time.) Certain LTA Line Items may be limited to delivering to (or not to) certain regions. Although this can be expressed in the subline item-specific pricing information, it may be desirable, as a matter of policy, or of contractual obligation, to limit a particular LTA Line Item's delivery area. DEALMAKER's database currently supports a tabular representation

of non-overlapping geographical regions based on states and zip codes. An editor allowing non-programmers to create and modify multiple named custom regional divisions of the continental U.S. is planned.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.ShipToDODAAC
ICP/CBU Default: This rule is likely to be an ICP default using LTA Line Item database attributes
ICP/CBU Enforcement: none
Sample rule: LTA SP020094D0007 -- CTS CORPORATION Line Item 1 does not ship to "Heavy Equipment" region "SouthWest".
Test case 1: requisition(customer=Navy, Line Item 1 supplied by LTA SP020094D0007, shipto=Seattle WA)
show rule does not kill proposal
Test case 2: requisition(customer=Navy, Line Item 1 supplied by LTA SP020094D0007, shipto=San Diego CA)
show rule kills proposal for this LTA and Line Item
Return to Template Index Page

LTA Line Item Supports Unit of Issue

Description: Requisitions specify a Unit of Issue in conjunction with the Quantity. When the Requitision specifies an NSN, the Unit of Issue must match the NSN's Standard Unit of Issue or the LTA's Unit of Purchase (ISI: ???). The LTA may specify a conversion factor to convert the NSN Standard Unit of Issue to the Unit of Purchase for use when placing a DO. When a Requisition specifies a CAGE and Part Number, then the Unit of Issue specified in the Requisition must match the Unit of Purchase in the LTA (the requisitioner is likely unaware of the NSN Standard Unit of Issue, and its meaning). No confident conversions are possible for other vendor-specific Unit of Issue codes. Note that this rule will likely be global for each ICP so that it need not be specified for each LTA and Line Item. It is included in this list for completeness and to introduce the idea that a particular LTA might be more liberal by having overriding rules for specific cases where conversion can be specified (this interacts with the rule Convert to LTA Line Item Unit Of Purchase).
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.UnitOfIssue; Proposal.LTALineItem.UnitOfPurchase; Requisition.NSN.StandardUnitOfIssue
ICP/CBU Default: This rule is likely to be only an ICP default using LTA Line Item database attributes
ICP/CBU Enforcement: none
Sample rule: LTA SP020094D0007 -- CTS CORPORATION Line Item 1 "ball bearing" supports only NSN Standard Unit of Issue ("EA")
Test case 1: requisition(..., NSN 12345678901234 "ball bearing", Unit of Issue "EA")
show rule does not kill proposal for its LTA

Test case 2: requisition(..., NSN 12345678901234 "ball bearing", Unit of Issue "PKG")
show rule kills proposal for its LTA
Test case 3: requisition(..., CAGE for CTS CORPORATION, Part Number for "ball bearing", Unit of Issue "EA")
show rule does not kill proposal for its LTA
Test case 4: requisition(..., CAGE for CTS CORPORATION, Part Number for "ball bearing", Unit of Issue "PKG")
show rule kills proposal for its LTA
Return to Template Index Page

LTA Line Item Convert to Unit Of Purchase

Description: The LTA may specify a conversion factor to convert the NSN Standard Unit of Issue for use when placing a DO. This rule makes the conditions for applying this conversion factor explicit. Note that this rule will likely be global for each ICP so that it need not be specified for each LTA and Line Item. It is included in this list for completeness and to introduce the idea that a particular LTA might be more liberal by having rules for specific cases where conversion can be performed. If the conversion has a remainder (must order more than requested), then an increase in quantity is called for (with rule Request AAC Allows Up-To-5-Dollars-More validating the amount allowed).
Phase: Elaboration or Filter
Action: Set VendorUI, VendorQty or Kill
Attributes Modified: Proposal.QuantityIncrement; Proposal.UnitOfIssue (ISI: does QuantityIncrement work here or conflict with other uses?)
Attributes Accessed: Proposal.LTALineItem.ConversionFactor; Requisition.UnitOfIssue; Proposal.LTALineItem.UnitOfPurchase; Requisition.NSN.StandardUnitOfIssue
ICP/CBU Default: This rule is likely to be only an ICP default using LTA Line Item database attributes
ICP/CBU Enforcement: none
Sample rule: LTA SP020094D0007 -- CTS CORPORATION Line Item 1 "ball bearing" can supply Unit of Issue "PKG" by multiplying Unit Of Purchase "EA" by 4
Test case 1: requisition(..., NSN 12345678901234 "ball bearing", Unit of Issue "EA")
show rule does not apply
Test case 2: requisition(..., NSN 12345678901234 "ball bearing", Unit of Issue "PKG")
show rule supplies a new Unit of Issue "EA" with Quantity computed by multiplying the requisition Quantity by 4
Note that a generic rule LTA Line Item Supports Unit Of Purchase interacts with this rule (and might kill the proposal) (ISI: analyse how this will be handled; possibly by combining the kill and conversion rules or through overrides?)
Test case 3: requisition(..., CAGE for CTS CORPORATION, Part Number for "ball bearing", Unit of Issue "EA")
show rule does not apply

Test case 4: requisition(..., CAGE for CTS CORPORATION, Part Number for "ball bearing", Unit of Issue "PKG")
show rule supplies a new Unit of Issue "EA" with Quantity computed by multiplying the requisition Quantity by 4
Note that a generic rule LTA Line Item Supports Unit Of Purchase interacts with this rule (and might kill the proposal) (ISI: analyse how this will be handled; possibly by combining the kill and conversion rules or through overrides?)
Return to Template Index Page

LTA Line Item DO Line Item Maximum Dollars
LTA Line Item DO Line Item Maximum Quantity

Description: If a maximum is exceeded, kill the proposal.
Phase: Filter
Action: Kill
Attributes Modified: Proposal.QuantityIncrement (possible adjustment to meet a minimum)
Attributes Accessed: LINE_DO_MIN_DOL, LINE_DO_MIN_QTY, Requisition.AAC (?)
ICP/CBU Default: This rule is likely to be an ICP default using the database LTA Line Item attribute values
ICP/CBU Enforcement: none
Sample rule 1: LTA SPD0200094D008 -- General Plumbing Line Item 1 requires DO Line Item Maximum of $50,000.00
Sample rule 2: LTA SPD0200094D008 -- General Plumbing Line Item 1 requires DO Line Item Maximum Quantity of 7,500
Test case 1: requisition(..., Line Item 1 supplied by SPD0200094D008, quantity 100) show that rule 1 kills the proposal if the (Quantity + QuantityIncrement) multiplied by the item cost exceeds $50,000
Test case 2: requisition(..., Line Item 1 supplied by SPD0200094D008, quantity X) show that rule 2 kills the proposal if the Quantity exceeds 7,500

Check minimum dollars on DO Line Item
Check minimum quantity on DO Line Item

Description: If a proposal does not meet this minimum, increase the quantity so that it does and track the increase for Dollars-More handling. Each Line Item on an LTA can have constraints on the quantity and extended price for that Line Item on a Delivery Order (DO). These are recorded in database fields with these rules for enforcement (or exceptions). If the minimum is not met, then there may be an increase in quantity allowable (with rule Request AAC Allows Up-To-5-Dollars-More verifying the amount allowed).
Phase: Elaborate (after initial order quanity/price determination)
Action: Kill proposal or Boost Qty & Dollars-more
Attributes Modified: Proposal.QuantityIncrement (possible adjustment to meet a minimum)
Attributes Accessed: LINE_DO_MIN_DOL, LINE_DO_MIN_QTY, Requisition.AAC (?)

ICP/CBU Default: This rule is likely to be an ICP default using the database LTA Line Item attribute values
ICP/CBU Enforcement: none
Sample rule 1: LTA SPD0200094D008 -- General Plumbing Line Item 1 requires DO Line Item Minimum of $30.00
Sample rule 2: LTA SPD0200094D008 -- General Plumbing Line Item 1 requires DO Line Item Minimum Quantity of 100
Test case 1: requisition(..., Line Item 1 supplied by SPD0200094D008, quantity 100) show that rule 1 increases the quantity if the item cost is under $0.30 to meet $30 minimum by adding to the Proposal.QuantityIncrement
Test case 2: requisition(..., Line Item 1 supplied by SPD0200094D008, quantity X) show that rule 3 increases the quantity if necessary to meet the minimum by adding to the Proposal.QuantityIncrement
Return to Template Index Page

Compute LTA Line Item Delivery Lead Time

Description: **Priority and ranking** requires the delivery lead time. This computation may require variation based on conditions that may be examined by rules giving non-programmers some flexibility. In the future, certain adjustments to lead time may be specifiable such as when a vendor consistently performs better than contracted. (ISI: Should this be per LTA? Can this example be for just one Line Item?)
Phase: Elaboration
Action: Set computed sda or kill proposal
Attributes Modified: **RDD absent, computed sda**, **meets RDD**, **delivery days ARO**, **delivery days ARO less than 30**
Attributes Accessed: priority, request delivery in days, IPG1_SDA, IPG2_SDA, IPG3_SDA, SDA, VENDOR_DELIV_DAYS
ICP/CBU Default: an ICP/CBU default will provide the typical computation
ICP/CBU Enforcement: none
Sample rule: Lead time for LTA SP020094D0007 -- CTS CORPORATION when priority = IPG1 is computed by "24 hours plus UPS BLUE shipping lead time"
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007, priority=1)
show rule applies specified lead time computation
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007, priority=13)
show rule applies normal lead time computation
Return to Template Index Page

Compute LTA Line Item UI

Description: Compute the Unit of Issue necessary to use this proposal. This computation may require variation based on conditions that may be examined by rules giving non-programmers some flexibility.
Phase: Elaboration
Action: Kill proposal or set vendor Qty, vendor UI, and excess order qty in vendor UI

Attributes Set: **vendor Qty, vendor UI**, **excess order qty in vendor UI**, excess in NSN UI Qty (ISI: this is increase to QTY so need to bump moreDollars)
Attributes Accessed: request quantity, PURCH_UNIT, request unit of issue, CONVERSION_FACTOR, NSN standard Unit of Issue
ICP/CBU Default: an ICP/CBU default will provide the typical computation
ICP/CBU Enforcement: none
Sample rule: LTA Line Item UI is computed by "Use matching UI or convert quantity for NSN-standard UI"
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007,UI=DZ)
show rule applies specified unit of issue computation
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007,UI=mote)
show rule kills proposals for unacceptable units of issue
Return to Template Index Page

Compute LTA Line Item Quantity

Description: Compute the quantity necessary to use this proposal. This computation may require variation based on conditions that may be examined by rules giving non-programmers some flexibility.
Phase: Elaboration
Action: Kill proposal or set order Qty, order UI, and excess order qty in vendor UI
Attributes Set: **order Qty, order UI**, **excess order qty in order Qty** (ISI: this is increase to QTY so need to bump moreDollars)
Attributes Accessed: vendor Qty, vendor UI, CONVERSION_FACTOR, MULTIPLE_ORDER_QTY
ICP/CBU Default: an ICP/CBU default will provide the typical computation
ICP/CBU Enforcement: none
Sample rule: LTA Line Item quantity is computed by "Use vendor Multiple Order Quantity"
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007,qty=20)
show rule applies specified quantity computation, possibly adding excess to meet Multiple Order Quantity
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007,qty=10000)
show rule kills proposals for Price Tiers that do not cover the qty
Return to Template Index Page

Compute LTA Line Item Price

Description: Compute the price to the customer and the cost to DLA. This computation may require variation based on conditions that may be examined by rules giving non-programmers some flexibility. (ISI: Should this be per proposal (all items)?)

Phase: Elaboration
Action: Kill proposal or set order Unit Price, order Ext Price, customer Unit Price, customer Ext Price
Attributes Set: **order Unit Price, order Ext Price, customer Unit Price, customer Ext Price** (ISI: should be allowed to increase QTY and bump moreDollars)
Attributes Accessed: order Qty, order UI, DLA Cost Recovery Rate, LTA Price Adjustment Factor, PTIER_LOCATION_ID, PTIER_START_DT, PTIER_END_DT, BREAK_PT_LOW_QTY, BREAK_PT_HI_QTY, LTA_UNIT_PRICE
ICP/CBU Default: an ICP/CBU default will provide the typical computation
ICP/CBU Enforcement: none
Sample rule: Order price is computed by "Apply the LTA PAF and DLA CRR"
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007,qty=20)
show rule applies specified price computation
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007,qty=10000)
show rule kills proposals for Price Tiers that do not cover the qty (or ship-to location, or effective date)
Return to Template Index Page

LTA Line Item DO Rejected For Requisition

Description: This rule handles the case where a vendor has rejected a Delivery Order for this requisition and the system should not generate another to that vendor.
Phase: TBD
Action: Kill
Attributes Modified:
Attributes Accessed: DO[Requisition.ID].Status; DO[Requisition.ID].LTA_ID
ICP/CBU Default: This rule is to be an ICP/CBU default
ICP/CBU Enforcement: none
Sample rule: LTA Line Item DO Rejected For Requisition
Test case 1: any requisition
show rule does not apply except to kill proposals that duplicate one that has been rejected (DO.Status in database set according to response from Vendor cluster)
Return to Template Index Page

I.1.3.7 Priority and Ranking

Add Depot Backorder Lead Time

Description: **Priority and ranking** requires the delivery lead time which may be implicitly impacted for medium and low priority orders for Depot Stock. A quantity of depot inventory is reserved to handle possible future high priority orders. Thus, there may be enough stock to fill a low priority order, but that order will be backordered to retain the reserved inventory.

Phase: Elaboration
Action: Boost VendorSDA
Attributes Modified: Proposal.ComputedLeadTime
Attributes Accessed: Requisition.Priority; Proposal.QuanityInStock; Proposal.ReserveQuantity
ICP/CBU Default: This rule is likely to be an ICP default only
ICP/CBU Enforcement: none
Sample rule: Add Depot Backorder Lead Time when Priority in IPG2 or IPG3 [if there is not enough to fill order and retain reserve]
Test case 1: requisition(customer=Navy, item supplied by Depot, priority=7)
show rule applies specified stock computation and adds backorder lead time to depot proposal if inadequate stock in addition to reserve level.
Test case 2: requisition(customer=Navy, item supplied by supplied by Depot, priority=2)
show rule applies normal stock computation (all depot stock is available for high-priority orders)
Return to Template Index Page

Current Inventory Fails No-Backorder Request AAC

Description: The No-Backorder AAC option indicates a desire to fill the requisition from current stock. This certainly applies to Depot deliveries and PartNet sources -- with up-to-date inventory data, Sourcing avoids sending backorders to these.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: Requisition.AAC
ICP/CBU Default: This is an ICP default rule only
ICP/CBU Enforcement: none
Sample rule: Current Inventory Fails No-Backorder Request AAC
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007, AAC does not imply "no backorder")
show rule does not kill proposal
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007, AAC implies "no backorder")
show rule kills proposal for this LTA only if current inventory is inadequate.
Return to Template Index Page

Inform Vendor No-Backorder Request AAC

Description: [Future.] The No-Backorder AAC option indicates a desire to fill the requisition from current stock. It is possible that inventory available at the time of Sourcing will be depleted before the DO is processed by the vendor (or Sourcing was lacking valid inventory data). Further analysis is required to understand how this will work with EDI DOs for DVD. If the electronic DO cannot specify no backorder ("fill or kill"), then one solution might be to have the Vendor Cluster examine the response from the Vendor for a DO to verify that the expected delivery time is consistent with the No-Backorder AAC code when present. If the delivery time is

too long, the Vendor Cluster might cancel the order and trigger resourcing. Note that the API with the Vendor Cluster may need enhancement to support this.
Phase: Elaboration
Action: Set NoBackorderAAC
Attributes Modified: Proposal.NoBackorderAAC
Attributes Accessed: Requisition.AAC
ICP/CBU Default: This rule will be ICP default only
ICP/CBU Enforcement: none
Sample rule: Inform Vendor No-Backorder Request AAC
Test case 1: requisition(customer=Navy, item supplied by LTA SP020094D0007, AAC does not imply "no backorder")
show rule does not annotate proposal (and message to Vendor Cluster)
Test case 2: requisition(customer=Navy, item supplied by LTA SP020094D0007, AAC implies "no backorder")
show rule annotates the proposal for this LTA and informs the Vendor Cluster to monitor for a backordered response from vendor.
Return to Template Index Page

Request AAC Allows Up-To-5-Dollars-More

Description:
Phase: Elaboration / Filter
Action: Set MaxDollarsMore / Kill

Allow Dollars More

Description: The Requisition Acquisition Advice (AAC) may indicate a customer's flexibility to pay more and/or the ICP may have a policy of allowing some number of dollars more. This means that the Sourcing module may increase the quantity of a Line Item in order to meet a vendor minimum limitation. The following rules use the proposal attribute MaxDollarsMore as the amount by which the cost may be boosted by increasing the quantity to reach a price tier or meet a minimum or as a result of a conversion of units:

Propose Price Break (Implicit);
LTA Header DO Line Item Minimum Dollars;
LTA Header DO Total Minimum Dollars;
Convert to LTA Line Item Unit Of Purchase;
LTA Line Item DO Line Item Minimum Dollars;
LTA Line Item DO Line Item Minimum Quantity.

The ranking algorithm that runs after all the rules needs to kill proposals that exceed the MaxDollarsMore allowed. The actual dollars added is found by multiplying the QuantityIncrement by the UnitPrice (ISI: this is incomplete; consider the implementation of Propose Price Break (Implicit).)
These rules could be qualified by the Request.Priority to vary the MaxDollarsMore according to Priority or to override the AAC limitation for high priority requesitions. More effort would be needed to allow MaxDollarsMore to be proportional to some other factor, say Unit Price or the total for the order.

Phase: Elaboration / Filter
Action: Set MaxDollarsMore / Kill
Attributes Modified: Proposal.MaxDollarsMore
Attributes Accessed: Requisition.AAC
ICP/CBU Default: This rule is likely to be an ICP default only
ICP/CBU Enforcement: none
Sample rule 1: Request AAC Allows Up-To-5-Dollars-More
Sample rule 2: Allow 50 Dollars More (ICP default rule)
Test case 1: requisition(...,AAC allows Up-To-5-Dollars-More)
show rule 1 provides the MaxDollarsMore annotation to be used by the minimum-testing rules to enable a quantity change when feasible
show rule 2 does not provide a more liberal MaxDollarsMore
Test case 2: requisition(...,AAC does not include allow Up-To-5-Dollars-More)
show rule 1 does not modify this the MaxDollarsMore annotation
show rule 2 provides the ICP default of 50 DollarsMore
Return to Template Index Page

Proposal Does Not Meet Require-for-Free Request AAC

Description: [Further analysis is in progress.] This is a placeholder for an additional rule template dealing with the processing of Acquisition Advice (AACs).
Phase: Filter
Action: Kill
Return to Template Index Page

Lead Time Meets Priority

Description: The requisition priority implies how important it is to meet the requisition Required (Requested?) Delivery Date (RDD). This rule prevents orders to proposed LTAs that cannot fulfill the order in time. Note that in the current design, if all proposed sources are ruled out, then the requisition is kicked out for alternative processing.
Phase: Filter
Action: Kill
Attributes Modified:
Attributes Accessed: computed sda (needs expected shipping duration added), request delivery in days, priority
ICP/CBU Default: these rules are likely to be ICP defaults only
ICP/CBU Enforcement: none
Sample rule 1: Expected delivery lead time may not exceed 3 days for orders with priority more urgent than IPG2 and Requested/required Delivery Date (RDD) in less than 3 days
Sample rule 2: Expected delivery lead time may not exceed 60 days for orders with priority as urgent as IPG2 and Requested/required Delivery Date (RDD) in less than 61 days
Test case 1: requisition(..., priority=7, RDD=today+45days)
show rule 1 does not apply
show rule 2 kills proposals with lead times greater than or equal to 60 days

Test case 2: requisition(..., priority=2, RDD=today+2days)
show rule 1 kills proposals with lead times greater than 3 days
show rule 2 does not apply
Return to Template Index Page

Price/Priority/LeadTime Rating

Description: The final selection among non-excluded proposals follows a ranking algorithm. There is potential for interaction with the rule "Lead Time Meets Priority" -- because that rule "Kills" proposals, it takes precedence over this rule that only rates/ranks proposals. After all applicable rules have been applied, there will be a coded ranking algorithm. The ranking algorithm will first find all proposals that are not killed. Then it will successively partition not killed proposals by the sequence of partitioning factors (see example in next paragraph). By having rules tag proposals with partitioning factors, each ICP may excercise flexibility in the ranking algorithm. The example below implements this algorithm is for Columbus ICP (based on notes from meeting with Frankie Stuart 4/22/98 at ISI, pp 20 [URL?]).

If no Required Delivery Date (RDD) in the requisition
  IPG1 and IPG2, use fastest then cheapest
  IPG3, use cheapest
If the RDD can be met
  use cheapest
If the RDD has passed or cannot be met
  IPG1 and IPG2, use fastest then cheapest
  IPG3, use cheapest if delivery within 30 days, else use fastest

ISI is planning an enhancement whereby an ICP can control the ranking algorithm. A sequence of partitioning factors is declared. Usually "Killed" will be the first partitioning factor with killed proposals being placed below all not killed proposals. Later partitioning will apply only within these partitions so as to uphold the precedence relationship of partitioning factors. When paritioning by fastest or cheapest, the value of the related proposal attribute will be used to partially order the proposals (rather than a binary partitioning). The table below represents actions for a set of rules to rank proposals according to the Columbus ICP algorithm above when used with the partitioning factors sequenced by "Not Killed", "RDD Met", "RDD Near Miss", "Fastest", "Cheapest". Note that if the "Fastest" attribute is not present on any proposal in the partition in consideration, then sorting by fastest is effectively skipped. Also, if any proposal in the partition has the "Fastest" attribute, then at that step all those proposals in the partition without this attribute will be sorted lowest.

Requisition RDD \ Requisition Priority	**IPG 1 and IPG 2**	**IPG3**
RDD absent	Fastest, Cheapest	Cheapest
RDD met	RDD Met; Cheapest	RDD Met, Cheapest
RDD passed/not met; lead	Fastest, Cheapest	RDD Near Miss, Cheapest

time <= 30 days		
RDD passed/not met; lead time > 30 days	Fastest, Cheapest	Fastest, Cheapest

Phase: Rank
Action: Set Ranking Criteria across proposals (error if two proposals try to set different criteria)
Attributes Modified:
Attributes Accessed: VARIES BY PROGRAMMED CRITERIA; Columbus described here uses RDD absent, meets RDD, delivery days ARO, delivery days ARO less than 30, customer Ext Price
ICP/CBU Default: these rules are likely to be ICP defaults only
ICP/CBU Enforcement: none
Sample rule 1: If RDD is absent and Priority in IPG1 or IPG2
then label proposal with "Fastest" and "Cheapest"
Sample rule 2: If RDD is absent and Priority in IPG3
then label proposal with "Cheapest"
Sample rule 3: If RDD is present and met by this proposal
then label proposal with "RDD Met" and "Cheapest"
Sample rule 4: If RDD is present and not met and lead time is <= 30 days and Priority in IPG3
then label proposal with "RDD Near Miss" and "Cheapest"
Sample rule 5: If RDD is present and not met and lead time is <= 30 days and Priority in IPG1 or IPG2
then label proposal with "RDD Near Miss" and "Cheapest"
Sample rule 6: If RDD is present and not met and lead time is > 30 days
then label proposal with "Fastest" and "Cheapest"
Test case 1: requisition(RDD absent, Priority in IPG1)
show that the fastest LTA is highest ranked
Test case 2: requisition(RDD absent, Priority in IPG3)
show that the cheapest LTA is highest ranked
Test case 3: requisition(RDD in future, Priority in IPG1)
show that the cheapest proposal meeting the RDD is selected or else the fastest is highest ranked
Test case 4: requisition(RDD in future, Priority in IPG3)
show that the cheapest proposal meeting the RDD is selected or else the cheapest near miss or fastest
Test case 5: requisition(RDD passed, Priority in IPG1)
show that the fastest LTA is highest ranked
Test case 6: requisition(RDD passed, Priority in IPG3)
show that the cheapest near miss proposal is selected or else the fastest is highest ranked
Return to Template Index Page

I.1.4 Sourcing Data from PartNet

To find the root or latest version of these documents, visit ISI DELTA Sourcing Data.
Please send comments and inquiries to DealMaker@isi.edu

I.1.4.1 PartNet Return Fields Used by DELTA Sourcing

Response Header (one per query)

Attribute Name	xxx	Comments
Commercial-VDI-Queried		Sample values: 1
Commercial-VDI-Responded		Sample values: 1
Depot-VDI-Responded-Flag		Sample values: N | Y
DRMS-VDI-Responded-Flag		Sample values: N
LTA-VDI-Responded-Flag		Sample values: N
Manufacturer-CAGE (Queried)		Sample values:
Manufacturer-Part-Number (Queried)		Sample values:
NIIN		Sample values: | 002504441 | 004822833
Part-Query-Request-ID		Sample values: 11 | 14 | 3
Supplier-Part-Number (Queried)		Sample values: | 36T NE567N

Response Detail (zero or more per query)

Attribute Name	xxx	Comments

Availability		Sample values: 0 | 1224 | 1224 | 2626
Contract-Number		Sample values:
Days-ARO		Sample values:
Item-Description		Sample values: MICROCIRCUIT,LINEAR | RF DEVICES | SUTURE,ABSORBABLE,SURGICAL
Manufacturer-CAGE		Sample values: | 18324
Manufacturer-Name		Sample values: | PHILIPS SEMI CONDUCTORS
Manufacturer-Part-Number		Sample values: | NE567N
NSN		Sample values: | 5962002504441 | 6515004822833
Price		Sample values: 0.38 | 0.9 | 49.4
Supplier-Name		Sample values: DPSC | DSCC | Newark
Supplier-Part-Number		Sample values: | 36T NE567N
Unit-of-Issue		Sample values: | EA | PG

Response Overhead Fields

Attribute Name	Comments
Begin-Object	Sample values: PartItemDetail | PartRequestHeader | PartRequestResult
End-Object	Sample values: PartItemDetail | PartRequestHeader | PartRequestResult

I.1.5 Sourcing Data Stored in DELTA Database

To find the root or latest version of these documents, visit ISI DELTA Sourcing Data. Please send comments and inquiries to DealMaker@isi.edu

Here are some quick links to the old-format tables documented here. Where appropos, the XML field serving that role has been added. Also, there are many XML fields for which no equivalent was apparent in the old-format tables.

- LTA
- LTA_PER_OF_PERF
- LTA_LINE_ITEM
- LTA_SUB_LINE_ITEM
- LTA_PRICE_TIER
- LTA_ITEM
- ITEM
- DLSC_CATALOG_ITEM
- PRODUCT
- ORG; ORG_ID_QUAL; ORG_RELATE_ROLE; ORG_RELATIONSHIP
- VENDOR
- ADDRESS

I.1.5.1 LTA Contracts

LTA Contracts -- XML Fields with no apparent historical equivalent

These three tables document fields in the XML that are accessible to the Sourcing Module that were apparently absent in the preceding representation. They have been added by ISI as a consequence of a functional requirement or in response to a request from DLA or its representative.

XML Fields for LTA Contract Data without variable-controlled values

XML Field Name	Comments
accountable-station-num	
buyer-dodaac	
buyer-name	
buyer-telephone-number	
contract-id/serial-number	
contracting-officer-first-name	
contracting-officer-last-name	
contracting-officer-telephone-number	
edi-sequence	Supports EDI
first-available-call-number	
last-available-call-number	
lta-effective-date	
partial-shipments-are-allowed	
payment-discount-terms	

remit-to	
report-contract-type-code	
solicitation-id	Part of LTA ID?
state-country-code-country	
trading-partner-id	Supports EDI
vendor-mail-to	
vendor-name	

XML Fields for LTA Contract Data with alternative values based on the control variable "Method-of-Support"

XML Field Name	Comments
account-class-reference-number	ACRN
customer-request-unit-of-issue	Expected UI in requisition, presumably the from units for the conversion factor
item-output-routing-code	
mark-for-ship-to	
price-adjustment-factor	Used to adjust all or blocks of price information to allow electronic import of unmanipulated data with contractual discount (markup)
shipment-method-of-payment	•
smcc	
special-reimbursement-provisions	

XML Fields for NSN Standard Item Information

XML Field Name	Comments
standard-item-info/narcotic-code	
standard-item-info/output-routing-code	

LTA Contracts -- Header

Table name: LTA

Field Name	Field Type	XML Field Name
LTA_ID	VARCHAR2(13) NOT NULL	
LTA_REQMT_PROJ_ID	VARCHAR2(10) NOT NULL	project-id
ORG_ID	VARCHAR2(30) NOT NULL	
ORG_ID_QUAL	CHAR(2) NOT NULL	
ASSIGNMENT_ID	VARCHAR2(6) NOT NULL	
PERSON_ID	• CHAR(9) NOT NULL	

VENDOR_ID	VARCHAR2(5) NOT NULL	vendor-cage
VENDOR_ID_TYP_IND	CHAR(1) NOT NULL	
CNTR_ACT	CHAR(1)	contract-id/contracting-activity
CNTR_VEH_TYP	CHAR(1)	contract-id/contracting-vehicle-type
FY	VARCHAR2(2)	contract-id/fiscal-year
GSA_FSS_CNTR_ID	VARCHAR2(10)	
TRANS_LABEL_IND	CHAR(1)	
NEW_NEGN_AUTHY	CHAR(1)	new-negotiation-authority-code
PRICE_COMP	CHAR(1)	price-competition-code
QUOTE	CHAR(1)	
RPT_CNTR_TYP	CHAR(1)	
SCD_DESIGNATOR	CHAR(1)	samms-criticality-designator-code
STATE	VARCHAR2(2)	state-country-code-state
DODAAC	VARCHAR2(6)	
SERIAL_NUM	VARCHAR2(4)	
PURCH_PROCDR	CHAR(1)	purchase-procedure-code
LTA_STAT	CHAR(1)	
SOCIO_ECON_IND	CHAR(1)	socio-economic-indicator
SRC_TYP	CHAR(1)	source-type-code
LTA_AWD_DT	DATE	lta-award-date
LTA_SYS_ENTRY_DT	DATE	
LTA_PERF_START_DT	DATE	
LTA_EXPIRE_DT	DATE	lta-expiration-date
LTA_LAST_DT_USED	DATE	
LTA_MIN_DOL	NUMBER(11,2)	contract-lifetime-total-minimum-value-guarantee
LTA_MAX_DOL	NUMBER(11,2)	contract-lifetime-total-maximum-value-limit
LTA_DO_MIN_DOL	NUMBER(11,2)	delivery-order-total-minimum-value-limit
LTA_DO_MAX_DOL	NUMBER(11,2)	
LTA_DOL_TO_DT	NUMBER(11,2)	

CNTR_PLACMNT	CHAR(1)	
ML_OK	CHAR(1)	multiline-delivery-orders-are-allowed
VAL_ENG_CLAUSE	CHAR(1)	value-engineering-clause-is-included

LTA Contracts -- Period of Performance

For the ISI-developed XML representation, the Period-of-Performance notion was modified. Instead of combining the storage of contract terms with data about the use of the contract, the contract XML contains only the contract terms and the usage data is stored separately. This allows improved versioning of the contract data and improved performance of the usage data management. Further, instead of discrete periods of performance that do not accurately model production capacity, the XML design contains a rolling window of production capacity of variable length. This could allow constraints of the form: up to 300 units per month but not over 200 in any five day period.

Table name: LTA_PER_OF_PERF

Field Name	Field Type
LTA_ID	VARCHAR2(13) NOT NULL
PPERF_NUM	VARCHAR2(2) NOT NULL
PPERF_START_DT	DATE
PPERF_END_DT	DATE
PPERF_STAT	CHAR(1)
PPERF_MIN_DOL	NUMBER(11,2)
PPERF_MAX_DOL	NUMBER(11,2)
PPERF_DO_MIN_DOL	NUMBER(11,2)
PPERF_DO_MAX_DOL	NUMBER(11,2)
PPERF_DOL_TO_DT	NUMBER(11,2)

LTA Contracts -- Line Item

In the XML representation, fields for LTA Line Item and LTA Subline Item are flexibly stored so that fields with a single value are stored once and fields with multiple values are stored such that the correct value is found based on the dynamic values of the controlling "selector" variables. This is a key to avoiding redundant manual data entry. XML fields that are controlled by Method-of-Support are flagged with '^'. The XML Field Name is actually an abstraction that maps to the qualified/controlled field in the XML DTD. This mapping is contained in selectors.xml and is read in at runtime currently. This is an additional form of flexibility in that changes in the XML representation can be tracked

Table name: LTA_LINE_ITEM

Field Name	Field Type	XML Field Name
LTA_ID	VARCHAR2(13) NOT NULL	
LTA_LINE_NUM	VARCHAR2(4) NOT NULL	
ORG_ID	VARCHAR2(30) NOT NULL	
ORG_ID_QUAL	CHAR(2) NOT NULL	
VENDOR_ID	VARCHAR2(5) NOT NULL	^vendor-item-id/cage
VENDOR_ID_TYP_IND	CHAR(1) NOT NULL	
ITEM_ID	VARCHAR2(31) NOT NULL	
VENDOR_PRODUCT	VARCHAR2(24) NOT NULL	^vendor-item-id/pn
VENDOR_PRODUCT_CODE_TYP_IND	CHAR(1) NOT NULL	
PROCMT_METH	CHAR(1)	procurement-method
LINE_MIN_QTY	NUMBER(9)	
LINE_MIN_DOL	NUMBER(11,2)	
LINE_MAX_QTY	NUMBER(9)	
LINE_MAX_DOL	NUMBER(11,2)	
LINE_DO_MIN_QTY	NUMBER(9)	^delivery-order-line-item-minimum-quantity-limit
LINE_DO_MIN_DOL	NUMBER(11,2)	delivery-order-line-item-minimum-value-limit
LINE_DO_MAX_QTY	NUMBER(9)	
LINE_DO_MAX_DOL	NUMBER(11,2)	
LINE_QTY_TO_DT	NUMBER(9)	
LINE_DOL_TO_DT	NUMBER(11,2)	
WARRANTY	CHAR(1)	warranty
TRADE_DISCNT_PERCENT	NUMBER(5,2)	

LTA Contracts -- Sub-line Item

See the note under LTA Line Item above about fields that migrate between Line Item and Subline Item in the XML representation.

Table name: LTA_SUB_LINE_ITEM

Field Name	Field Type	XML Field Name
LTA_ID	VARCHAR2(13) NOT NULL	
LTA_LINE_NUM	VARCHAR2(4) NOT NULL	^lta-item-clin
LTA_SUB_LINE_NUM	VARCHAR2(2) NOT NULL	^lta-item-subclin
METH_OF_SPT	CHAR(1) NOT NULL	method-of-support
ADMIN_BY	VARCHAR2(3)	^administered-by
CONVERSION_FACTOR	NUMBER(14,7)	^unit-of-issue-conversion-factor
EDI_IND	CHAR(1)	^payment-office-edi-code
FAST_PAY	CHAR(1)	^fast-pay
FOB_PT	CHAR(1)	^free-on-board/fob-point
PRINT_IND	CHAR(1)	
OTHER_COST	CHAR(1)	
PAY_OFFICE	VARCHAR2(2)	^payment-office-samms-code
POA	CHAR(1)	^free-on-board/place-of-acceptance
POI	CHAR(1)	^free-on-board/place-of-inspection
PURCH_UNIT	VARCHAR2(2)	^delivery-order-unit-of-issue
QTY_VARIANCE	VARCHAR2(3)	^quantity-variance/overship-percentage ^quantity-variance/undership-percentage
REVIEWER	CHAR(1)	^price-reasonableness/reviewer-code
SHIPMENT_MODE	CHAR(1)	^shipment-mode
TYP_ANALYSIS	CHAR(1)	^price-reasonableness/type-analysis-code
IPG1_SDA	NUMBER(3)	[Same as VENDOR_DELIV_DAYS]
IPG2_SDA	NUMBER(3)	[Same as VENDOR_DELIV_DAYS]
IPG3_SDA	NUMBER(3)	[Same as

		VENDOR_DELIV_DAYS]
MULTIPLE_ORDER_QTY	NUMBER(7)	^delivery-order-multiple-quantity
SDA	NUMBER(3)	[Same as VENDOR_DELIV_DAYS]
VENDOR_DELIV_DAYS	NUMBER(3)	delivery-days [controlled by Issue-Priority-Group from the requisition]
SRP	CHAR(1)	
PRICE_REASON	VARCHAR2(2)	
ACTIVATION_DT	DATE	
SUB_LINE_MIN_QTY	NUMBER(9)	
SUB_LINE_MIN_DOL	NUMBER(11,2)	
SUB_LINE_MAX_QTY	NUMBER(9)	
SUB_LINE_MAX_DOL	NUMBER(11,2)	
SUB_LINE_DO_MIN_QTY	NUMBER(9)	
SUB_LINE_DO_MIN_DOL	NUMBER(11,2)	
SUB_LINE_DO_MAX_QTY	NUMBER(9)	
SUB_LINE_DO_MAX_DOL	NUMBER(11,2)	
SUB_LINE_QTY_TO_DT	NUMBER(9)	
SUB_LINE_DOL_TO_DT	NUMBER(11,2)	

LTA Contracts -- Price Tier

Table name: LTA_PRICE_TIER

Field Name	Field Type	XML Field Name
LTA_ID	VARCHAR2(13) NOT NULL	
LTA_LINE_NUM	VARCHAR2(4) NOT NULL	
LTA_SUB_LINE_NUM	VARCHAR2(2) NOT NULL	
LTA_PRICE_TIER_NUM	VARCHAR2(4) NOT NULL	
LOCATION_ID	VARCHAR2(30) NOT NULL	
LOCATION_QUAL	CHAR(2) NOT NULL	
PTIER_START_DT	DATE	^price-tier-list
PTIER_END_DT	DATE	^price-tier-list
BREAK_PT_LOW_QTY	NUMBER(9)	^price-tier-list
BREAK_PT_HI_QTY	NUMBER(9)	^price-tier-list

UNIT_PRICE	NUMBER(19,6)	^price-tier-list
PRICE_SYS_ENTRY_DT	DATE	

LTA Item

This DELTA database table is probably obsolete. This is because this table is not used by the Cluster that could update these fields.

Table name: LTA_ITEM

Field Name	Field Type
LTA_ID	VARCHAR2(13) NOT NULL
ITEM_SPEC_ID	VARCHAR2(10) NOT NULL
ITEM_LIFE_MIN_QTY	NUMBER(9)
ITEM_LIFE_MIN_DOL	NUMBER(11,2)
ITEM_LIFE_MAX_QTY	NUMBER(9)
ITEM_LIFE_MAX_DOL	NUMBER(11,2)
ITEM_LIFE_QTY_TO_DT	NUMBER(9)
ITEM_LIFE_DOL_TO_DT	NUMBER(11,2)

I.1.5.2 Database data for FGI Items

Data about items has common fields and then specialized fields depending on its mode of identification, NSN or vendor/manufacturer number.

Common Per-item Data

Table name: ITEM

Field Name	Field Type	XML Field Name
ITEM_ID	NOT NULL VARCHAR2(31)	
ITEM_SPEC_ID	NOT NULL VARCHAR2(10)	
ORG_ID	NOT NULL VARCHAR2(30)	
ORG_ID_QUAL	• NOT NULL CHAR(2)	
ASSIGNMENT_ID	NOT NULL VARCHAR2(6)	
PERSON_ID	NOT NULL CHAR(9)	
HAZMAT	CHAR(1)	standard-item-info/hazmat-label-requirement
ITEM_NM	VARCHAR2(40)	
PGC	VARCHAR2(5)	standard-item-info/procurement-group-code

SAFETY_SHEET	CHAR(1)	standard-item-info/material-safety-data-sheet-required
VAL_ENG_PKG	CHAR(1)	standard-item-info/value-engineering-and-packaging
VENDOR_SHLF_LIFE_IND	CHAR(1)	standard-item-info/shelf-life-marking-required

I.1.5.3 Per-NSN Data

Table name: DLSC_CATALOG_ITEM

Field Name	Field Type
ITEM_ID	NOT NULL VARCHAR2(31)
NSN	NOT NULL VARCHAR2(13)
FSC	VARCHAR2(4)
NIIN	VARCHAR2(9)
NSN_POPS_SPT	CHAR(1)
NSN_STD_UNIT_PRICE	NUMBER(9,2)
NSN_STD_UI	VARCHAR2(2)
NSN_STD_MIN_ORD_QTY	NUMBER(7)
NSN_STD_MULT_ORD_QTY	NUMBER(7)

I.1.5.4 Per-vendor-item Data

Table name: PRODUCT

Field Name	Field Type
ORG_ID	NOT NULL VARCHAR2(30)
ORG_ID_QUAL	NOT NULL CHAR(2)
VENDOR_ID	NOT NULL VARCHAR2(5)
VENDOR_ID_TYP_IND	NOT NULL CHAR(1)
ITEM_ID	NOT NULL

	VARCHAR2(31)
VENDOR_PRODUCT	NOT NULL VARCHAR2(24)
VENDOR_PRODUCT_CODE_TYP_IND	NOT NULL CHAR(1)

I.1.5.5 Customer and other Entities

Customer data is stored in the generic ORG set of tables.

Table name: ORG

Field Name	Field Type
ORG_ID	NOT NULL VARCHAR2(30)
ORG_ID_QUAL	NOT NULL CHAR(2)
ORG_NM	NOT NULL VARCHAR2(80)
ORG_NM_UC	NOT NULL VARCHAR2(80)

Table name: ORG_ID_QUAL

Field Name	Field Type
ORG_ID_QUAL	NOT NULL CHAR(2)
ORG_ID_QUAL_NM	NOT NULL VARCHAR2(50)

Table name: ORG_RELATE_ROLE

Field Name	Field Type
ORG_RELATE_ROLE_ID	NOT NULL VARCHAR2(6)
ORG_RELATE_ROLE_NM	VARCHAR2(50)

Table name: ORG_RELATIONSHIP

Field Name	Field Type
ORG1_ID	NOT NULL VARCHAR2(30)
ORG1_ID_QUAL	NOT NULL CHAR(2)
ORG2_ID	NOT NULL VARCHAR2(30)
ORG2_ID_QUAL	• NOT NULL CHAR(2)
ORG_RELATE_ROLE_ID	NOT NULL VARCHAR2(6)

I.1.5.6 Vendor

In the current implementation, Vendor is just another Entity stored in ORG.

Table name: VENDOR

Field Name	Field Type
ORG_ID	NOT NULL VARCHAR2(30)

ORG_ID_QUAL	NOT NULL CHAR(2)
VENDOR_ID	NOT NULL VARCHAR2(5)
VENDOR_ID_TYP_IND	NOT NULL CHAR(1)
VENDOR_NM	VARCHAR2(20)
PC_IND	CHAR(1)

I.1.5.7 Addresses and Geographic Regions

In the current implmentation, Geographic Regions are stored as XML.

Address

Table name: ADDRESS

Field Name	Field Type
ADDRESS_ID	NOT NULL VARCHAR2(6)
ORG_ID	NOT NULL VARCHAR2(30)
ORG_ID_QUAL	NOT NULL CHAR(2)
LOCATION_ID	NOT NULL VARCHAR2(30)
LOCATION_QUAL	NOT NULL CHAR(2)
STREET_ADDRESS_1	VARCHAR2(50)
STREET_ADDRESS_2	VARCHAR2(50)

DODAAC

Geographic Region

I.1.6 Source Module Requistion Input

To find the root or latest version of these documents, visit ISI DELTA Sourcing Data.
Please send comments and inquiries to DealMaker@isi.edu

ALPINE defined these classes in the FGI cluster code.

Java class name: interface Requisition

Property Name	Java Type	Comments
getRequisitioner()	DodaacLocation	Returns DodaacLocation representation of Requisitioner DODAAC. * Comes from A0 Document Number
getShipToDODAAC()	DodaacLocation	Returns DodaacLocation representation of DODAAC ship to address
isDomesticShipment	boolean	Is this shipment CONUS or OCONUS?

()		
getPriority()	int	Returns integer from 1 to 15 representing priority level of A0.
isExpeditedTransportation()	boolean	Expedite Xport?
isNonMissionCapableSupply()	boolean	Returns true if unit is non-mission-capable.
getFundingCode()	String	Returns String representation of funding code.
getProjectCode()	String	Returns three character String for project code.
getRawMILSTRIP()	String	Returns String of raw A0 record.
getBillToDODAAC()	DodaacLocation	Same idea as ShipTo.
isRecurringDemand()	boolean	A0 demand indicator.
getRequisitionAdviceCode()	String	Return String representing one of possible advice codes.
getDocumentIdentifierCode()	String	Returns String representation of A0 document ID
getDocumentNumber()	String	Returns String representation of A0 document number.
getStatusToDODAAC()	DodaacLocation	Return DodaacLocation for shipment status
getCBUClusterTimestamp()	Date	Not in A0 record; Used for testing and auditing.
getRic()	String	
getMediaStatusCode()	String	
getSuffixCode()	String	
getSupplAddr()	String	
getSignalCode()	String	
getDistrCodeMDN()	String	

Java class name: interface DodaacLocation extends NamedPosition

Property Name	Java Type	Comments
getDodaacCode()	String	The dodaac code representing this position
getAddress()	String	The address representing this position
getShippingAddress()	MailingAddress	The MailingAddress object representing the shipping address
getBillingAddress()	MailingAddress	The MailingAddress object representing the billing address

getServiceID()	String	The String representing the service ID

Java class name: interface NamedPosittion extends Position

Property Name	Java Type	Comments
getName()	String	The string name representing this position

•

Java class name: interface Position extends Location

Property Name	Java Type	Comments
getLattitude()	Latitude	The Latitude representing this position
getLongitude()	Longitude	The Longitude representing this position

Java class name: interface Location extendsALPObject

Property Name	Java Type	Comments
		nothing now

J Design Rationale

Not due in this installment. To be written in upcoming installment.

Contract and Rule Editor

K Background

Among many requirements that emerged for the FGI system, the ISI system gave particular emphasis to addressing those concerned with flexibility and end-user control. Part of the concern was to allow DLA to explore new business models and alternative relationships with suppliers. Another aspect of the issue was to enable the system to grow in functionality over time. Another aspect was to support local control of policies at different centers. The last aspect of this requirement was to reduce dependence on costly and delaying software modifications, by supporting modification at the end user and system administrator level.

In order to address these requirements, ISI developed a set of capabilities to support end user specification of contracts and programming of business policies. In order to make this set of capabilities attractive and friendly from the users' perspective, a great deal of attention needed to be paid to the look-and-feel of the user interface, as well as the design of the navigation process (i.e., the dialogue or alternative sequences of screens that users would encounter in order to accomplish each of many tasks supported by the system).

This section describes both the capabilities supported and the user interface issues addressed.

L Functional Goals

L.1 Usability Goals

This section describes the usability goals and requirements for the contract and rule editor. These goals and requirements were never stated explicitly during the project. This section is based on knowledge acquired during the project by talking to users and DLA officers, and represents ISI's point of view of usability requirements reasonable for this application.

L.1.1 Users and Frequency of Use

The users of the contract and rule editor are contract managers and buyers.

It is envisioned that the users will use the editor infrequently, perhaps not as often as once a week, and perhaps only once a month. Typically, users will enter a new contract in the system after it is negotiated, and then perform updates to it sporadically, perhaps as infrequently as once or twice a year. Because users do not manage a very large number of contracts, we envision that they will not need to use the editor more than once a week.

The policies for managing a contract are typically established when the contract is created and updated, so we expect the rule editing features of the contract editor to be used mostly when contracts are entered or updated. However, DLA can adjust policy for using contracts at any time, but again, we don't expect policy changes to be frequent.

Based on the frequency of use it is reasonable to assume that users of the contract and rule editor will not become experts at using the editor. Ease of learning is of primary importance. A quantifiable metric was not formulated for ease of learning, but we offer the following as a reasonable goal:

- First time users should be able to enter a contract and establish policies on their own, after two hours of instruction, and using a five page introductory user guide.

L.1.2 Minimize Errors

It is imperative that contracts entered into the system be error free. The two main approaches for achieving this goal are prevention and correction. The prevention approach calls for interface techniques that prevent users from making errors. The correction approach typically allows users to enter information rapidly, and the system invokes an error correction dialogue when it detects incorrect input.

We favor the error prevention approach, and state as a goal that the editor should prevent errors whenever possible.

There are several reasons for selecting the error prevention approach. First, users are infrequent and are not expected to become experts at using the editor, so they are likely to make errors if the system does not prevent them. Second, the editor requires users to enter a large number of codes. Users are not expected to remember what all the codes stand for, and most importantly, cannot be expected to accurately recall the correct code for a value that they want to enter. Should they remember incorrectly, they may enter a valid code for a field, but not the one that they intended. Such an error would not be detected by an error correction scheme, but may be detected by an error prevention scheme that shows the users the codes and explanations for them.

L.1.3 Show Explanations for Codes

A refinement of requirement L.1.2 is that for fields requiring users to enter codes, the editor should always show explanations for the codes.

L.1.4 Expedite Entry of Repetitive Fields and Values

Contracts contain two types of fields, global fields such as the contract id and vendor information, and per-item fields such as item-id and inspection point. In contracts with multiple items it is often the case that the per-item fields have the same value for all or most items. For such cases, the editor should provide a capability that allows users to enter the value once as a default for all items, and that allows users to override the value for particular items.

The capability for entering defaults should be flexible so that users can decide on a per-contract basis which fields have defaults. Contracts vary so that in some contracts a field is common for most items where as in other contracts the same field has different values for different items.

The goal of the default capability is to eliminate the need for users to enter the same value in multiple screens. This is especially important for large contracts where the amount of repetitive work would be substantial.

L.1.5 Provide an Expert Mode

Even though section L.1.1 states that users of the editor are not expected to become experts, the editor should provide a mode that allows users to quickly enter all the fields in a contract. The DLA users are familiar with contract entry interfaces that allow all the information for a contract item to be entered in a single screen. These interfaces do not satisfy most of the usability goals stated in this section, but users feel strongly about the ability to enter contracts very quickly.

However, users were very receptive to screens that display explanations of fields, values and codes. The only viable solution seems to be to provide an expert mode that allows users to enter the information faster, and to allow them to switch between the normal and expert mode as they see fit.

L.1.6 Provide Web-based Access

The interface should be accessible from a Web-browser. Compatibility between multiple web browsers is not required, although it is preferred. An implementation for Microsoft Internet Explorer version 4 is acceptable.

L.1.7 Provide Only Minimal Customization and Personalization Features

It is not necessary to provide extensive customization and personalization features that allows users to change the visual appearance of the way the editor works. The only requirement in this regard is the expert mode.

L.1.8 Provide a Professionally Designed Look and Feel

The DLA users are familiar with the Web, and they expect the contract and rule editor to provide a professionally designed look and feel comparable to well designed sites on the Internet.

L.2 Contract Editor Goals

This section describes the tasks that the users need to be able to perform using the contract editor.

L.2.1 Login

The contract editor is the gateway to all DELTA information in the system. Each user should have a login id and password, and the system should authenticate users before giving them access to any information in DELTA. Users should be able to change their password.

Appropriate security measures should be established so that unauthorized users cannot access DELTA.

L.2.2 User Privileges

Once a user logs in, they should have read-only access to all contract and business rules in the system. In addition, users should get privileges to modify information that is appropriate to their task (see section XXX for a specification of the privileges mechanism). The editor should read the privileges associated with a user from the DELTA database, and prevent users from modifying information that they do not have privileges for. The editor should either gray out or not present menu items that would allow users to issue commands outside their privileges.

The privileges mechanism is not required to be secure, in the sense that it is intended to prevent inadvertent modifications to unprivileged information, but is not required to prevent users from intentionally subverting the privilege mechanism by directly writing programs or issuing low level http requests to the server[4]. The notion is that if a user successfully logs in, it is assumed that he or she will not subvert the system. Should a user subvert the privileges mechanism, the database logs will allow system administrators to identify the login id of the person who did it.

The next sections describe all the tasks that users should be able to perform using the editor. It is understood that only users with the appropriate privileges should be able to perform the tasks that create, modify or delete information.

L.2.3 Open Contracts For View Or Edit

Users should be able to open contracts for viewing or editing. It is envisioned that the DELTA database will have hundreds or perhaps thousands of contracts. The editor should allow users to

[4] It is assumed that implementing a secure version of the privileges mechanism is significantly more complex, and the benefits do not warrant the extra cost.

specify the contract id of the contract to open, and also offer a way for users to search for the contract they want to open and to select it from a list.

L.2.3.1 Specifying A Contract To Open

The search mechanism should support searching based on any combination of the following criteria:

- Vendor information. Contracts with vendors, who can be identified using one or more of the following fields: vendor name, cage code, city, state or zip code.
- Item identification. Contracts containing a particular item identified by NSN or part number.
- Contract start date. Contracts that start after the given date.
- Contract end date. Contracts that end before the given date.
- Expiring within X days. Contracts that will expire within the number of days that the user specifies.
- Reaching maximums. Contracts that are getting close to exceeding one of the many performance maximums that are specified in the contract (see section XXX for details about these maximums). The system may use a heuristic to determine which contracts satisfy this criterion.
- With warnings. Contracts saved in draft mode which contain errors or warnings that need to be corrected before the contract can be put into production.
- Deactivated. Contracts that have been deactivated.

The following convenience features are not required but are worth considering:

- Shortcut that allows users to quickly open contracts that they have opened recently.
- Ability for users to define a shortcut list to contain contracts that they want to access quickly.

L.2.3.2 Open Draft or Production Version

The editor should provide a capability to let users open either the latest draft of a contract or the production version of the program.

If a user does not have modification privileges for a contract, the editor should only allow the user to open the contract in read only mode. If a user has modification privileges, the editor should open the contract in edit mode.

If a user opens the production version of a contract, the editor should open it in read only mode. The editor should make it obvious to users that they are viewing the production version of a contract rather than the latest draft, which may be different from the production version.

L.2.4 View and Edit All Contract Fields

Users should be able to view and edit all fields in a contract. This requirement breaks down into finer-grained requirements.

L.2.4.1 Present Field Values in A Logical Way

Many field values correspond to codes used in SAMMS. Whenever possible, values should be presented in their logical uncoded representation, as well as in their coded representation.

For example, consider the price discount factor. In SAMMS this field is represented using a 4-position coded value. The code 3001 specifies that a 01% discount will be given for payment within 30 days. The code BBNN specifies that a ½% discount will be given for payment within 10 days and ¼% discount within 20 days.

The interface for viewing should show the codes as well as the logical explanation of the codes. The interface for editing should allow users to enter either the codes, or specify the logical field values, constraining entry so that only logical values corresponding to valid codes can be entered.

L.2.4.2 Provide Facility to Specify Defaults and To Override Them

Users should have the ability to specify defaults for fields whose values are the same in all or most items in a contract.

When specifying the value for a field in an item, users should be able to see the default value if one is defined. Users should not have to take any action to accept the default.

Users should be able to override a default value for a field, and should also be able to un-override a value, that is, to revert an overridden value back to the default.

Users should be able to specify that no default is provided for a field and that its value should be specified for every item.

L.2.4.3 Allow Users to Split Fields

As mentioned in the contract representation section, fields can be split into sub-fields, allowing the user to specify a different value for each sub-field. For example, a user can split the inspection point for an item according to method of support. Doing so allows the user to specify that the inspection point for method of support DVD should be destination, and the inspection point for method of support Stock should be origin.

Field splits can be thought of as splitting a cell to convert it into a table. The value of a field can be thought of as a cell where the user can enter a value. When the field is split, the value becomes a table. The headings on the table are the criteria for splitting the field, and the cells in the table correspond to the values for each split. In the example above, inspection point was split into a table with one row where the headings are DVD and Stock. Fields can be split according to multiple criteria, yielding a multi-dimensional table. For example, the price for an item is often split according to method of support, quantity and region. This leads to a table which conceptually has three dimensions. This information can be displayed as two tables one for DVD and one for Stock. Each table would have rows corresponding to regions and columns corresponding to quantity breaks.

Splits and defaults can be combined so that users can specify split values as defaults, and then override only some of the cells in the split. For example, a user might specify a split for inspection point according to method of support, and then in a specific item override only the value for method of support DVD.

The requirements for split fields are the following:

5) Users can specify that they want to split a field, and specify the criteria for splitting the field.

6) Users can specify that the default value of a field is a split, and specify the default value for each cell in the split.

7) Users can override individual cells of split fields. If users want to override the criteria for splitting a field, they must override the complete field value (no cells should be inherited from the default).

The general requirement for split fields may prove difficult to design for and to implement. This requirement can be relaxed in two ways. First, only a few fields require that users be able to select the criteria for splitting a field (e.g., price). Second, most fields can be split only according to predetermined criteria (e.g., method of support). For those fields it is not necessary to allow users to choose the criteria. The editor can simply present the table with the predetermined splits and allow the user to specify a value in each cell. The trade-off is that users may need to enter the same value in each cell of the table in the cases when they do not require the field to be split.

L.2.4.4 It Should Be Easy To Add New Fields To A Contract

This is not a requirement for an end-user feature but rather a requirement for system administrators or developers. It is likely that DLA will find the need to support more complex contracts in the future, requiring the contract editor to capture the values for those fields. It should be easy to extend the editor to incorporate screens to prompt for new fields. Ideally, system administrators should be able to add new fields without developer involvement.

L.2.5 Validate Fields

The contract editor should ensure that data entered into the system is correct and consistent. Requirement L.1.2 specifies that the editor should use interface designs that minimize and prevent errors whenever possible. This can be done by using menus instead of allowing the user to type in information, and by making menus context-sensitive so that illegal choices are disabled. However, these techniques cannot eliminate all errors, so it is necessary for the editor to validate the information after the user enters it. When the system detects errors or inconsistencies, it should flag the appropriate fields and values so that the user can identify the errors and correct them. The system should prevent the user from placing into production any contract that does not satisfy the validation tests.

The editor should include two types of validations:

1) Errors flag situations where the data is incorrect and the user must correct it.

2) Warnings flag situations where the data is correct, but unusual. The user can ignore warnings.

We currently do not have a field by field specification of all the validations that the editor should perform. The DTD for the contract representation is a starting point for the validation in that it defines the syntax of each field and the range of possible values that each field can hold. The semantic and cross-field validations have not been specified.

L.2.5.1 Single Field Validation

The editor should validate that the value of each field to ensure that it is syntactically correct (relevant for fields where the user is allowed to type in a value). The editor should also validate that

the values of fields are reasonable (e.g., the start date of a contract is not 500 years from today's date).

L.2.5.2 Cross-Field Validation

The editor should validate that the values of fields are consistent with each other (e.g., start date of a contract is before the end date). The editor should perform the cross-field validations after the single field validations.

L.2.6 Create New Contracts

The contract editor should provide a capability for creating new contracts. To create a new contract, the user must supply the following fields:

- Contract Id: identifies the contract uniquely. It is illegal to create contracts with duplicate ids.
- Contract type: the system supports several kinds of contracts as mentioned in section XXX.

Once the user supplies the required fields, the system should create a draft of the contract and record it in the database. The system should give the user the ability to fill in all the other fields in the contract and define items.

The system should allow the user to save drafts of the contract in the database and allow them to edit the drafts at a later time. The user may place a draft into production provided it has no errors. Contracts must be placed into production before order can be placed on them.

L.2.7 Delete Contracts

The contract editor should provide a capability to delete contracts provided that they have not been placed into production. This allows the users to delete drafts of contracts that they did not intend to create, for example if they notice that they entered an incorrect contract id. Once a contract is placed into production, it cannot be deleted from the system (see section L.2.9).

The editor should provide a capability to discard the current draft of a contract. In that case, the last production version becomes the current draft of the contract.

L.2.8 Approve Contracts

This requirement is optional.

The editor should provide a capability for users to approve contract drafts. A user with approval privileges must approve a contract before it can be placed into production. If a draft of an approved contract is modified, it must be approved again in order to place it into production.

A more general notion of approval may also be considered, which allows users to approve individual items of a contract. This would allow users to place an item into production even though there may be outstanding issues on other items.

L.2.9 Activate/Deactivate Contracts

Activating a contract places it into production and allows orders to be placed against it. A contract can be deactivated to temporarily or permanently remove it from consideration for placing orders.

The editor should provide a capability for a user with activation privileges to activate and deactivate contracts.

See section L.2.15 for requirements regarding activation of periods of performance.

L.2.10 Import And Validate

The contract editor should provide a capability to import a contract into the DELTA database. At this time we have only defined a requirement to import contracts in XML representation. The editor should prompt the user for the URL and read the XML representation from the corresponding location.

The contract editor should be forgiving with respect to errors in the XML representation. It should incorporate as much of the information as it can, even if it does not conform to the DTD for contracts. The editor should flag any errors or missing information so that the user can correct it and complete it. The errors should be flagged in the same way as if the user had incorrectly entered the information.

It is important to note that an imported XML contract may contain errors that the contract editor user interface would prevent. This requirement implies that the validation component of the editor should check for all possible errors regardless of whether the editor interface may prevent some of the errors.

The editor should only reject the file if the contents are not valid XML syntax.

L.2.11 Provide Support for Updating Contract Information from Files

The contract editor should provide a facility for uploading new prices and other fields for items from files supplied by contractors. The contract editor should be able to upload plain text files with tab separated fields:

- The first line in the file is a tab separated list of field names.
- The information for each item is provided in subsequent lines, each one tab separated according to the format defined in the first line.

The names in the first line must correspond to the names of fields as defined in the XML DTD for contracts.

The interface for uploading files into a contract should provide facilities to allow the user to specify the following conditions:

- Items that appear in the file and didn't appear in the contract: the user should be able to specify that these items should be rejected or that they should be added to the contract draft.
- Items that belong to the contract but do not have a corresponding line in the file: the user should be able to specify that those items should be removed from the contract or that those items should be left untouched (i.e., not updated).

L.2.12 Provide Support for Grouping Items

The contract editor should provide a facility to allow users to define groups of items. The most important use of the grouping facility is to support the upload of new item data from contractors.

Contractors will often group items according to certain criteria and provide separate update files for different groups. Allowing users to define the same groups in the editor should simplify the update process. In addition, the grouping facility can be used to define different defaults for different groups of items.

L.2.12.1 Define New Groups

The contract editor should provide a facility to allow users to define groups of items. The editor should provide a predefined group that serves as the root of all groups, and where all new items are placed if no group is specified for them. To define a new group a user must specify the parent group, and a name for the group. The parent group is optional, and if it is not provided, the new group should be defined as a child of the root group.

It is not necessary to provide a facility to define groups based on the properties of items.

L.2.12.2 Remove Groups

The contract editor should provide a facility to let users remove groups. The interface should let users specify what to do with items or sub-groups of the group to be removed. The choices should be to also remove the contents of the group, or to move all the members to the parent group.

L.2.12.3 Rename Groups

The contract editor should provide a facility to allow users to rename groups.

L.2.12.4 Move Groups

The contract editor should provide a facility to allow users to move a group so that it becomes a child of another group.

L.2.12.5 Move Items

The contract editor should provide a facility to allow users to move items from a group to other groups.

L.2.13 Add/Remove Items

The contract editor should provide a facility to add and remove items from a contract.

The facility to add items to a contract should prompt the user for the item id and the group where the item should be placed. If no group is defined, the item should be added to the root group or the current group (interface designers are free to choose whatever option they see fit, but the design should make clear to the user which design option has been chosen). When the user adds a new item to the contract, the editor should provide screens to allow the user to fill in all the fields for the item.

The facility to remove items from the contract should place the item into a deleted-items group, rather than completely removing it from the contract. This capability allows the user to undo the deletion by moving the item out of the deleted-items group.

L.2.14 Support a Notion of Drafts

When users edit a contract, they edit a draft of the contract. Users can save the draft without affecting the production version of the contract.

The editor should support a linear notion of drafts. When a user saves a draft, it supercedes the previous draft. There is no need for the system to retain superceded drafts. The editor should not allow users to create parallel drafts.

The contract editor should allow users to delete a contract draft. In that case, the last production version becomes the current draft of the contract.

An alternative formulation of this requirement would to require the system to retain all previous drafts. This would allow users to revert to previous drafts of the contract. Implementation of this option may require significant additional storage, and requires interface features to allow users to view previous drafts of a contract and to restore a previous draft as the current draft. It does not appear necessary at this time to provide this level of support.

L.2.15 Support Periods Of Performance and Time Phases

The contract editor should allow users to enter information that will take effect at some date in the future. For example, users should be able to enter item information (e.g., price) that will take effect when a new period of performance goes into production.

The contract editor should provide a general capability that allows users to define a contract phase. A contract phase has a start date and expires when the next contract phase starts or the contract end date. When a contract is first created, the editor creates a base phase that spans the lifetime of the contract. When users define periods of performance, the editor should create a new contract phase corresponding to each period of performance.

Contracts contain two kinds of fields, time-independent fields, and phased fields. For example, the contract id is a time-independent field because it cannot change in future phases of the contract. The price of an item is a phased field because it can hold a different value for each phase of a contract. Appendix XXX specifies which contract fields are time-independent, and which ones are phased.

In order to view or edit phased fields, users must first select a contract phase. The editor should show the values defined for that phase. When users modify phased fields the editor should behave as follows:

- Fields for expired phases cannot be modified. A phase is deemed expired if its end date is before the current date.

- Modifications to field values should be recorded in the currently selected phase.

- By default, values should be copied to all future phases. The editor should offer users the option to specify that a modification applies only to the currently selected phase.

All modifications to contract data are subject to the rules stated above. In particular, these rules also cover groups and items. Therefore, this mechanism allows users to introduce new items and or retire items in future periods.

The editor represents periods of performance as phases, but periods of performance are subject to an additional requirement. Users can activate and deactivate periods of performance in the same way that they can activate and deactivate contracts. In fact, if the current date reaches the start date of a period of performance that has not been activated, the system should stop placing orders against the contract. Issue: should the activation/deactivation mechanism be defined only on a period of performance basis rather than on a contract as a whole?

L.2.15.1 The Now Phase

The editor should also provide support for a phase called the now phase. The now phase is a sliding phase whose start date is always the current date. The now phase allows users to edit information corresponding to the current phase of the contract. When the user saves the contract, the editor defines a new phase that contains the new changes, and whose start date is by default the current date. During the dialogue for saving the now phase, the user should have the option of setting the start date in the future so that the changes only go into effect at that future date.

L.2.15.2 Create New Phase

The editor should offer the capability to create new contract phases. To create a new phase users must specify the date when the new phase starts. The editor should splice in the new phase by splitting the phase that contains the date that the user specifies. In addition, the editor should copy all the data from the phase that got split into the new phase.

L.2.15.3 Change Phase Start Date

The editor should offer the capability to modify the start date of a phase. If the new start date causes the phase to shift after an already existing phase, then that phase should be split and the moved phase should be spliced in. The data for the moved phase should remain intact.

L.2.15.4 Delete Phase

The editor should provide a capability to delete phases. When a phase is deleted, all the information defined in it should be deleted too.

L.2.16 Support Flexible Pricing Models

The general mechanism for supporting split fields (see section L.2.4.3) implicitly specifies that the editor should support flexible pricing models. This section makes these requirements concrete for the price field of contracts.

The editor should allow users to split the price field according to the following fields:

- Method of support, so that users can specify different prices for DVD and for Stock buys.
- Geography, so that users can specify regional prices. If users specify that price should be split according to geography, the editor should prompt for the name of a geography, which specifies the regions to be used for regional pricing.
- Quantity breaks, so that users can specify tiered pricing. If users specify that price should be split according to quantity, the editor should prompt the user for the quantities that define each price tier.

Once users specify the criteria for splitting prices, the editor should allow users to enter the different prices.

L.2.17 Provide Integration With Order Management

The contract editor should integrate contract editing and order management.

The contract editor should provide global menus to invoke the order management screens. The details of what order management commands should be offered is not described in this document (contact ALPINE for details).

The contract editor should provide order management commands associated with open contracts. When a contract is open, users should be able to invoke order management commands that show the order activity for the corresponding contract.

The contract editor should provide a capability that allows the order management screens to offer commands to open a contract. If a user is browsing order information in one of the order management screens, he or she should be able to issue a command that opens the contract on which a specific order was placed.

L.2.18 Undo

There is no requirement to provide an undo feature. In lieu of undo, the following capabilities should be provided to give users a way to recover from errors:

- All changes in a session should be auto-saved in a new draft of the contract. Users should have the option to discard all changes since the last time they explicitly invoked the save command, and have the ability to restore the editor to the last saved version.

- The editor is free to choose the granularity for performing auto-save operations. This granularity should be at least at the item level so that when users switch to a new item all changes for the current item are auto-saved. Should the system crash, the editor should restore the last auto-saved version of the data.

- Dialogue boxes should provide reset and cancel buttons. The reset button allows users to revert the contents of a dialogue box to the values prior to any user changes. The cancel button allows users to dismiss a dialogue box without committing any changes.

L.3 Geographies

The contract and rule editor should allow users to define geographies, which are specifications of a group of regions. Geographies should be used in the contract editor as a criterion for splitting fields. In particular, the price field should support geography-based splitting to allow users to define regional pricing (i.e., different prices for different regions). Geographies should also be used in the rule editor to allow users to specify rules that depend on regions.

Geographies should have the following fields:

- Name: specifies the name of a geography.

- Regions: a list of regions.

Regions should have the following fields:

- Name: the name of the region. A region is uniquely identified by the name of the geography it belongs to and its name. In other words, the name of a region need only be unique within a geography. The same region name can be defined differently in different geographies.

- Contents: the states, countries, zip-codes (5 digits) or DODDACs contained in the corresponding region.

The regions in a geography should not overlap. This requirement is difficult to achieve because the editor may not have access to the address information of DODAACs. In addition, if zip-codes are added to a region, care should be taken to make sure that if a region contains the corresponding state, the relevant zip-codes should be subtracted from that region. For example, suppose that region A is defined as containing Arizona and the 90292 zip-code, and that region B contains California and Nevada. The 90292 zip-code is in California, so the editor should make sure that when 90292 is added to region A it is subtracted from region B.

The regions of a geography do not have to cover the whole world.

L.3.1 Open Geography

The editor should provide a capability to allow users to open a geography for viewing or editing. If the logged in user has privileges for modifying a geography, the editor should open the geography in edit mode. Otherwise, the geography should be opened in read-only mode.

L.3.1.1 Open Geography – Global Command

Users should be able to specify the geography to be opened in the following ways:

- By name: users should be able to type-in the name of the geography they want to open. If users type in a partial name, the editor should present a list of all the geography names that contain the name that the user provided, and allow the user to select from the list of matches.

- By selecting from a list: the editor should offer a menu of all the geographies defined in the system and let users select the one they want to open.

There is no need to support capabilities to allow users to filter down the list of geographies they want to open.

There is no need to support a capability to open recently used geographies, or to allow users to define shortcuts to open geographies.

L.3.1.2 Open Geography – Contextual Command

The editor should provide a capability to allow users to open a geography from all screens that use geographies. Presumably those screens will mention the geography by name. This capability would allow users to open the corresponding geography to verify that its definition is appropriate for the context in which it is being used. The geography would be opened in edit mode if the user has modification privileges. Otherwise it would be opened in read-only mode.

L.3.2 Create Geography

The editor should offer a capability to let users define new geographies. The editor should prompt the user for the name of the geography, and create an empty geography that the user can edit to specify the regions.

L.3.3 Add Regions To Geographies

The editor should provide a capability to let users add regions to existing geographies. The editor should require the user to specify a name for the region, and should offer capabilities to define the contents of the region.

As mentioned at the beginning of section L.3, region members can be states, countries, zip-codes and DODAACs. The editor should provide facilities to specify each of these types of members. The editor should provide capabilities to add and remove members of a region.

L.3.4 Remove Regions From Geographies

The editor should offer a capability to remove a region from a geography. When a region is removed, its members become eligible to become members of other regions.

L.3.5 Rename Geography

The editor should offer a capability to let users change the name of a geography. This capability should only change the name of a geography that is displayed in all editor screens. For example, if a geography was used to define regional prices in one or more contracts, changing the name should not affect the definition of the prices.

L.3.6 Delete Geography

The editor should offer a capability to let users delete geographies, provided that they are not used by any contract or rule.

L.4 Rule Editor Goals

To be written.

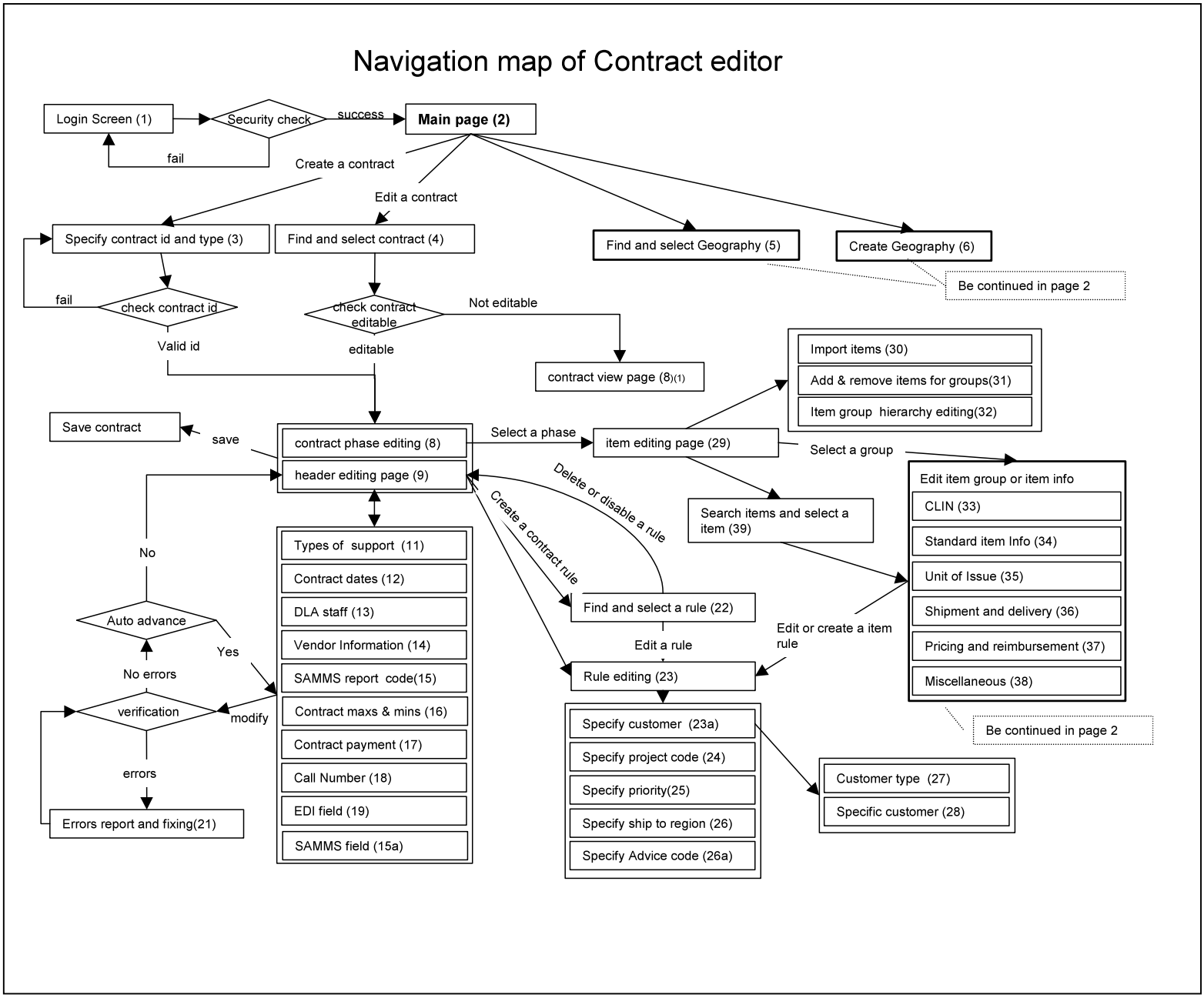
M Design and Architecture

Not due in this installment. To be written in upcoming installment.

M.1 Navigation Maps

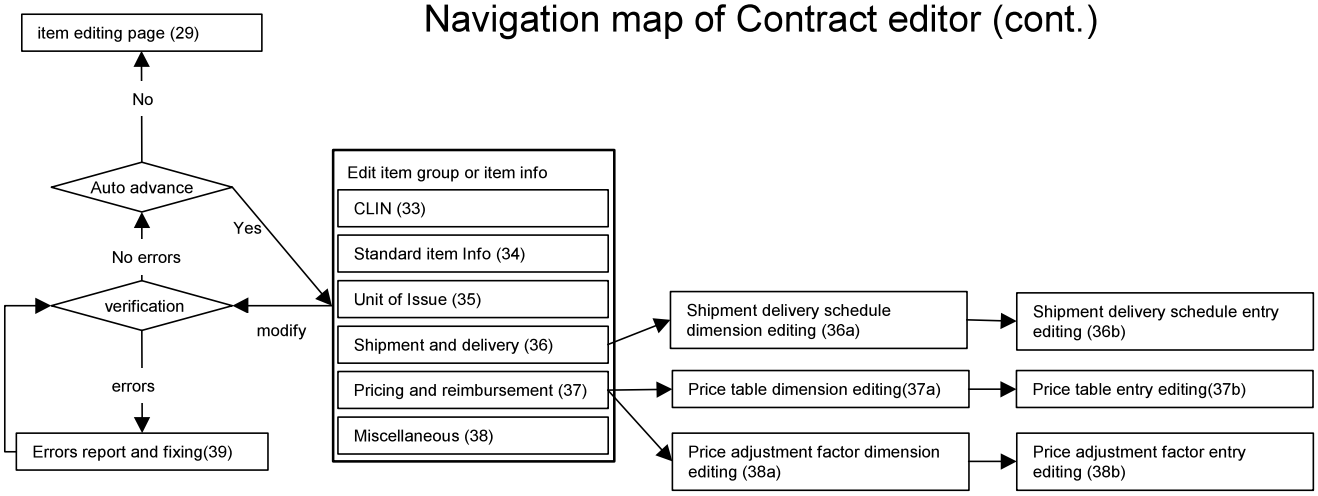
The following material is a preview of the navigation map section of the document. Each box in the navigation map is numbered. The pages following the navigation map are indexed with these numbers and show the corresponding screen. ***BECAUSE OF TRANSMISSION DIFFICULTIES, WE ARE DELETING THESE ADDITIONAL FIGURES FROM THIS DOCUMENT AND SENDING THEM AS A SEPARATE DOCUMENT.***

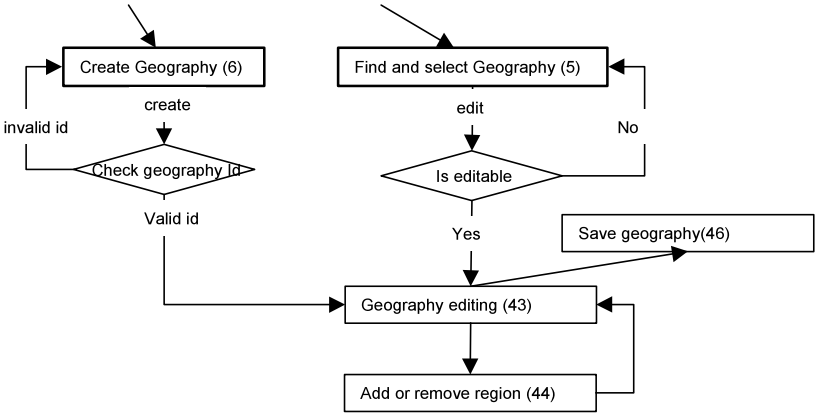
In the next installment we will write the text that explains the conditions for traversing from one screen to the next.



This page intentionally left blank.

Navigation map of Contract editor (cont.)





Notes:
(1) From contract view page (8), navigation will be the same except that there is “modify & verification”.

In the diagram, the one direction flow is reversible by clicking on the “cancel” button.

The closed related pages are represented as sub-blocks inside a block.

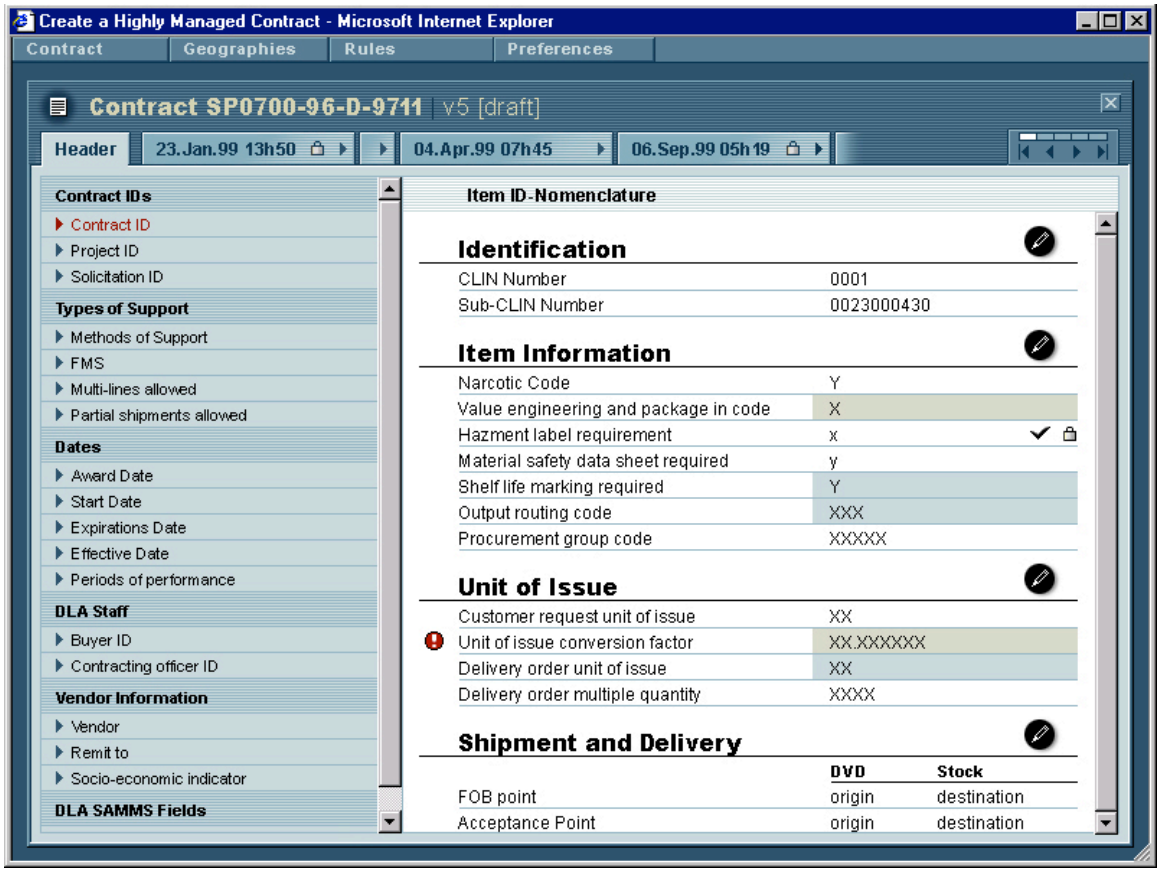
N Design Rationale

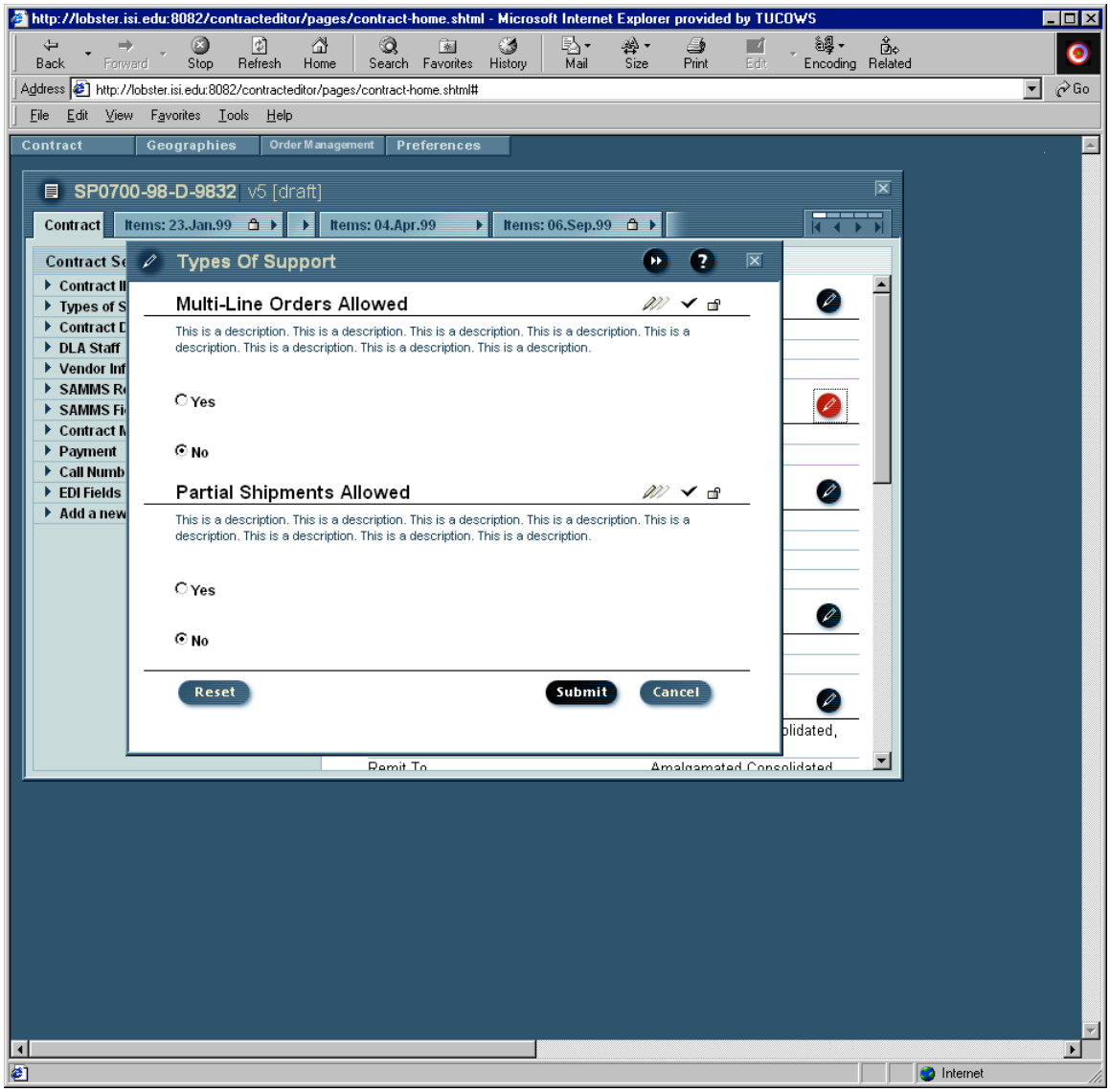
Not due in this installment. To be written in upcoming installment.

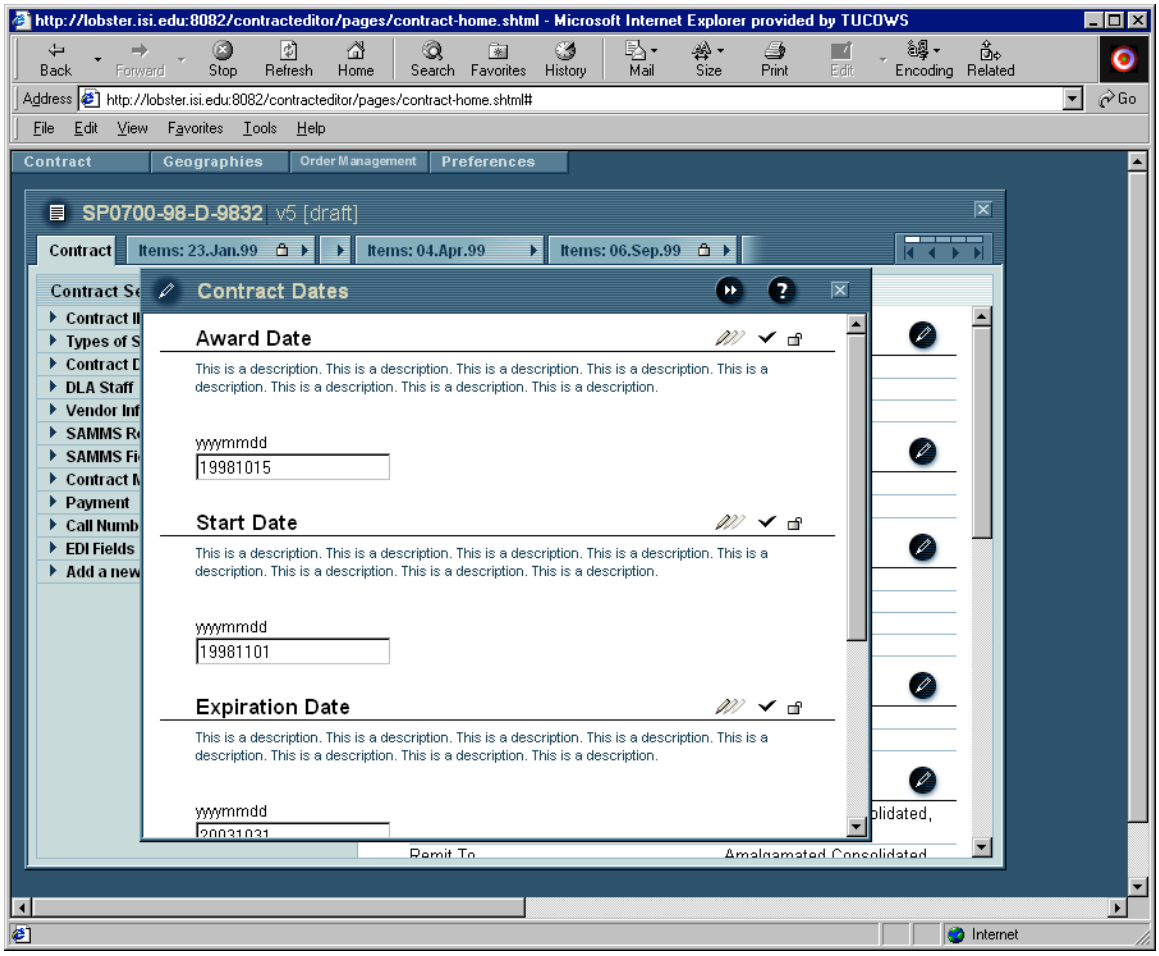
(1)

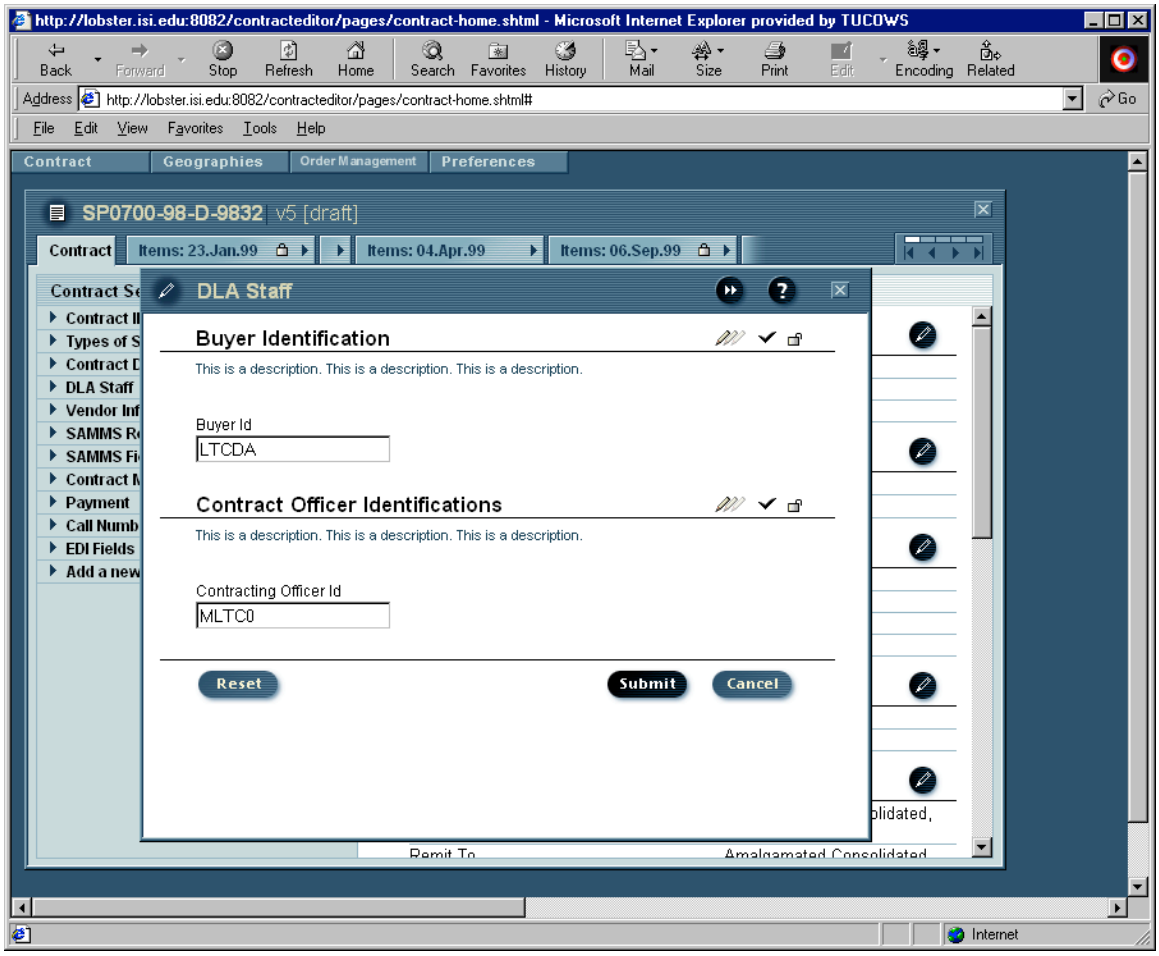

Create a Highly Managed Contract - Microsoft Internet Explorer
Contract
Geographies
Rules
Preferences
delta
(2)

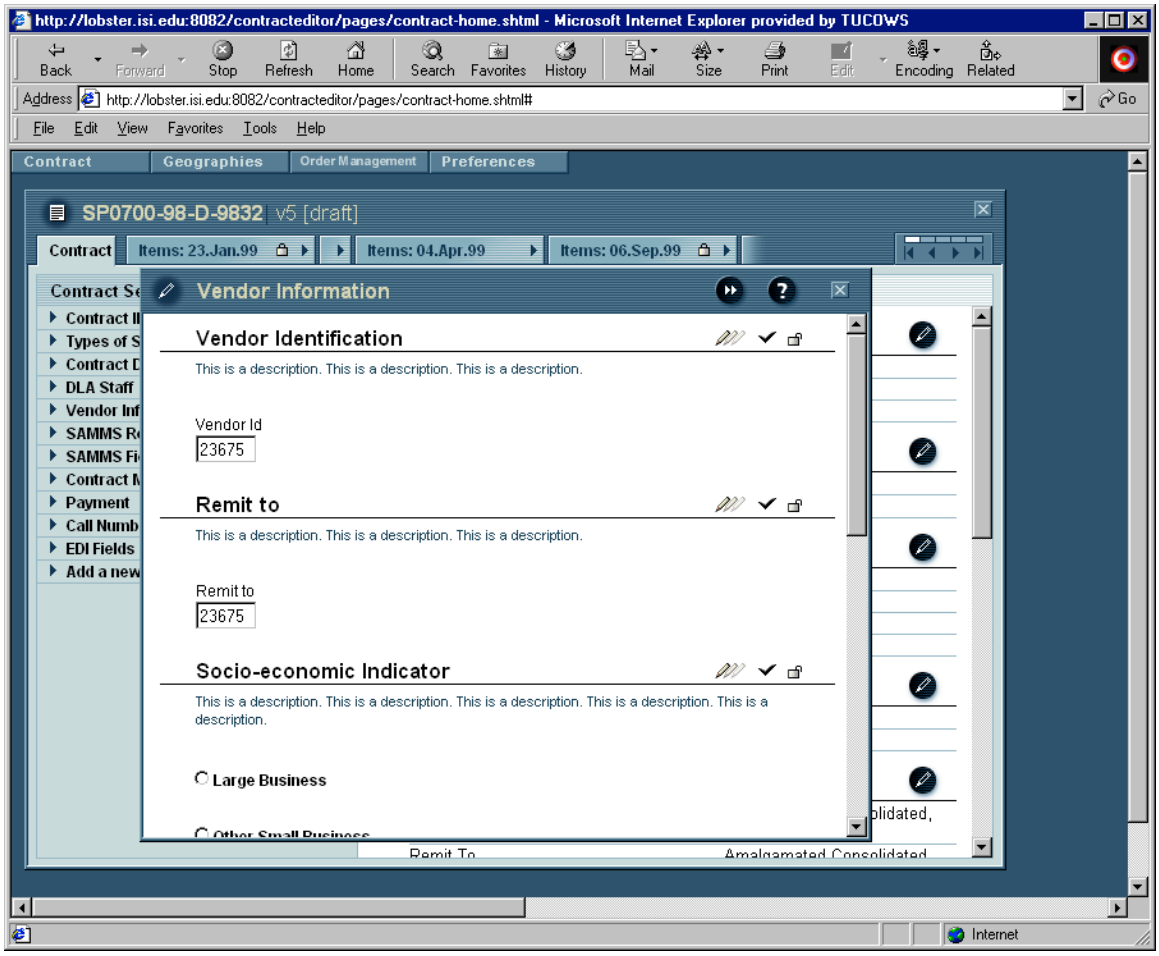
(9)


Create a Highly Managed Contract - Microsoft Internet Explorer
Contract
Geographies
Rules
Preferences
Contract SP0700-96-D-9711 v5 [draft]
Header
23.Jan.99 13h50
04.Apr.99 07h45
06.Sep.99 05h19
Contract IDs
Contract ID
Project ID
Solicitation ID
Types of Support
Methods of Support
FMS
Multi-lines allowed
Partial shipments allowed
Dates
Award Date
Start Date
Expirations Date
Effective Date
Periods of performance
DLA Staff
Buyer ID
Contracting officer ID
Vendor Information
Vendor
Remit to
Socio-economic indicator
DLA SAMMS Fields
Item ID-Nomenclature
Identification
CLIN Number 0001
Sub-CLIN Number 0023000430
Item Information
Narcotic Code Y
Value engineering and package in code X
Hazment label requirement x
Material safety data sheet required y
Shelf life marking required Y
Output routing code XXX
Procurement group code XXXXX
Unit of Issue
Customer request unit of issue XX
Unit of issue conversion factor XX.XXXXXX
Delivery order unit of issue XX
Delivery order multiple quantity XXXX
Shipment and Delivery
DVD
Stock
FOB point origin destination
Acceptance Point origin destination

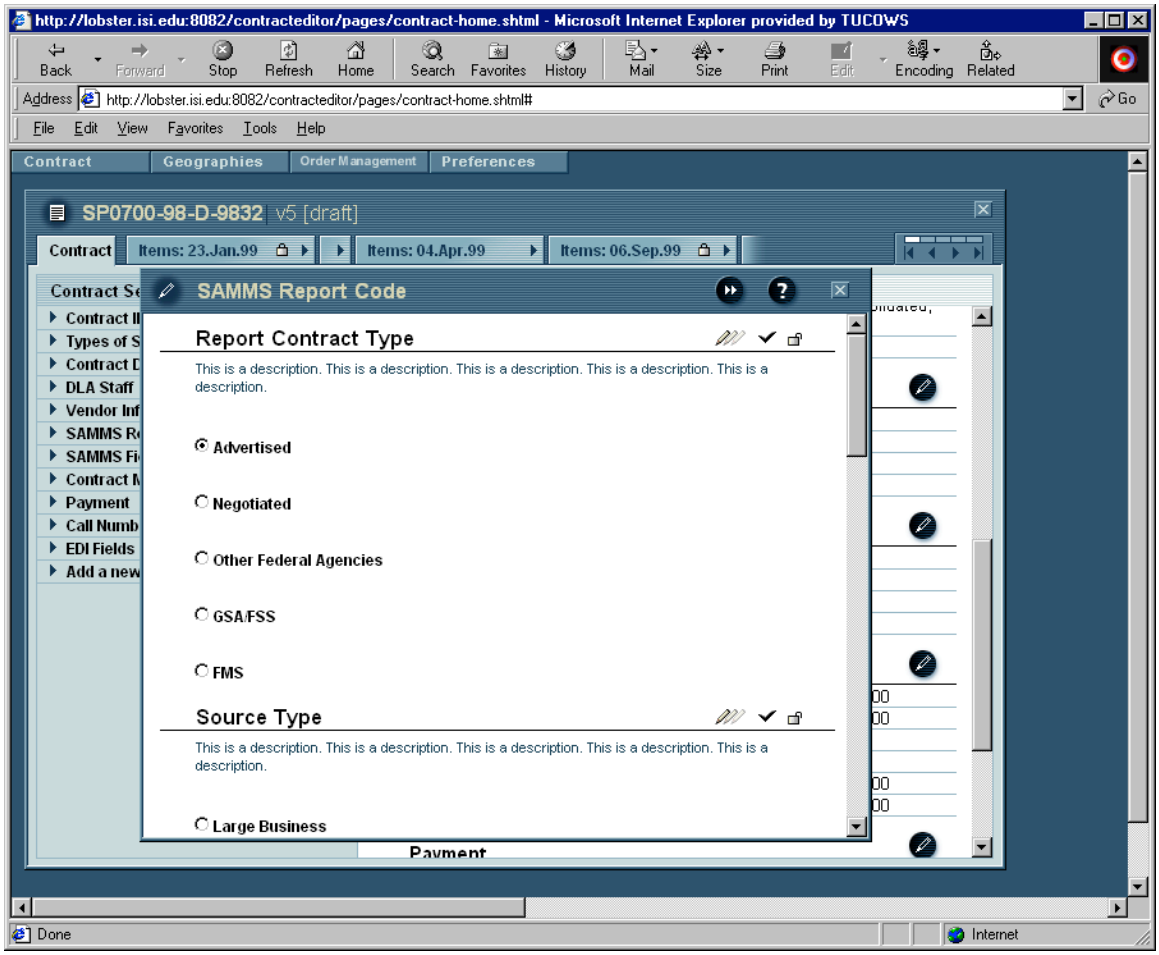

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Types Of Support
Multi-Line Orders Allowed
This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description.
Yes
No
Partial Shipments Allowed
This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description.
Yes
No
Reset
Submit
Cancel
Payment
Call Numb
EDI Fields
Add a new
Remit To
Internet
(11)

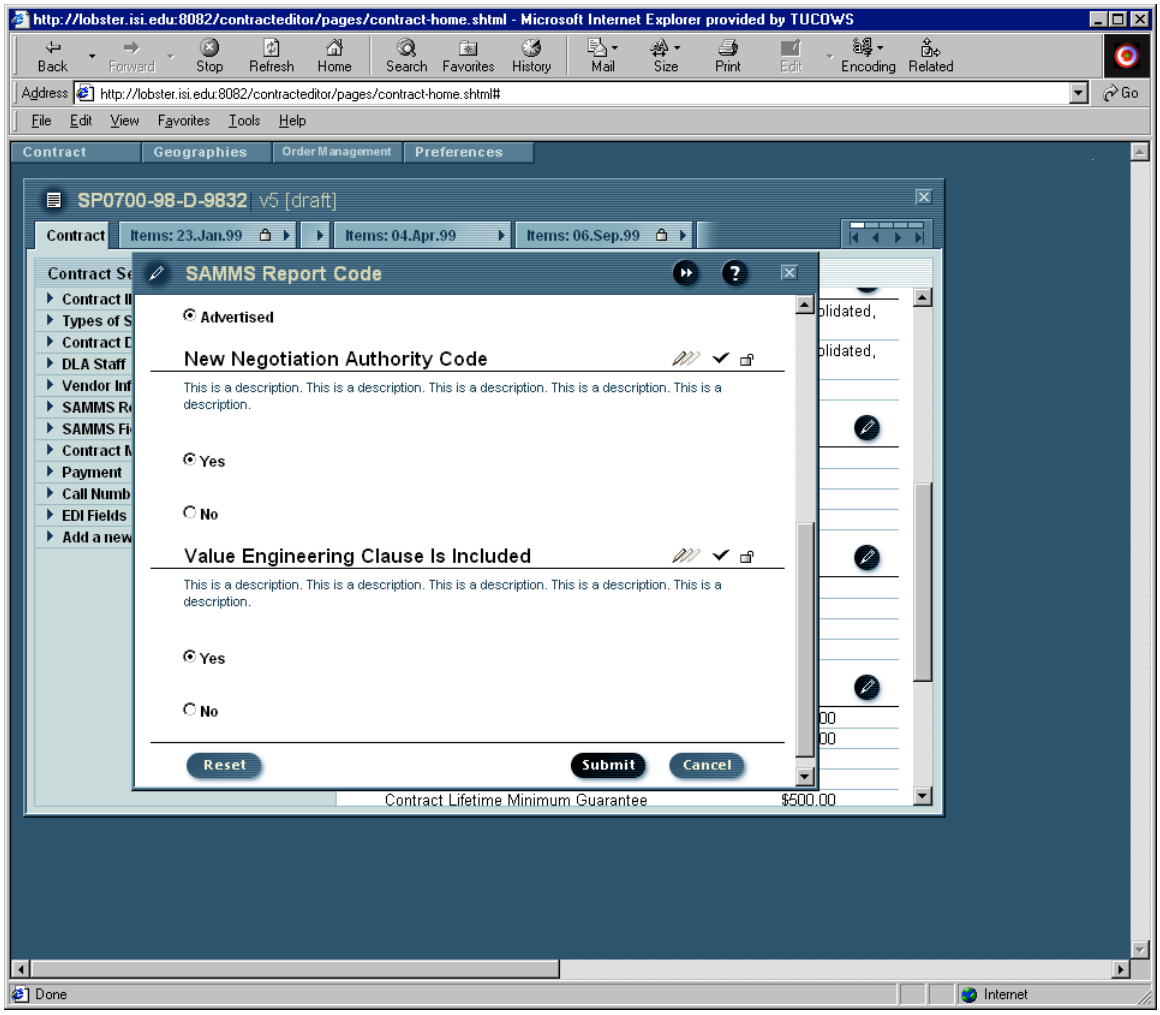

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Contract Dates
Award Date
This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description.
yyyymmdd
19981015
Start Date
This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description.
yyyymmdd
19981101
Expiration Date
This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description. This is a description.
yyyymmdd
20031031
Contract Types
DLA Staff
Vendor Inf
Payment
Call Numb
EDI Fields
Add a new
Remit To
Amalgamated Consolidated
Internet
(12)

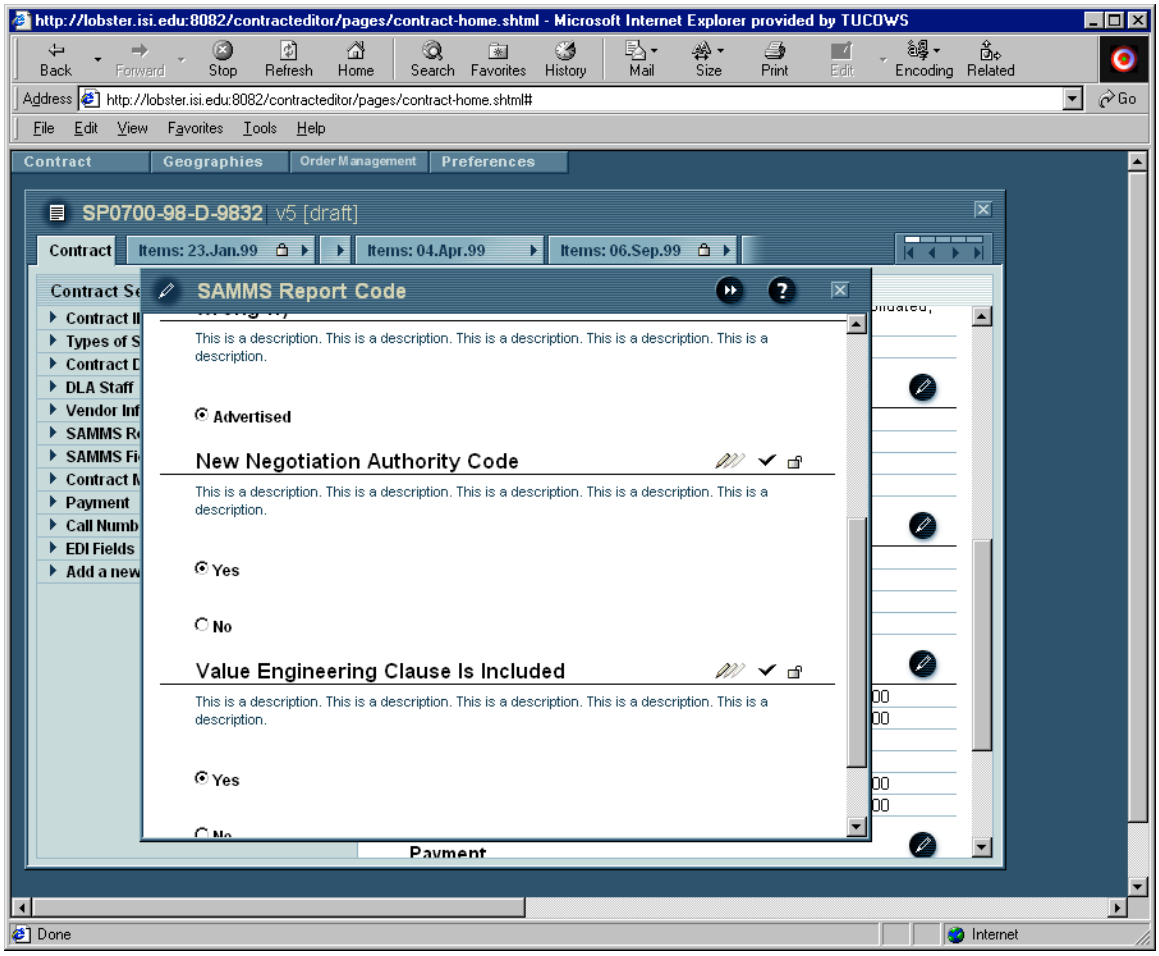

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
DLA Staff
Buyer Identification
This is a description. This is a description. This is a description.
Buyer Id
LTCDA
Contract Officer Identifications
This is a description. This is a description. This is a description.
Contracting Officer Id
MLTC0
Reset
Submit
Cancel
Internet
(13)


http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Vendor Information
Vendor Identification
This is a description. This is a description. This is a description.
Vendor Id
23675
Remit to
This is a description. This is a description. This is a description.
Remit to
23675
Socio-economic Indicator
This is a description. This is a description. This is a description. This is a description. This is a description.
Large Business
Internet
(14)

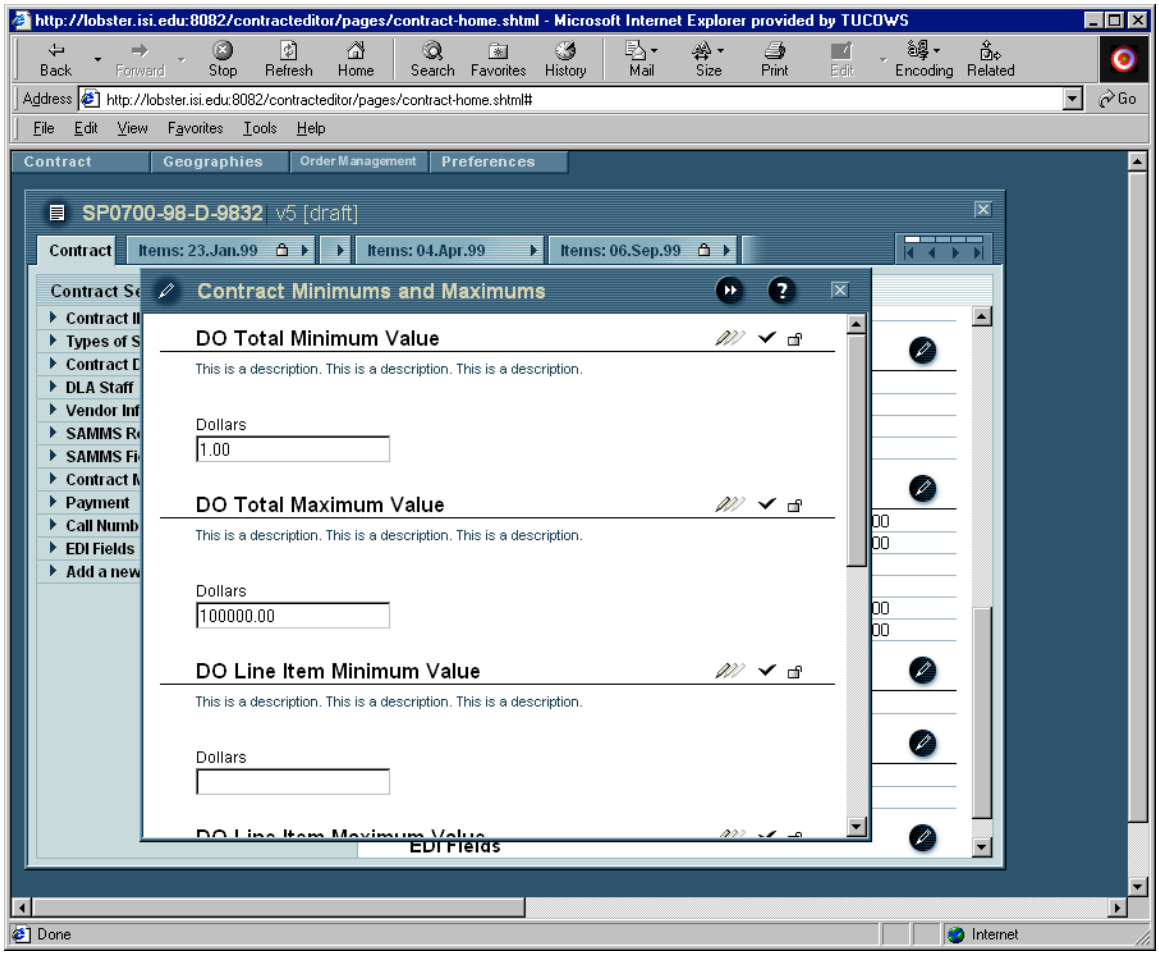
(15)

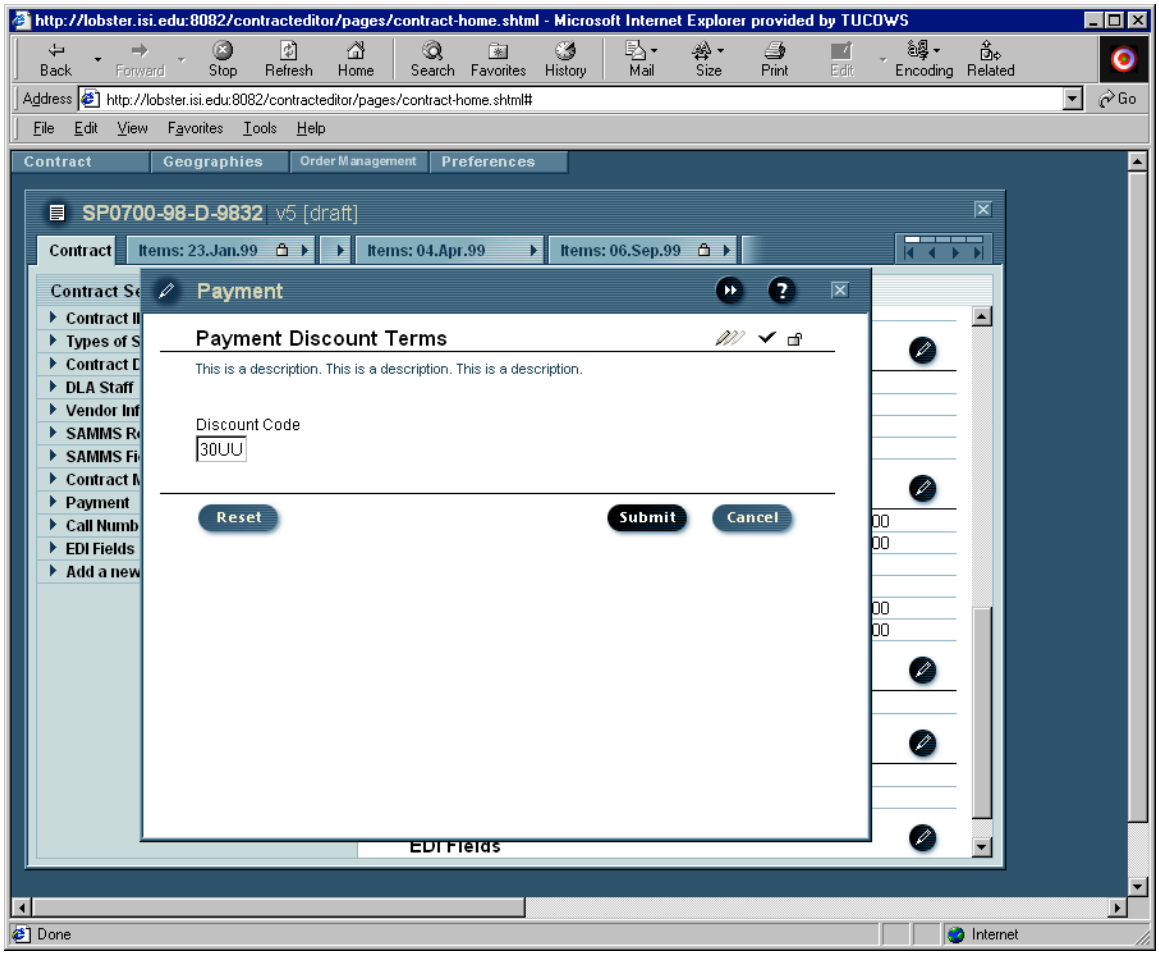

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
SAMMS Report Code
Report Contract Type
This is a description. This is a description. This is a description. This is a description. This is a description.
Advertised
Negotiated
Other Federal Agencies
GSA/FSS
FMS
Source Type
This is a description. This is a description. This is a description. This is a description. This is a description.
Large Business
Payment
Done
Internet

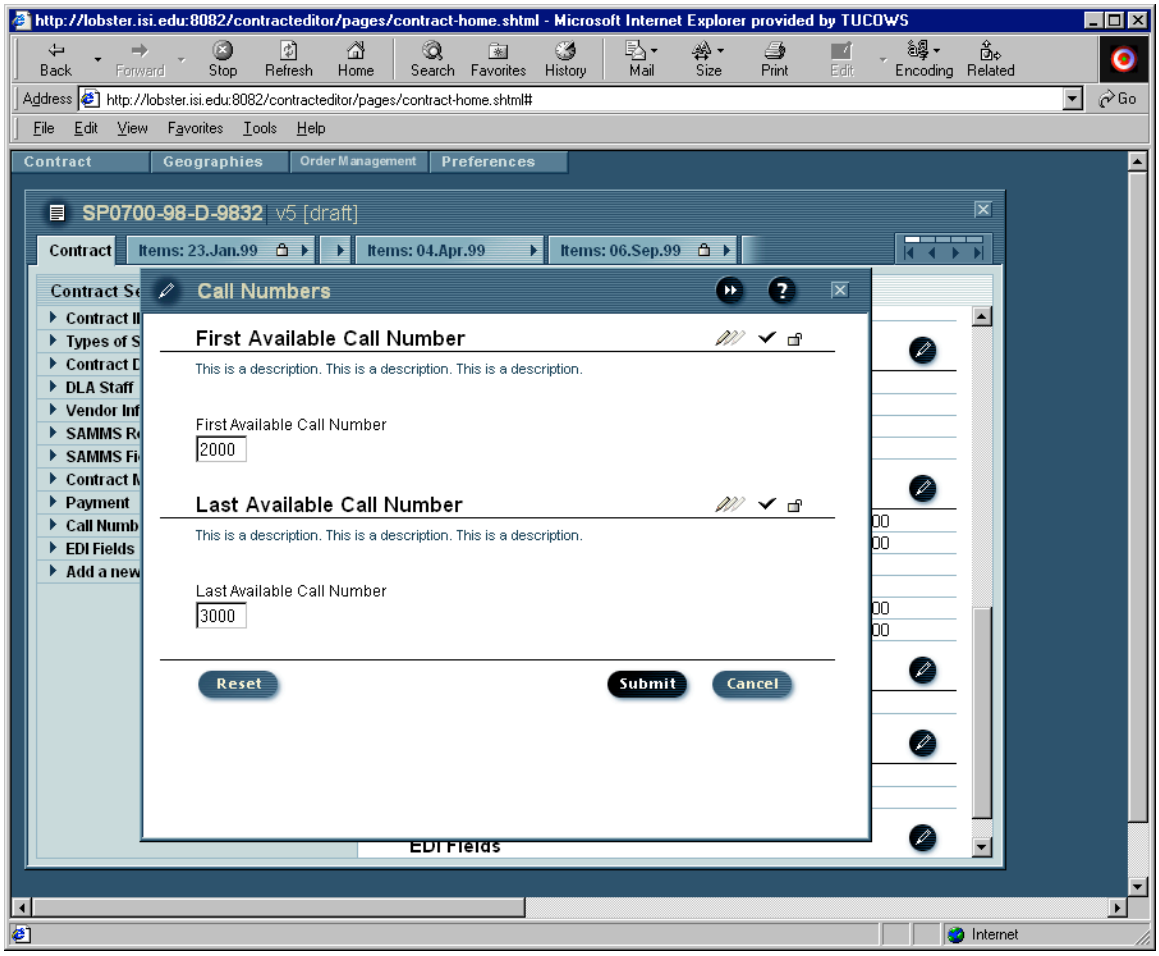

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Contract Se
Contract Il
Types of S
Contract D
DLA Staff
Vendor Inf
SAMMS R
SAMMS Fi
Contract M
Payment
Call Numb
EDI Fields
Add a new
SAMMS Report Code
Advertised
New Negotiation Authority Code
This is a description. This is a description. This is a description. This is a description. This is a description.
Yes
No
Value Engineering Clause Is Included
This is a description. This is a description. This is a description. This is a description. This is a description.
Yes
No
Reset
Submit
Cancel
Contract Lifetime Minimum Guarantee
$500.00
Done
Internet
(15a)

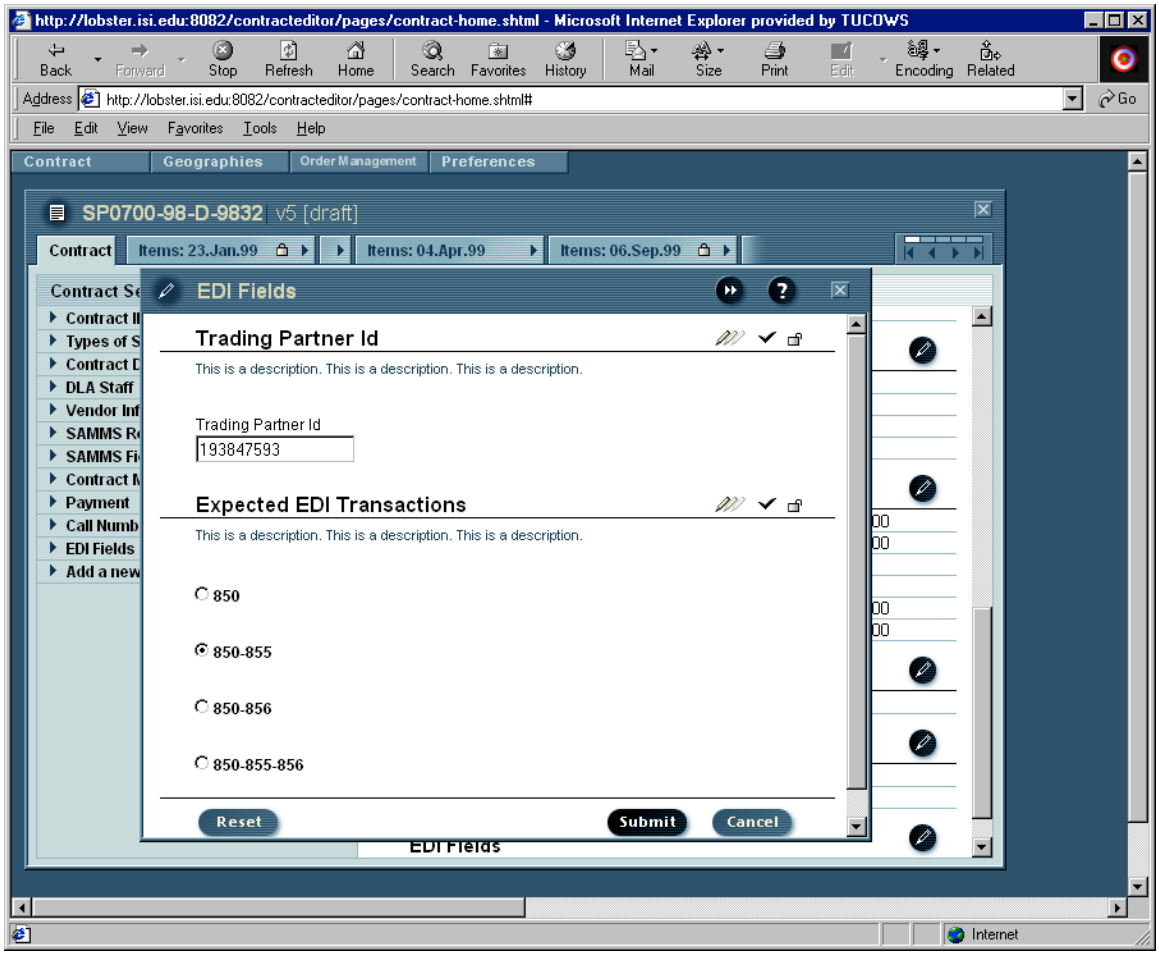

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
SAMMS Report Code
This is a description. This is a description. This is a description. This is a description. This is a description.
Advertised
New Negotiation Authority Code
This is a description. This is a description. This is a description. This is a description. This is a description.
Yes
No
Value Engineering Clause Is Included
This is a description. This is a description. This is a description. This is a description. This is a description.
Yes
No
Payment
Done
Internet
(16)

(16)

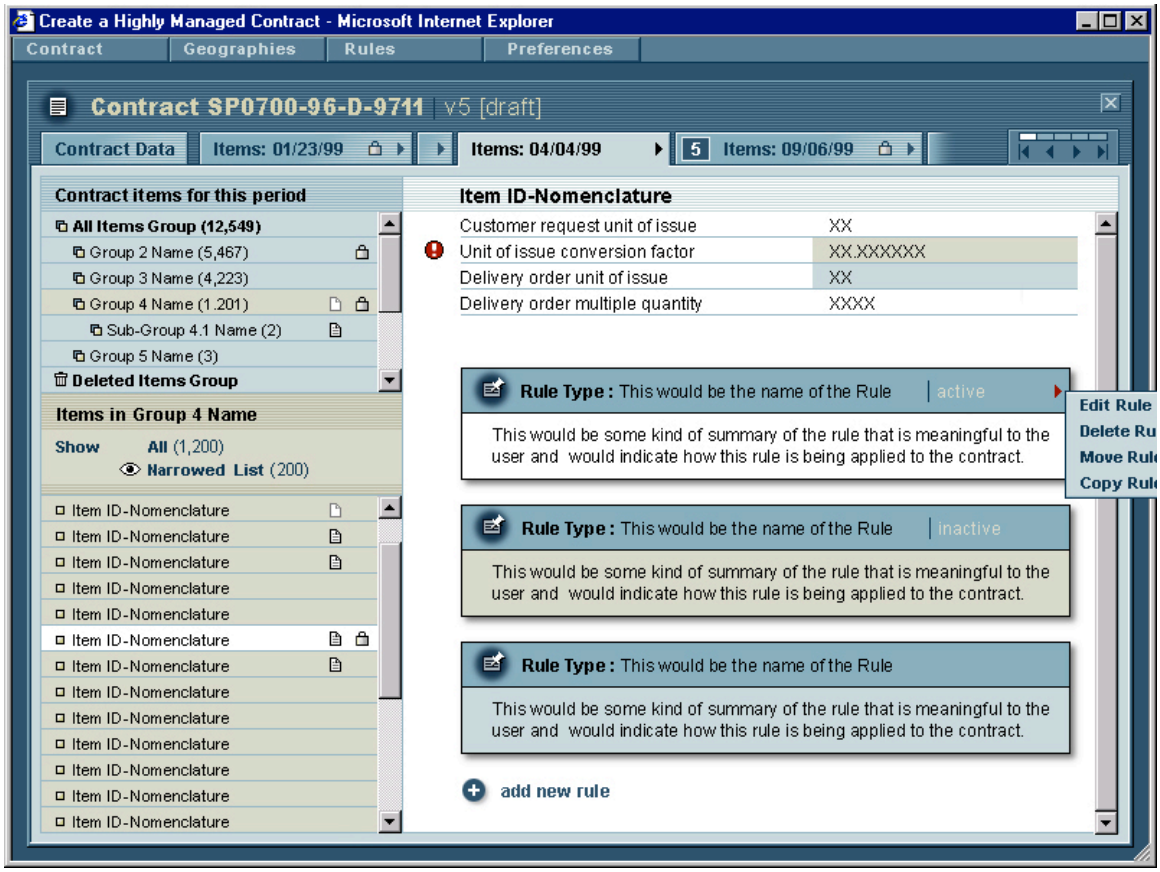

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Contract Minimums and Maximums
DO Total Minimum Value
This is a description. This is a description. This is a description.
Dollars
1.00
DO Total Maximum Value
This is a description. This is a description. This is a description.
Dollars
100000.00
DO Line Item Minimum Value
This is a description. This is a description. This is a description.
Dollars
EDI Fields
Done
Internet


http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Payment
Payment Discount Terms
This is a description. This is a description. This is a description.
Discount Code
30UU
Reset
Submit
Cancel
DLA Staff
Payment
Call Numb
EDI Fields
Add a new
EDI Fields
Done
Internet
(17)

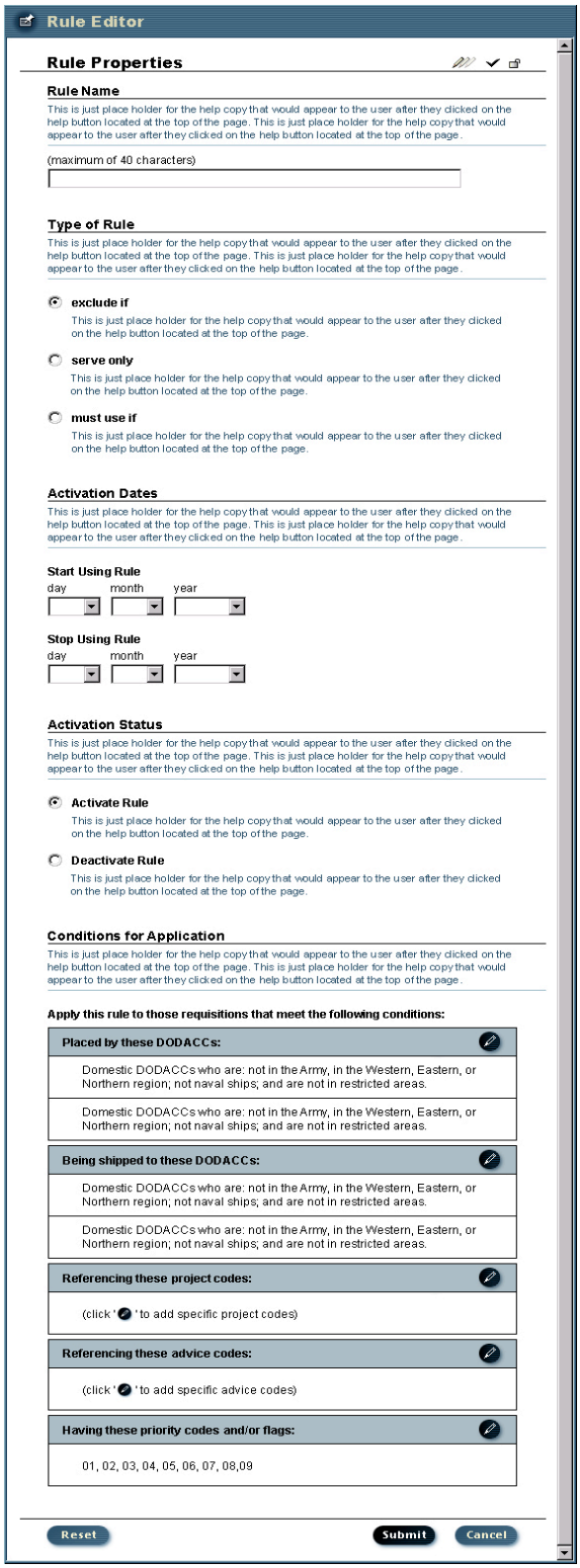

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Call Numbers
First Available Call Number
This is a description. This is a description. This is a description.
First Available Call Number
2000
Last Available Call Number
This is a description. This is a description. This is a description.
Last Available Call Number
3000
Reset
Submit
Cancel
DLA Staff
Payment
EDI Fields
Internet
(18)


http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Contract Se
Contract Il
Types of S
Contract D
DLA Staff
Vendor Inf
SAMMS Re
SAMMS Fi
Contract M
Payment
Call Numb
EDI Fields
Add a new
EDI Fields
Trading Partner Id
This is a description. This is a description. This is a description.
Trading Partner Id
193847593
Expected EDI Transactions
This is a description. This is a description. This is a description.
850
850-855
850-856
850-855-856
Reset
Submit
Cancel
EDI Fields
Internet
(19)

(22)


Create a Highly Managed Contract - Microsoft Internet Explorer
Contract
Geographies
Rules
Preferences
Contract SP0700-96-D-9711 | v5 [draft]
Contract Data
Items: 01/23/99
Items: 04/04/99
5
Items: 09/06/99
Contract items for this period
All Items Group (12,549)
Group 2 Name (5,467)
Group 3 Name (4,223)
Group 4 Name (1.201)
Sub-Group 4.1 Name (2)
Group 5 Name (3)
Deleted Items Group
Items in Group 4 Name
Show
All (1,200)
Narrowed List (200)
Item ID-Nomenclature
Item ID-Nomenclature
Customer request unit of issue
XX
Unit of issue conversion factor
XX.XXXXXX
Delivery order unit of issue
XX
Delivery order multiple quantity
XXXX
Rule Type : This would be the name of the Rule
active
This would be some kind of summary of the rule that is meaningful to the user and would indicate how this rule is being applied to the contract.
Edit Rule
Delete Ru
Move Rul
Copy Rul
Rule Type : This would be the name of the Rule
inactive
This would be some kind of summary of the rule that is meaningful to the user and would indicate how this rule is being applied to the contract.
Rule Type : This would be the name of the Rule
This would be some kind of summary of the rule that is meaningful to the user and would indicate how this rule is being applied to the contract.
add new rule

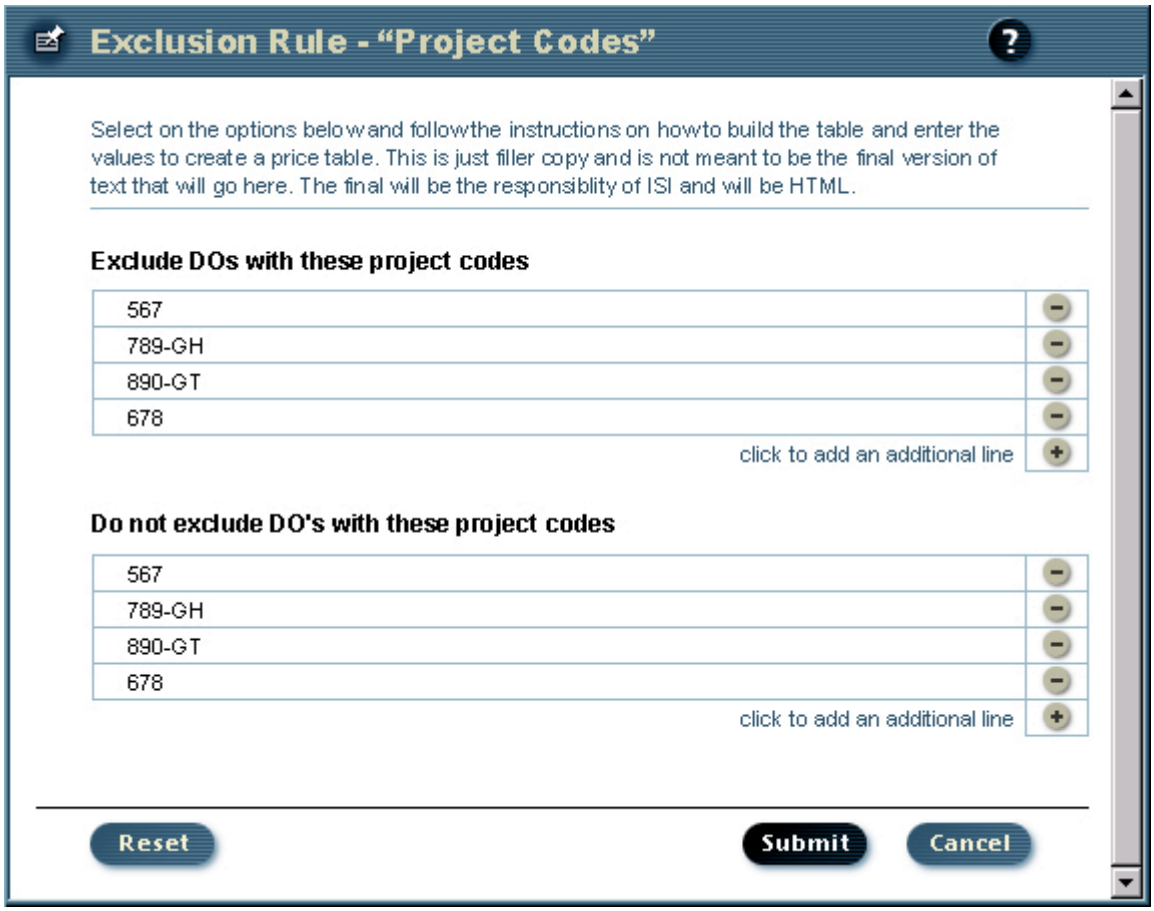
(23)


Rule Editor
Rule Properties
Rule Name
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page. This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
(maximum of 40 characters)
Type of Rule
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page. This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
exclude if
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
serve only
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
must use if
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Activation Dates
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page. This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Start Using Rule
day
month
year
Stop Using Rule
day
month
year
Activation Status
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page. This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Activate Rule
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Deactivate Rule
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Conditions for Application
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page. This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Apply this rule to those requisitions that meet the following conditions:
Placed by these DODACCs:
Domestic DODACCs who are: not in the Army, in the Western, Eastern, or Northern region; not naval ships; and are not in restricted areas.
Domestic DODACCs who are: not in the Army, in the Western, Eastern, or Northern region; not naval ships; and are not in restricted areas.
Being shipped to these DODACCs:
Domestic DODACCs who are: not in the Army, in the Western, Eastern, or Northern region; not naval ships; and are not in restricted areas.
Domestic DODACCs who are: not in the Army, in the Western, Eastern, or Northern region; not naval ships; and are not in restricted areas.
Referencing these project codes:
(click ' ' to add specific project codes)
Referencing these advice codes:
(click ' ' to add specific advice codes)
Having these priority codes and/or flags:
01, 02, 03, 04, 05, 06, 07, 08,09
Reset
Submit
Cancel

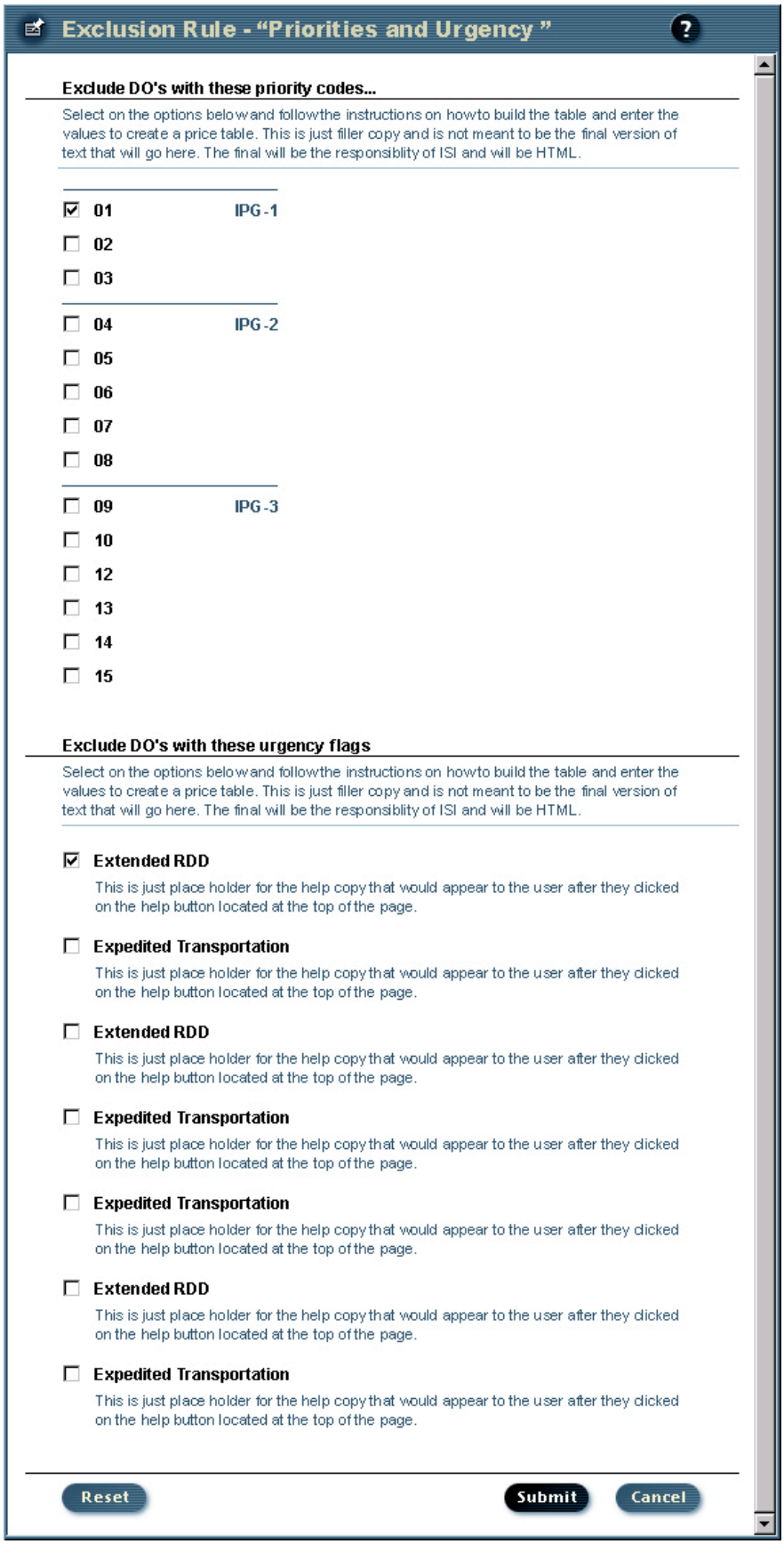

Exclusion Rule - "Placed by" Field
Type of DODACCs:
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
Exclude DOs placed by DODACCs who...
remove this phrase
...are Domestic DODACCs.
are not in the Army.
are in the Western, Eastern, or Northern Region of the Heavy Equipment Geography.
are not Naval Ships.
and are not in Restricted Areas.
click to add an additional line
Also Exclude DOs placed by DODACCs who...
remove this phrase
...are Domestic DODACCs.
are not in the Army.
are in the Western, Eastern, or Northern Region of the Heavy Equipment Geography.
are not Naval Ships.
and are not in Restricted Areas.
click to add an additional line
add new phrase
Specific DODACCs:
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
Always Exclude DOs placed by DODACCs who...
1234567890-DD-6789-98
Toxic Waste DODACCs
1234567890-DD-6789-98
1234567890-DD-6789-98
click to add an additional line
Never Exclude DOs placed by DODACCs who...
Special Forces DODACCs
1234567890-DD-6789-98
DODACCs in the state of Iowa
1234567890-DD-6789-98
click to add an additional line
Reset
Submit
Cancel

(23a)

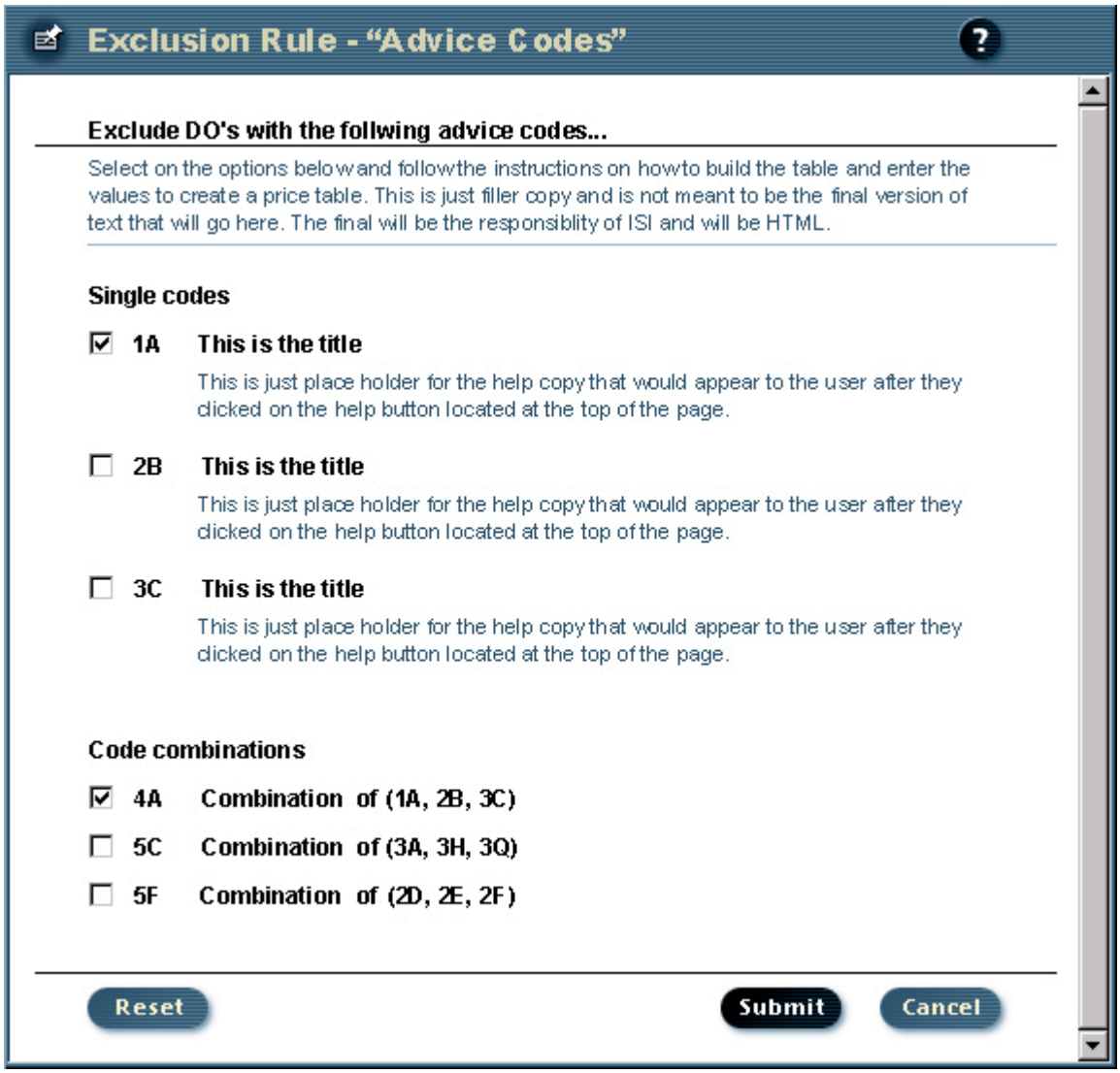
(24)


Exclusion Rule - "Project Codes"
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
Exclude DOs with these project codes
567
789-GH
890-GT
678
click to add an additional line
Do not exclude DO's with these project codes
567
789-GH
890-GT
678
click to add an additional line
Reset
Submit
Cancel

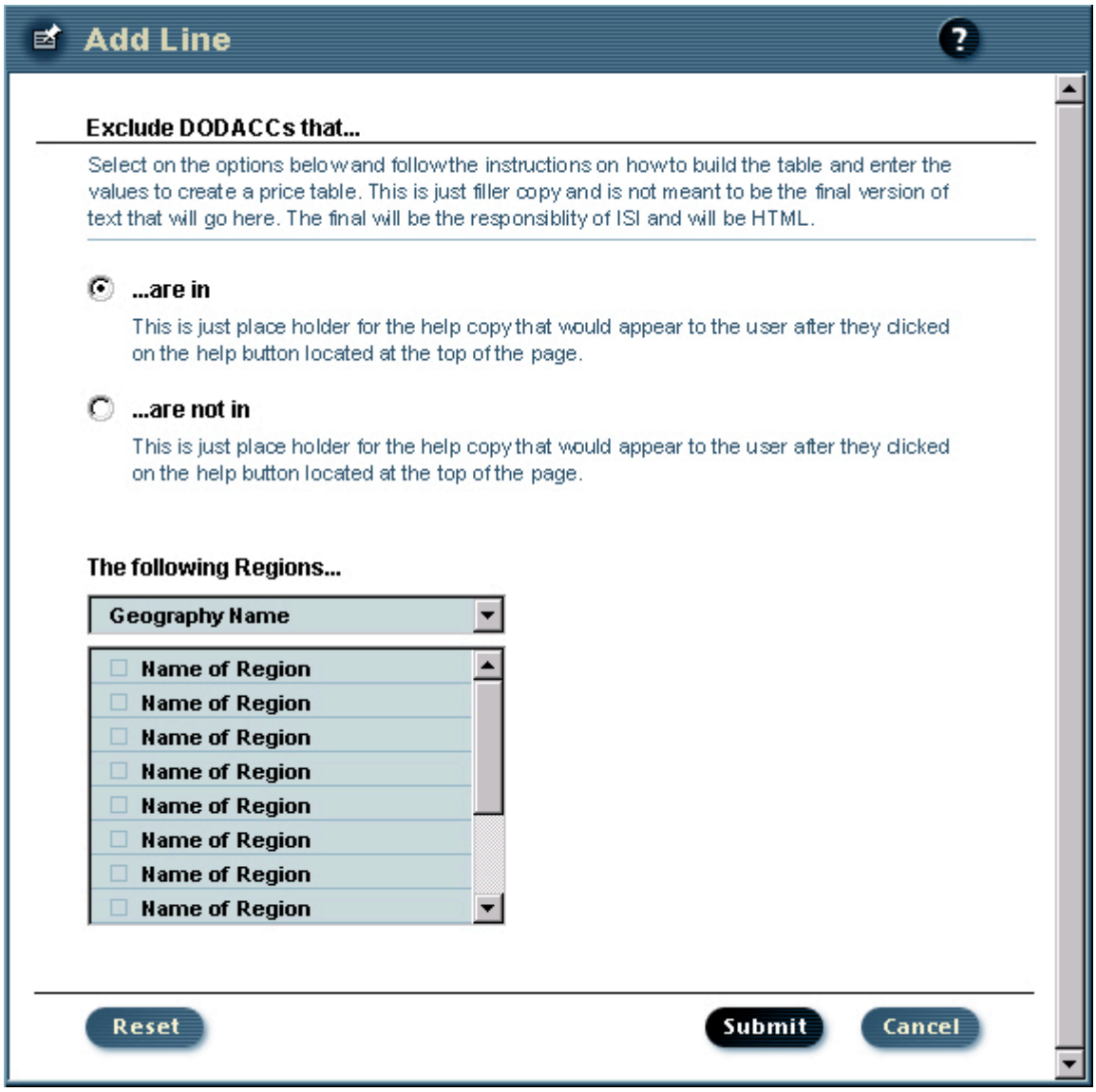
(25)


Exclusion Rule - "Priorities and Urgency "
Exclude DO's with these priority codes...
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
01 IPG-1
02
03
04 IPG-2
05
06
07
08
09 IPG-3
10
12
13
14
15
Exclude DO's with these urgency flags
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
Extended RDD
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Expedited Transportation
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Extended RDD
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Expedited Transportation
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Expedited Transportation
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Extended RDD
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Expedited Transportation
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Reset
Submit
Cancel

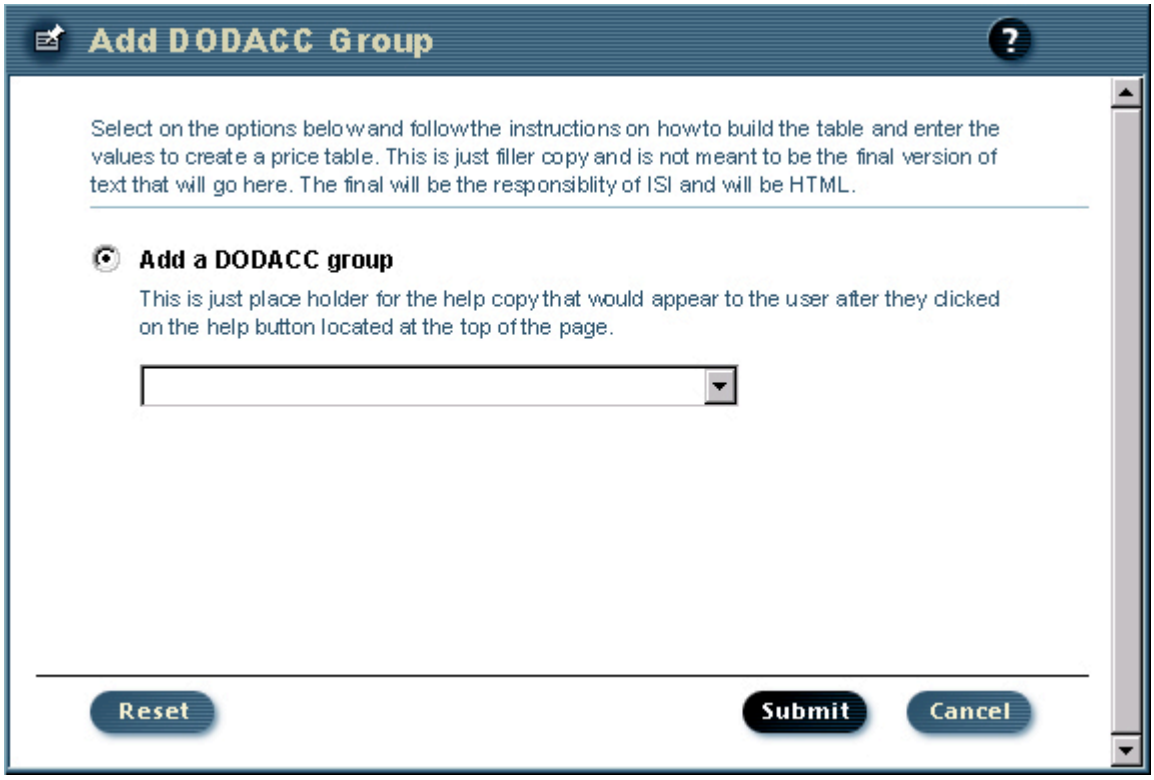
(26a)

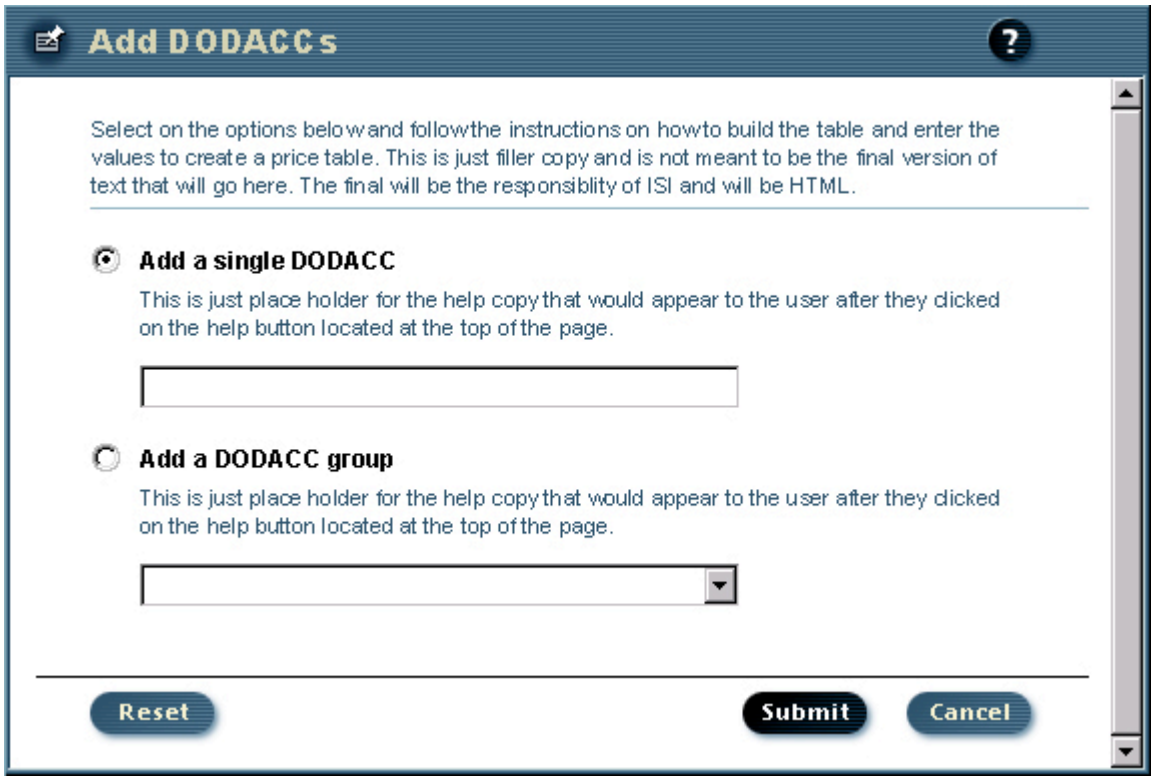

Exclusion Rule - "Advice Codes"
Exclude DO's with the follwing advice codes...
Select on the options below and followthe instructions on howto build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
Single codes
1A This is the title
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
2B This is the title
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
3C This is the title
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Code combinations
4A Combination of (1A, 2B, 3C)
5C Combination of (3A, 3H, 3Q)
5F Combination of (2D, 2E, 2F)
Reset
Submit
Cancel

(26)

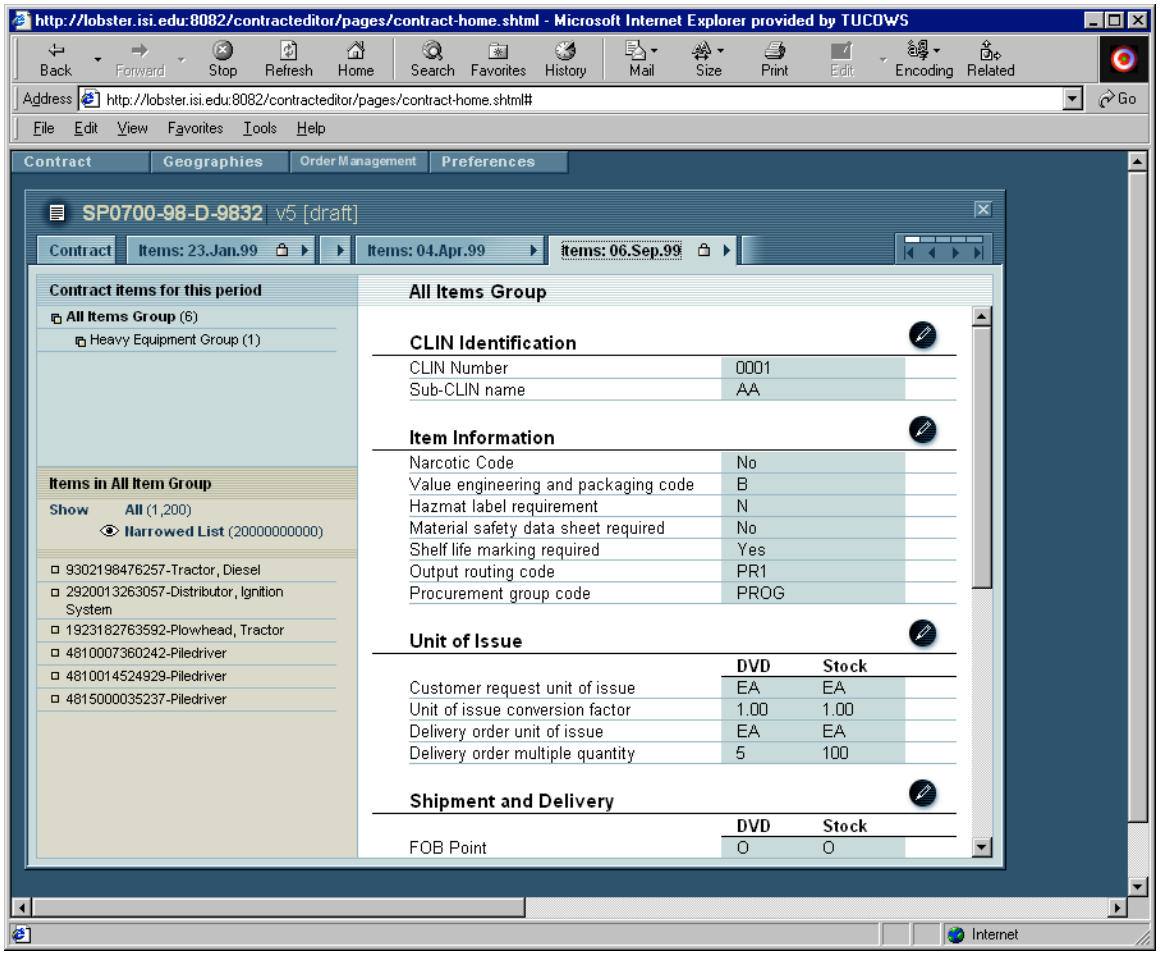

Add Line
Exclude DODACCs that...
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
...are in
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
...are not in
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
The following Regions...
Geography Name
Name of Region
Name of Region
Name of Region
Name of Region
Name of Region
Name of Region
Name of Region
Name of Region
Reset
Submit
Cancel

(27) * (28)

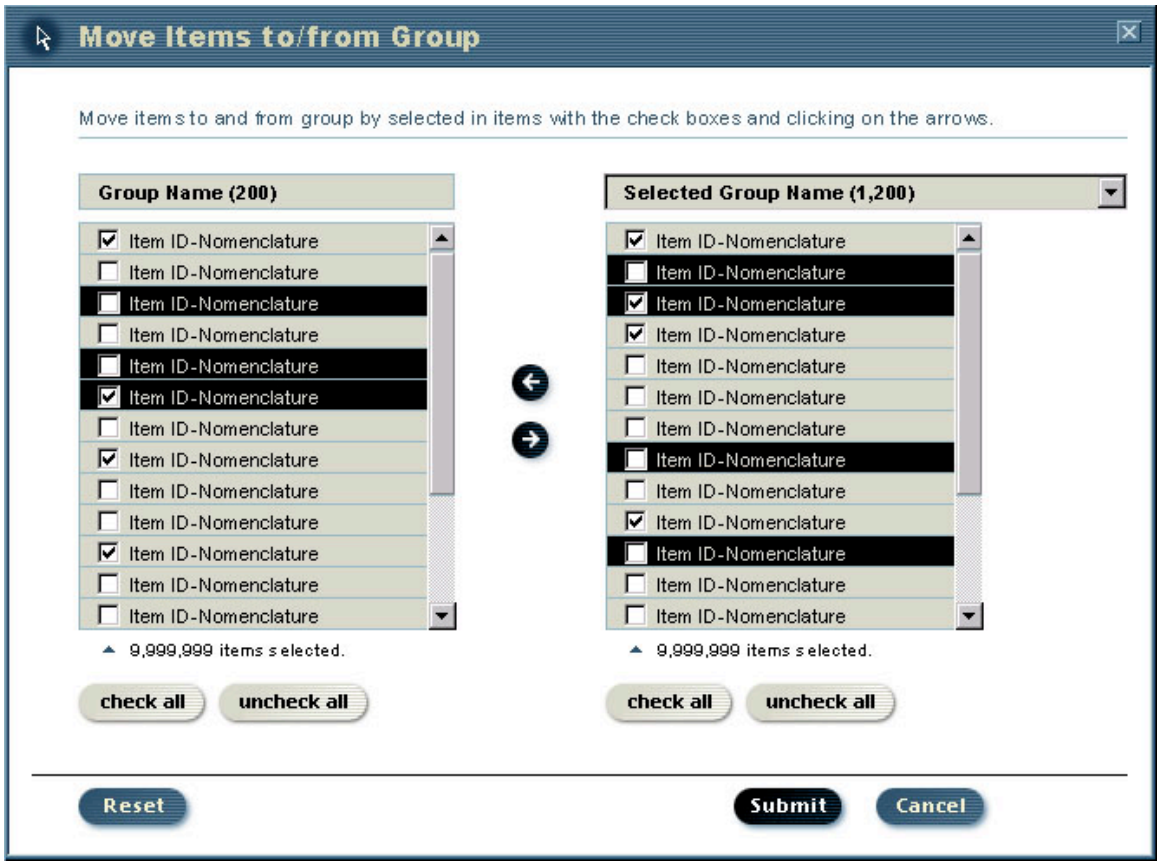

Add DODACC Group
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
Add a DODACC group
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Reset
Submit
Cancel


Add DODACCs
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
Add a single DODACC
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Add a DODACC group
This is just place holder for the help copy that would appear to the user after they clicked on the help button located at the top of the page.
Reset
Submit
Cancel

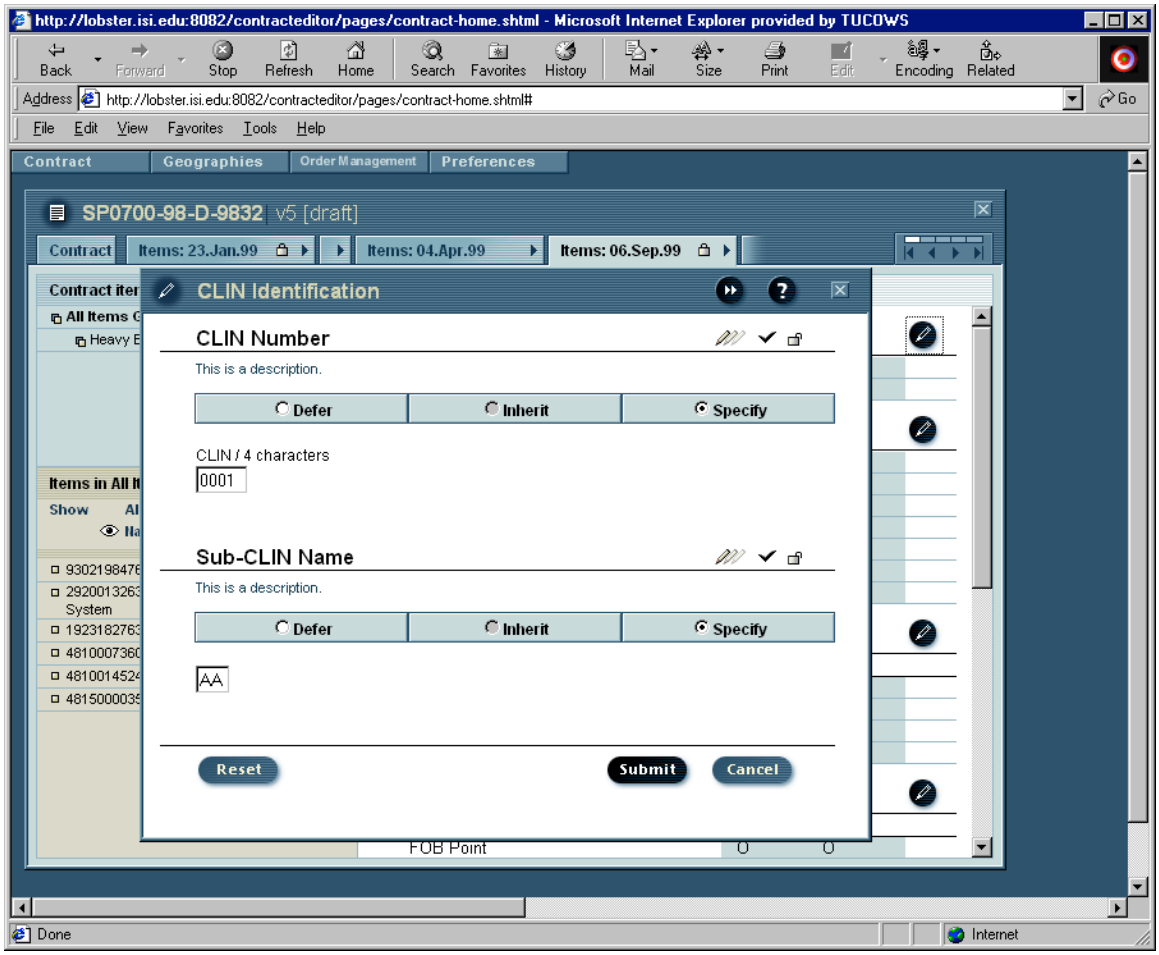
(29)


http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Contract items for this period
All Items Group (6)
Heavy Equipment Group (1)
Items in All Item Group
Show
All (1,200)
Narrowed List (20000000000)
9302198476257-Tractor, Diesel
2920013263057-Distributor, Ignition System
1923182763592-Plowhead, Tractor
4810007360242-Piledriver
4810014524929-Piledriver
4815000035237-Piledriver
All Items Group
CLIN Identification
CLIN Number 0001
Sub-CLIN name AA
Item Information
Narcotic Code No
Value engineering and packaging code B
Hazmat label requirement N
Material safety data sheet required No
Shelf life marking required Yes
Output routing code PR1
Procurement group code PROG
Unit of Issue
DVD Stock
Customer request unit of issue EA EA
Unit of issue conversion factor 1.00 1.00
Delivery order unit of issue EA EA
Delivery order multiple quantity 5 100
Shipment and Delivery
DVD Stock
FOB Point O O
Internet

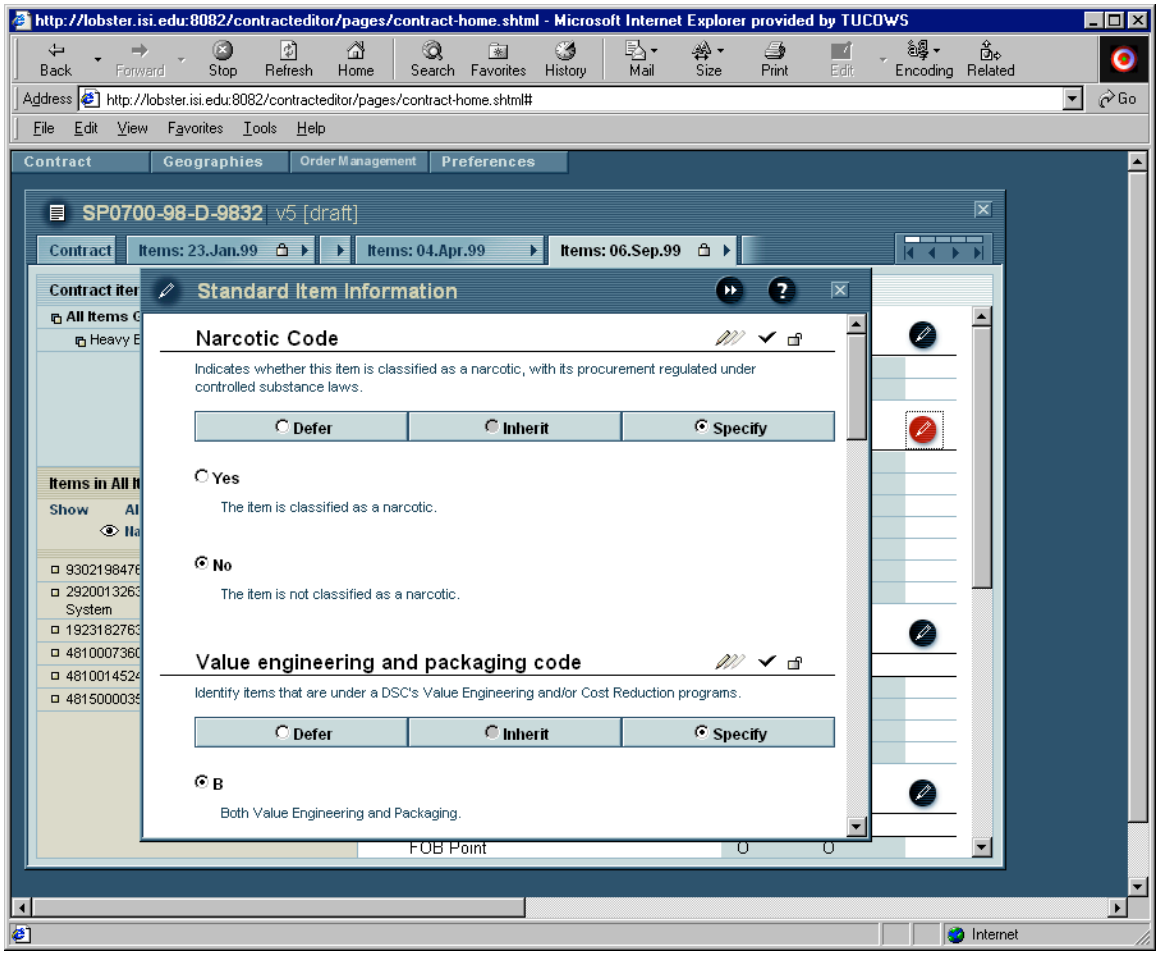
(31)


Move Items to/from Group
Move items to and from group by selected in items with the check boxes and clicking on the arrows.
Group Name (200)
Item ID-Nomenclature
9,999,999 items selected.
check all
uncheck all
Selected Group Name (1,200)
Item ID-Nomenclature
9,999,999 items selected.
check all
uncheck all
Reset
Submit
Cancel

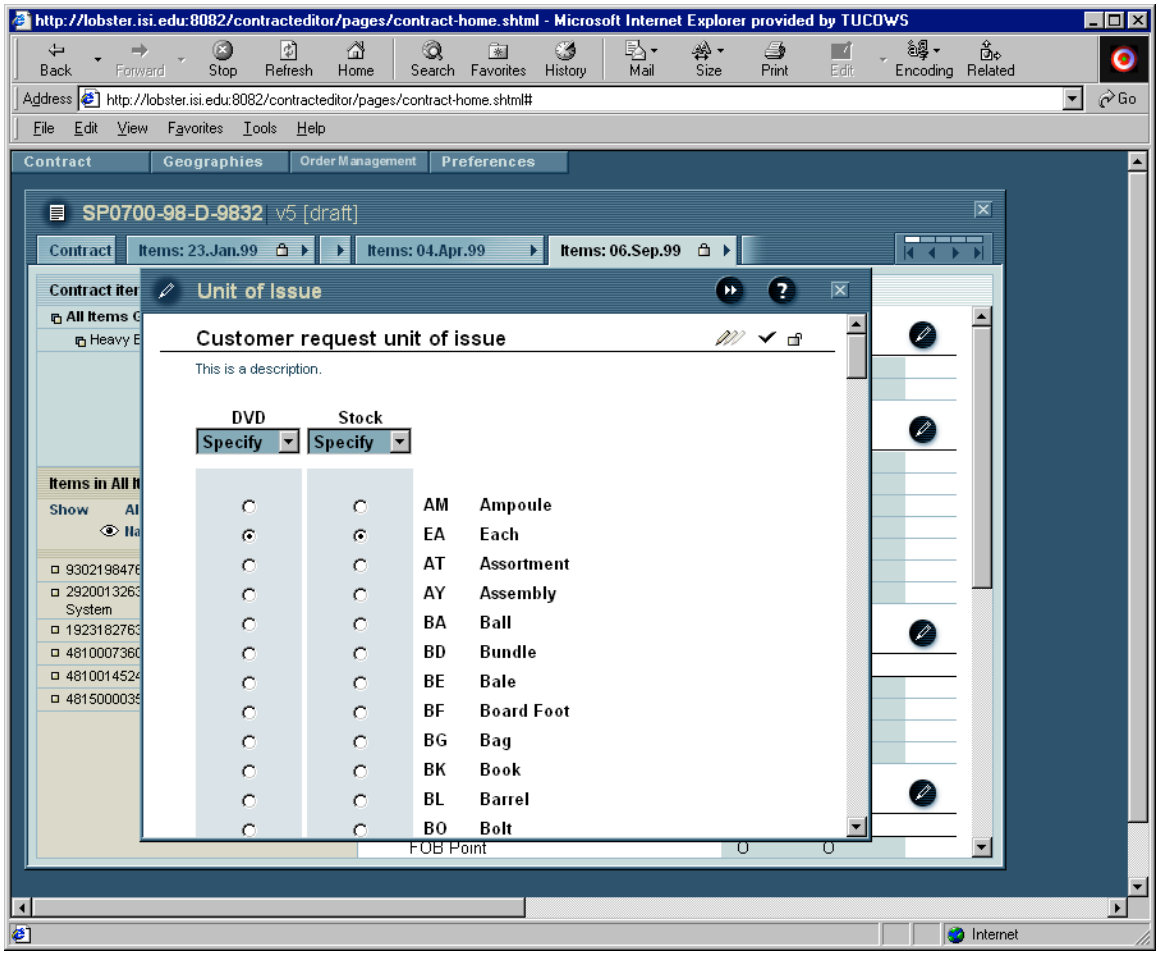
(33)


http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
CLIN Identification
CLIN Number
This is a description.
Defer
Inherit
Specify
CLIN / 4 characters
0001
Sub-CLIN Name
This is a description.
Defer
Inherit
Specify
AA
Reset
Submit
Cancel
FOB Point
Done
Internet

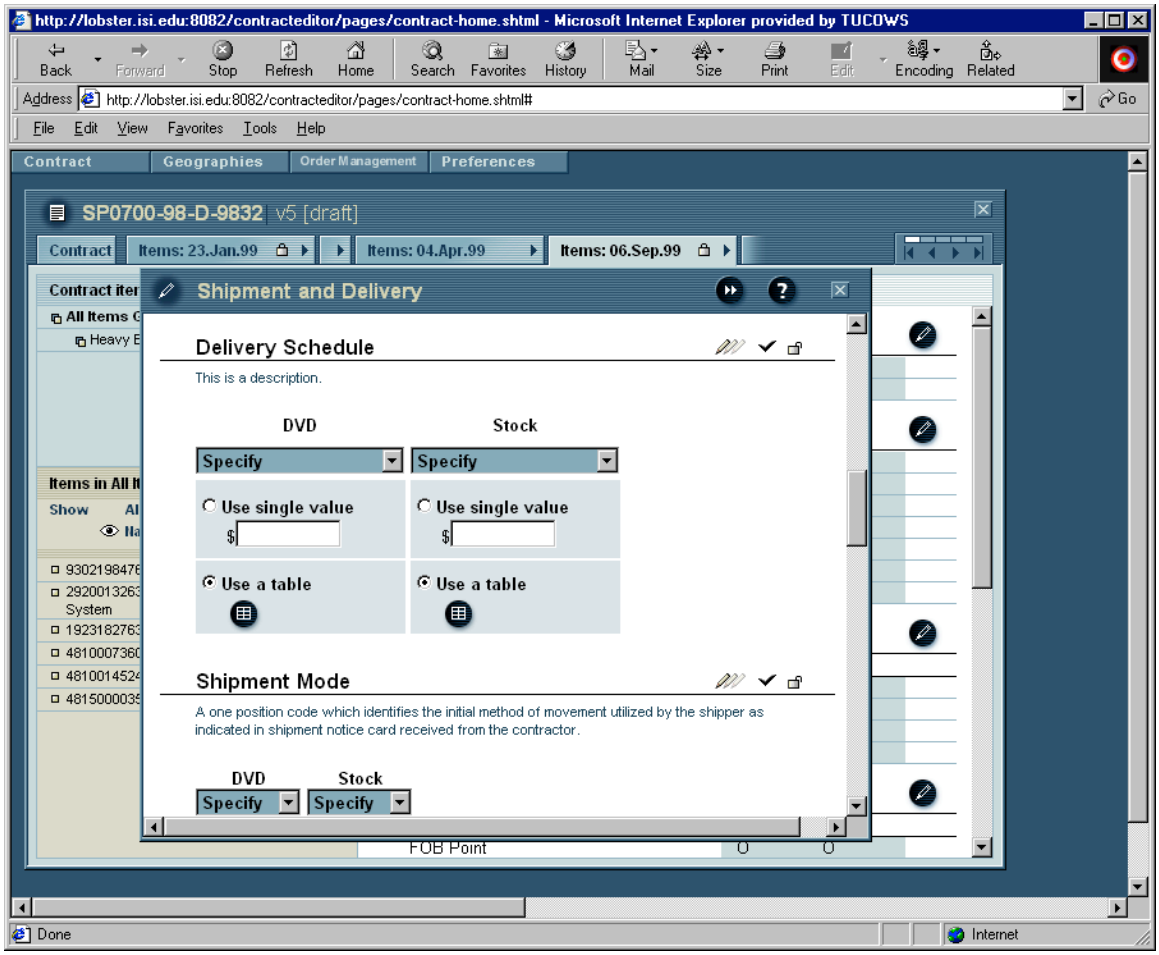
(34)


http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Standard Item Information
Narcotic Code
Indicates whether this item is classified as a narcotic, with its procurement regulated under controlled substance laws.
Defer
Inherit
Specify
Yes
The item is classified as a narcotic.
No
The item is not classified as a narcotic.
Value engineering and packaging code
Identify items that are under a DSC's Value Engineering and/or Cost Reduction programs.
Defer
Inherit
Specify
B
Both Value Engineering and Packaging.
FOB Point
Internet

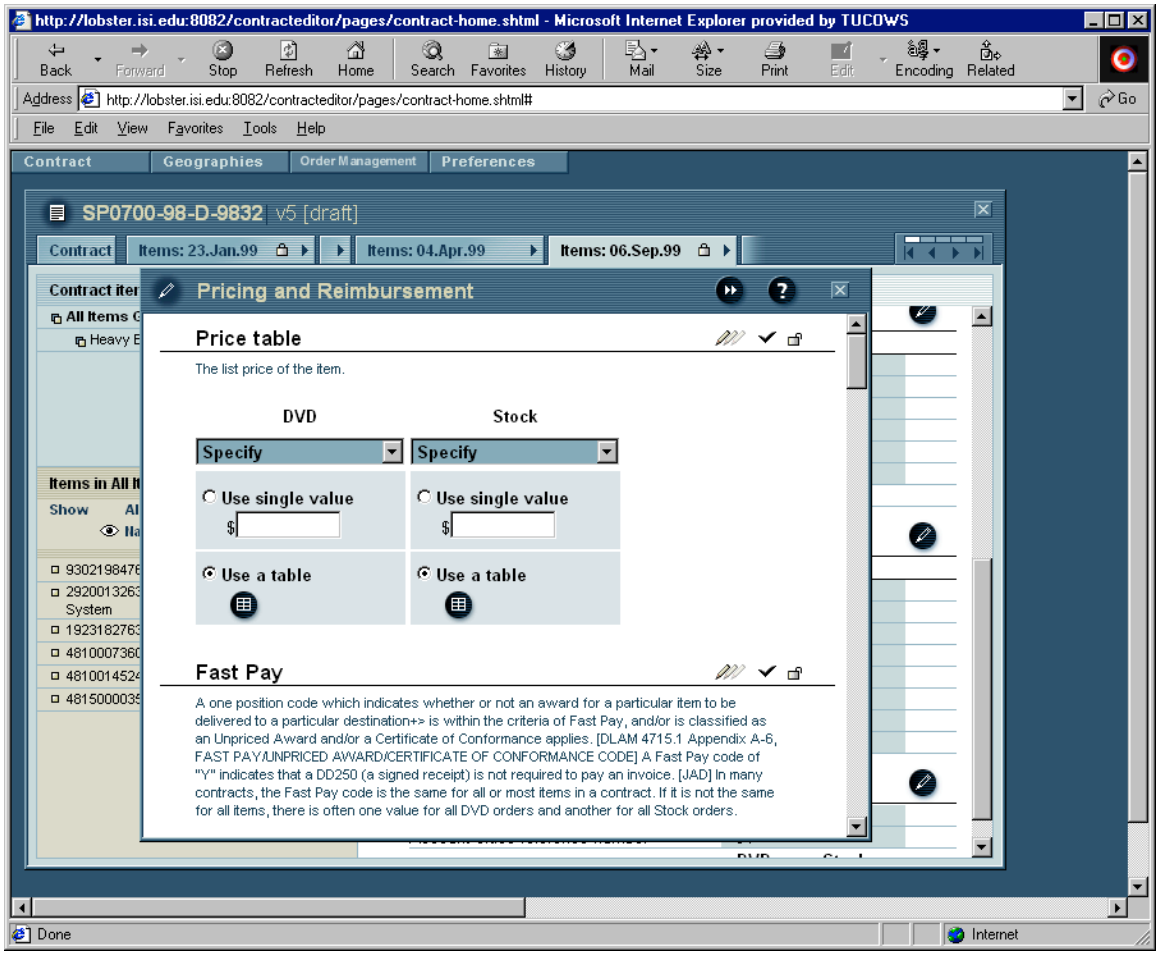
(35)

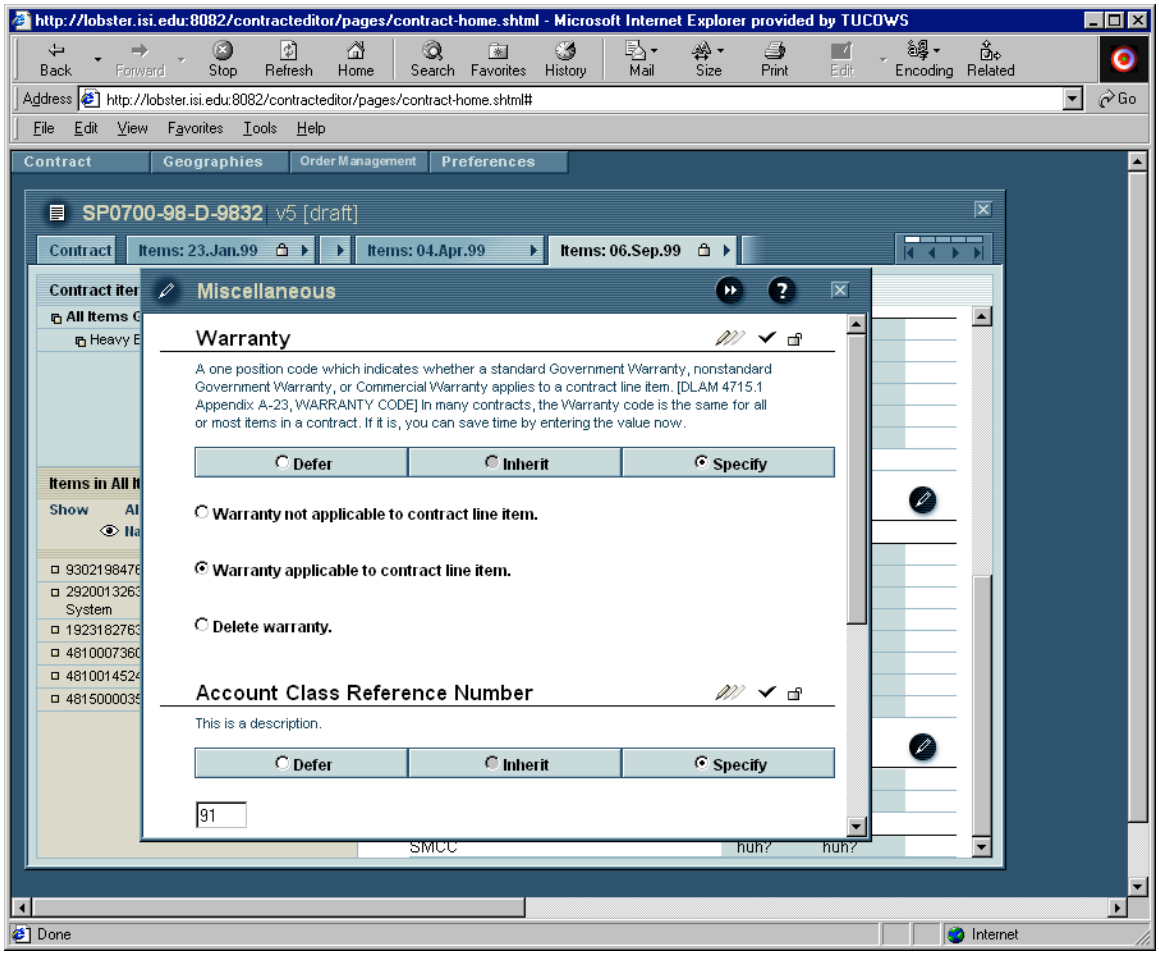

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Unit of Issue
Customer request unit of issue
This is a description.
DVD
Stock
Specify
Specify
AM Ampoule
EA Each
AT Assortment
AY Assembly
BA Ball
BD Bundle
BE Bale
BF Board Foot
BG Bag
BK Book
BL Barrel
BO Bolt
FOB Point
Internet

(36)


http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Shipment and Delivery
Delivery Schedule
This is a description.
DVD
Stock
Specify
Specify
Use single value
$
Use single value
$
Use a table
Use a table
Shipment Mode
A one position code which identifies the initial method of movement utilized by the shipper as indicated in shipment notice card received from the contractor.
DVD
Stock
Specify
Specify
FOB Point
Done
Internet

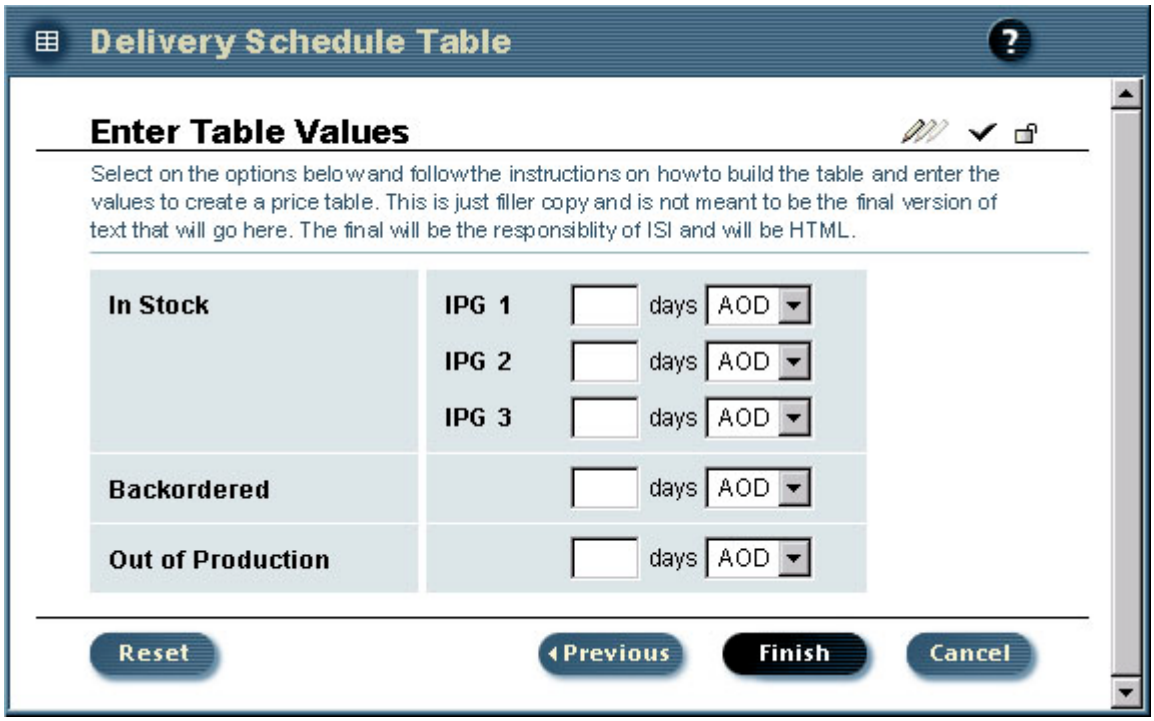
(37)


http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Pricing and Reimbursement
Price table
The list price of the item.
DVD
Stock
Specify
Specify
Use single value
$
Use single value
$
Use a table
Use a table
Fast Pay
A one position code which indicates whether or not an award for a particular item to be delivered to a particular destination+> is within the criteria of Fast Pay, and/or is classified as an Unpriced Award and/or a Certificate of Conformance applies. [DLAM 4715.1 Appendix A-6, FAST PAY/UNPRICED AWARD/CERTIFICATE OF CONFORMANCE CODE] A Fast Pay code of "Y" indicates that a DD250 (a signed receipt) is not required to pay an invoice. [JAD] In many contracts, the Fast Pay code is the same for all or most items in a contract. If it is not the same for all items, there is often one value for all DVD orders and another for all Stock orders.
Done
Internet

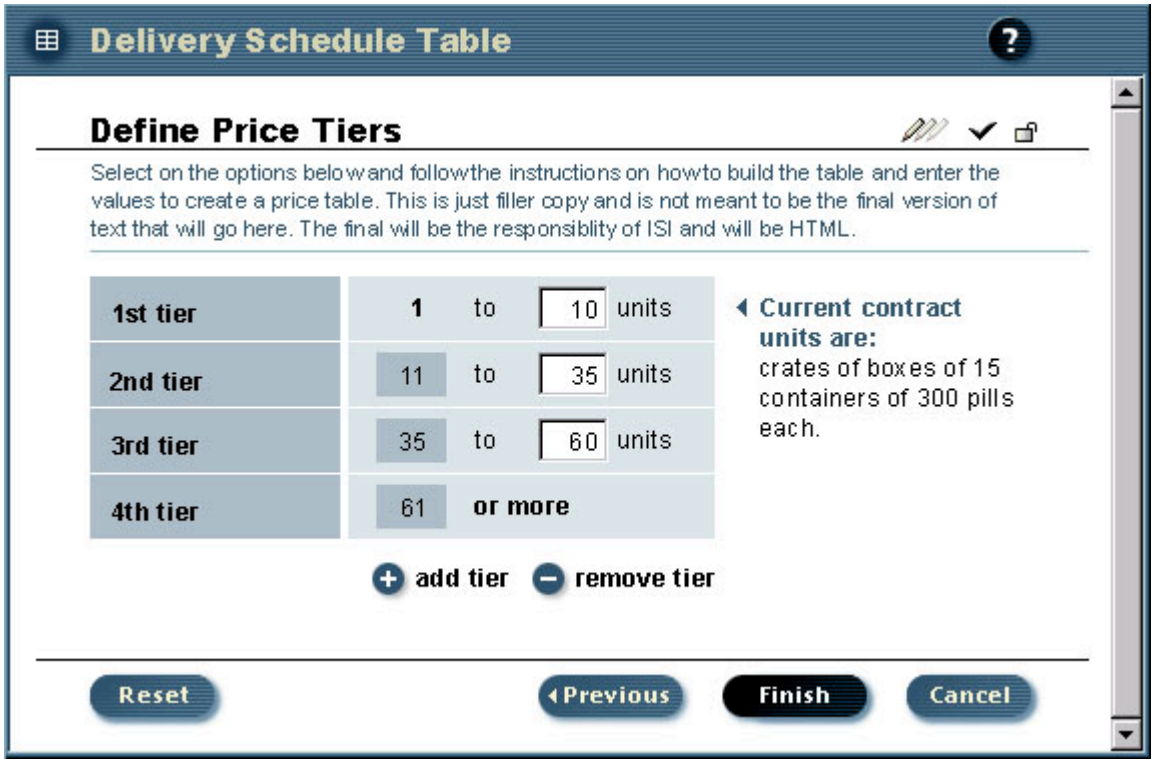

http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml - Microsoft Internet Explorer provided by TUCOWS
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Size
Print
Edit
Encoding
Related
Address
http://lobster.isi.edu:8082/contracteditor/pages/contract-home.shtml#
Go
File
Edit
View
Favorites
Tools
Help
Contract
Geographies
Order Management
Preferences
SP0700-98-D-9832 v5 [draft]
Contract
Items: 23.Jan.99
Items: 04.Apr.99
Items: 06.Sep.99
Miscellaneous
Warranty
A one position code which indicates whether a standard Government Warranty, nonstandard Government Warranty, or Commercial Warranty applies to a contract line item. [DLAM 4715.1 Appendix A-23, WARRANTY CODE] In many contracts, the Warranty code is the same for all or most items in a contract. If it is, you can save time by entering the value now.
Defer
Inherit
Specify
Warranty not applicable to contract line item.
Warranty applicable to contract line item.
Delete warranty.
Account Class Reference Number
This is a description.
Defer
Inherit
Specify
91
SMCC
huh?
huh?
Done
Internet

(38)

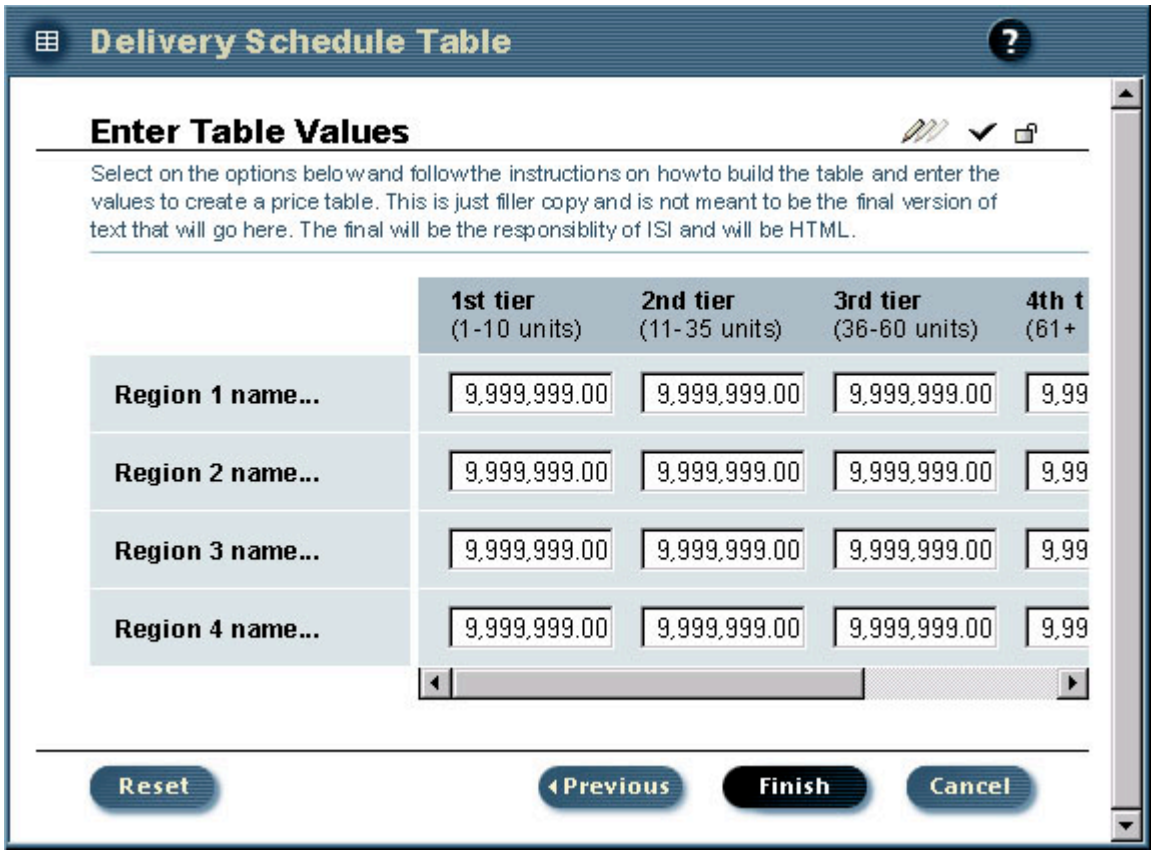
(36b)


Delivery Schedule Table
Enter Table Values
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
In Stock
IPG 1
days
AOD
IPG 2
days
AOD
IPG 3
days
AOD
Backordered
days
AOD
Out of Production
days
AOD
Reset
Previous
Finish
Cancel

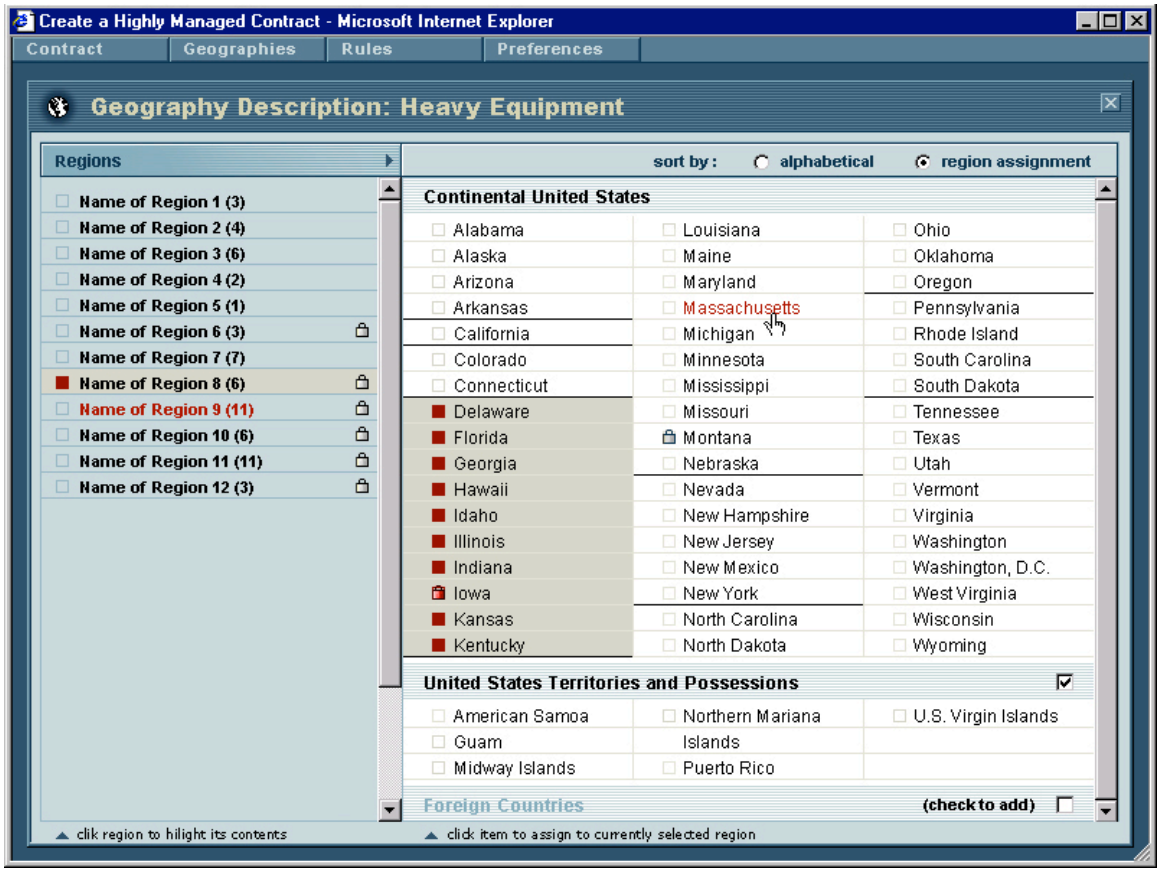
(37a)

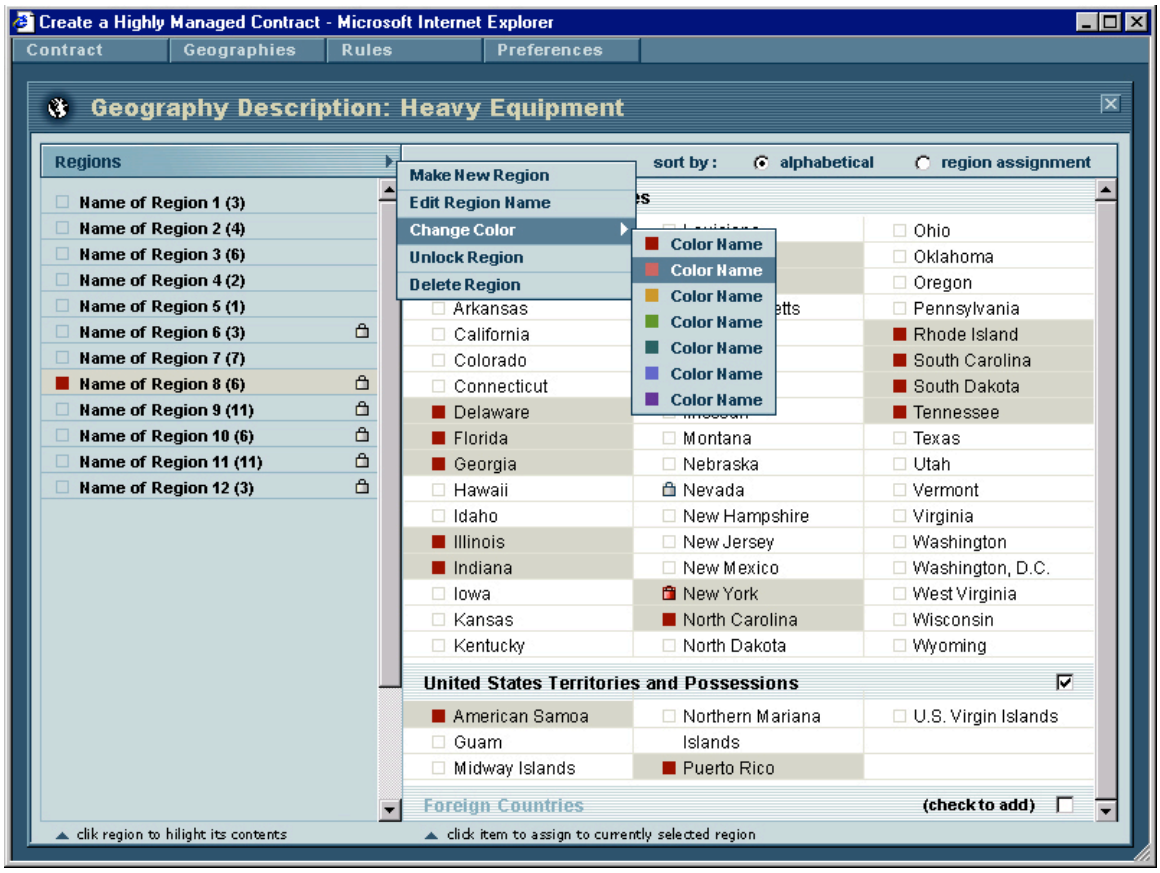

Delivery Schedule Table
Define Price Tiers
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
1st tier
1 to 10 units
2nd tier
11 to 35 units
3rd tier
35 to 60 units
4th tier
61 or more
add tier
remove tier
Current contract units are:
crates of boxes of 15 containers of 300 pills each.
Reset
Previous
Finish
Cancel

(37b)


Delivery Schedule Table
Enter Table Values
Select on the options below and follow the instructions on how to build the table and enter the values to create a price table. This is just filler copy and is not meant to be the final version of text that will go here. The final will be the responsiblity of ISI and will be HTML.
1st tier (1-10 units)
2nd tier (11-35 units)
3rd tier (36-60 units)
4th t (61+
Region 1 name...
Region 2 name...
Region 3 name...
Region 4 name...
9,999,999.00
9,99
Reset
Previous
Finish
Cancel

(43)


Create a Highly Managed Contract - Microsoft Internet Explorer
Contract
Geographies
Rules
Preferences
Geography Description: Heavy Equipment
Regions
sort by:
alphabetical
region assignment
Name of Region 1 (3)
Name of Region 2 (4)
Name of Region 3 (6)
Name of Region 4 (2)
Name of Region 5 (1)
Name of Region 6 (3)
Name of Region 7 (7)
Name of Region 8 (6)
Name of Region 9 (11)
Name of Region 10 (6)
Name of Region 11 (11)
Name of Region 12 (3)
Continental United States
Alabama
Alaska
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
Florida
Georgia
Hawaii
Idaho
Illinois
Indiana
Iowa
Kansas
Kentucky
Louisiana
Maine
Maryland
Massachusetts
Michigan
Minnesota
Mississippi
Missouri
Montana
Nebraska
Nevada
New Hampshire
New Jersey
New Mexico
New York
North Carolina
North Dakota
Ohio
Oklahoma
Oregon
Pennsylvania
Rhode Island
South Carolina
South Dakota
Tennessee
Texas
Utah
Vermont
Virginia
Washington
Washington, D.C.
West Virginia
Wisconsin
Wyoming
United States Territories and Possessions
American Samoa
Guam
Midway Islands
Northern Mariana Islands
Puerto Rico
U.S. Virgin Islands
Foreign Countries
(check to add)
clik region to hilight its contents
click item to assign to currently selected region


Create a Highly Managed Contract - Microsoft Internet Explorer
Contract
Geographies
Rules
Preferences
Geography Description: Heavy Equipment
Regions
Name of Region 1 (3)
Name of Region 2 (4)
Name of Region 3 (6)
Name of Region 4 (2)
Name of Region 5 (1)
Name of Region 6 (3)
Name of Region 7 (7)
Name of Region 8 (6)
Name of Region 9 (11)
Name of Region 10 (6)
Name of Region 11 (11)
Name of Region 12 (3)
clik region to hilight its contents
Make New Region
Edit Region Name
Change Color
Unlock Region
Delete Region
Color Name
Color Name
Color Name
Color Name
Color Name
Color Name
Color Name
sort by :
alphabetical
region assignment
Arkansas
California
Colorado
Connecticut
Delaware
Florida
Georgia
Hawaii
Idaho
Illinois
Indiana
Iowa
Kansas
Kentucky
Montana
Nebraska
Nevada
New Hampshire
New Jersey
New Mexico
New York
North Carolina
North Dakota
Ohio
Oklahoma
Oregon
Pennsylvania
Rhode Island
South Carolina
South Dakota
Tennessee
Texas
Utah
Vermont
Virginia
Washington
Washington, D.C.
West Virginia
Wisconsin
Wyoming
United States Territories and Possessions
American Samoa
Guam
Midway Islands
Northern Mariana Islands
Puerto Rico
U.S. Virgin Islands
Foreign Countries
(check to add)
click item to assign to currently selected region