Image Cover Sheet

CLASSIFICATION	SYSTEM NUMBER
UNCLASSIFIED	507695

TITLE

MESSAGE HANDLING TOOLKIT INTEGRATION IN A DREV R&D ACTIVITY

System Number:

Patron Number:

Requester:

Notes:

DSIS Use only:

Deliver to:

Defence Research Establishment Valcartier

MESSAGE HANDLING TOOLKIT

INTEGRATION IN A DREV R&D ACTIVITY

Presented to:

Director General, DREV
2459 Boulevard Pie XI North
Val-Bélair, Québec
G3J 1X5

Attention: Mr. Jean-Claude Labbé (Scientific Authority)

March 1998

SSC Solicitation No.:	XSK W7701-6-2842/A
CGI File:	181-075

PREFACE

This document contains the results of a study on the possible integration of the Message Handling Toolkit (MHT)[1] application in a DREV R&D activity. Most particularly, the aim of the study was to evaluate the costs and technically investigate the use of the MHT application in a DREV R&D activity called "Integration of heterogeneous sources providing non real-time information for the Global Command & Control System". The integration of the MHT in any other R&D activity could certainly benefit from the results of this study, but could imply further investigation. Comments or requests concerning this document should be submitted to:

Jean-Claude Labbé
Defence Research Establishment Valcartier
jean-claude.labbe@drev.dnd.ca

THIS ISSUE	NAME	SIGNATURE	DATE
SUBMITTED BY	*HARVEY, Gaétane*	Gaétane Harvey	26-3-98
	DUCHESNE, Michel	Michel Duchesne	26/3/98
REVIEWED/ APPROVED BY	*LABBÉ, Jean-Claude*	JMabbé	26/3/98

[1] For a complete description of the MHT, the reader is asked to consult the references listed in Section 2.2

Table of Contents

1. INTRODUCTION

1.1 Scope of Study

From its participation in the NATO Data Fusion Demonstrator (DFD) project over the last five years, DREV gained a great deal of experience with military message handling. In the same period of time, the Information System Technology section at DREV actively participated in the definition and capture of requirements for future Command and Control Information Systems (CCIS). It appears from this requirement definition activity that the handling of structured messages such as ADatP-3, USMTF and OTH T Gold is of major importance for the three Canadian Force (CF) environments (Air Force, Army and Navy). In parallel, research and development (R&D) activities that often involve the development of prototypes to experiment with concepts and technologies also require the use of structured military messages.

It was then important to reuse the effort invested in the DFD project and to propose related activities that could first help or provide support to R&D activities and second guide decision-makers in their choice for a message handling capability. Amongst these activities, the reuse of components developed within the NATO DFD project and the extend of its message handling capabilities to provide a set of tools called the Message Handling Toolkit (MHT) for the manipulation of ADatP-3 and USMTF messages was completed.

The aim of the present study is to pursue the work already done with the MHT and to investigate its integration in a DREV R&D activity. Most particularly, this study concerns the costs and the technical evaluation concerning the integration of the MHT in a DREV R&D activity called "Integration of heterogenous sources providing non real-time information for the GCCS (Global Command and Control System) ". Even if the study concerns a specific R&D activity, it is hoped that other activities will benefit from its results.

1.2 The Message Handling Toolkit (MHT) Application

The MHT was developed at DREV between January and September 1997. The application is issued from work performed for the NATO DFD to support military message handling. The main purpose of the MHT application is to provide a set of tools enabling a user to manipulate ADatP-3 and USMTF messages. Structured messages like ADatP-3 and USMTF offer many advantages to the military users. One of these advantages is that structured formats ensure that messsage contents can be correctly understood and interpreted by the different users. For that reason, structured messages become the way to exchange information between different military entities, mainly during national and international tactical military exercises or operations.

On the other hand, it can be very tedious to write such a kind of messages if the user is not assisted by appropriate tools. The formats are not easy to remember and the preparation of messages is difficult. To prepare a single message, the user must often navigate through several manuals to get the complete picture of its format.

The MHT provides facilities to enable users to create their own national formats or to use standard NATO ADatP-3 or USMTF formats by using a friendly graphical interface tool. It includes the required functionality to correctly prepare formatted messages and to automatically validate them if the corresponding formats have been previously created using the Message Format Editor. The MHT is a stand-alone application, but it can be integrated with other client applications by using the MHT application library. The application library contains a set of functions allowing developers to access the contents of messages created with the MHT and to integrate them into their own operational databases.

The MHT is currently available on a Sun Solaris platform, but there are possibilities to easily port it on any other platforms such as PC Windows NT an HP-UX.

1.3 Document Overview

- Chapter 2 lists the documents used or referred to during the study.
- Chapter 3 presents the detailed study on the integration of the MHT in the DREV activity.
- Chapter 4 includes a brief conclusion on the study.
- Chapter 5 presents the list of acronyms and abbreviations used in this document.

2. REFERENCED DOCUMENTS

2.1 Government Documents

The following government documents should be consulted to facilitate understanding of the study.

- Over-The-Horizon Targeting Gold, Operational Specification, OS-OTG (Rev B), October 1994, Unclassified
- NATO Message Text Formatting System (FORMETS), ADatP-3 Part I., Change 3, System Concept and Description, October 1993, Unclassified
- NATO Message Text Formatting System (FORMETS), ADatP-3 Part II, Change 9.0, Catalogue of Standard Message Text Formats, February 1995, Unclassified
- NATO Message Text Formatting System (FORMETS), ADatP-3 Part III, Change 9.0, Catalogue of Standard Set Formats, February 1995, Unclassified
- NATO Message Text Formatting System (FORMETS), ADatP-3 Part IV, Change 9.0, Catalogue of Standard Field Formats, February 1995, Unclassified
- United States Message Transfer Format (USMTF) System Standards, AFP 102, Unclassified

2.2 Non-Government Documents

The following non-government documents should be consulted to facilitate understanding of the study.

- Using the Message Handling Toolkit, Users' Guide, Version 1.0, November 1997, Unclassified
- Message Handling Toolkit, Software Design Document, Version 1.0, November 1997, Unclassified
- Message Handling Toolkit, Application Programming Interface, Version 1.0, November 1997, Unclassified
- Message Handling Toolkit, Models and Data Dictionary, Version 1.0, November 1997, Unclassified
- Message Handling Toolkit, Project Manual and Development Plan, Version 2.0, April 1997, Unclassified
- A Flexible Message Handling Toolkit, DREV TM-9740, December 1997, Unclassified

3. INTEGRATION OF THE MHT

3.1 Background

The aim of the study being the integration of the MHT in a DREV R&D activity called "Integration of heterogeneous sources providing non real-time information for the Global Command and Control System (GCCS)", it is of particular interest to know how the MHT application could help in achieving this R&D activity and the costs that would be involved.

For this particular activity, the following needs have been identified:

- Be able to validate and decompose some USMTF messages (e.g. TACELINT, LOCATOR, PURPLE, etc.) and some OTH T Gold messages (e.g. GOLD, JMCIS GOLD, etc.).
- Be able to access the decomposed messages to get information of interest prior to fuse information.
- Be able to compose new messages in any of the above message formats based on the information fusion results.
- Run the target application on the HP-UX platform.

3.2 Integration Issues

Given the requirements identified above, some additions and modifications are needed to the MHT application in order to get the expected results.

- The current MHT version is only able to process ADatP-3 and USMTF message formats. Thus, it must be enhanced so to provide the additional capabilities required to process the OTH T Gold message formats.
- Only the ADatP-3 "ENEMY SITREP" message format is currently populated in the MHT Message Format Database. So, this database shall be populated with the new required message formats.
- The only platform currently supporting the MHT is Sun-Solaris 2.5. The whole MHT software shall then be moved to the HP-UX platform, but in a first step, the MHT application will be used directly on the Sun Solaris platform.
- In the integration phase, it could be possible that some functions (ex.: query methods) will be missing at the MHT API level. The API contains a library of methods that return all the possible information on sets and fields contained in a message, but there is no specific method to return for example "all the coordinates contained in a message". This kind of request is specific to some message formats and is not currently covered by the API. Therefore, there is a possibility for adding new methods to the MHT API as needed by the DREV R&D activity.

3.3 Technical Steps to Use the MHT

The present section provides a list of instructions to follow for supporting the concerned DREV R&D activity with respect to its message handling capability (see Section 3.1).

The user is interested in using the MHT application in a transparent way with respect to the validation, decomposition and recomposition steps instead of using the Message Validation Tool (MVT) as proposed by the MHT application. To that end, it is required to build an application that calls directly the appropriate MHT functions to read USMTF, ADatP-3 or OTH T Gold messages as input, to validate and decompose these messages, to query the results of the decomposition as needed (retrieve appropriate fields in messages), to merge some information elements (fusion process), and then to recompose the results of the fusion process into a new OTH T Gold formatted message. The following shaded paragraph presents a description in pseudo-code of the main steps to follow.

```
Open the message formats database;
ForEach message
        Validate and decompose the message string
End ForEach;
Query and analyze the decomposed message with the help of the API;
Execute any calculation and fusion operations as needed;
Build a new message
        Get the appropriate message template from the template list;
        Set the value of the appropriate fields of the message template;
        Recompose the new message's string
End Build a new message;
Do any further processing with the newly built message;
Close the message formats database.
```

The list of templates can be built with the help of the Message Editor (ME) application and then decomposed with the help of the Message Validation Tool (MVT) that are part of the MHT application (see Section 2.2 of document entitled "Using the Message Handling Toolkit"). These steps are required to create the appropriate formats of the recomposed messages. The Message Editor is first used to create messages of many different formats. These messages can contain different variety of sets and segments of any format, as well as pre-defined values for some the fields.

A typical program written in C++ that demonstrates the use of the MHT library functions is provided in Annex A. This program applies to the targeted DREV R&D application according to the needs expressed in Section 3.1. The recompose function developed for the DFD is included and can readily be used.

3.4 Cost Estimates

3.4.1 Detailed Costs

The cost estimates presented hereafter concern the addition of the OTH T Gold formatting rules and the transfer of the MHT application from a Sun Solaris platform to a HP-UX platform.

For a maritime environment, the MHT is useless if it does not allow a user to create and manipulate OTH T Gold message formats. To support these rules, it is required to modify the MHT message parser and its interfaces. The addition of the OTH T Gold rules could be done on the Sun platform before being ported

on any other platform. This approach would allow to have the same functionality on all platforms without any additional effort during the conversion process. The particularities of the OTH T Gold rules compared to those of the ADatP-3 or USMTF rules are listed in Annex B.

The development costs associated with the inclusion of the OTH T Gold rules in the MHT are estimated to **60 person/days of effort or $ 33K (Cdn).**

As far as the transfer of the MHT on the HP-UX platform is concerned, there is a need to recompile the whole application in the new environment. More specifically, it is necessary to buy the HP-UX development software for the ObjectStore database and for the ILOG interface. A C++ library used for the development of the API must also be acquired. Table 1 provides detailed information concerning the software acquisition. The corresponding quotations are provided in Annex C.

Companies	Products	Costs (US$)
ILOG Contact: Graig Schumpert (Phone: 703-351-9005)	ILOG Views Development Licence for HP-UX Annual Maintenance (15%)	 $ 10 000. $ 1 500.
OBJECT DESIGN Contact: Brian Murphy (Phone: 781-674-5116)	ObjectStore DB Development Licence for HP-UX - 1 client Annual Maintenance (15%)	 $ 4 950. $ 742.
ROGUE WAVE	Tools.h++ 7.0.7 Support	$ 495. $ 195.
TOTAL		**$ 17 882. (US) or $ 25K (Cdn)**

Table 1. Software acquisition costs to support the DREV R&D activity

In addition to the software development and acquisition costs, it is necessary to add integration effort that include the following activities:

- import the MHT software from the Sun environment to the targeted HP-UX platform;
- review the "makefiles" used in the Sun environment and modify them as needed to the new environment; and
- recompile and test the whole application into the new environment.

The costs related to these activities appear in Table 2.

Activities	Effort (p/d)	Approx. Costs (Cdn$)
INTEGRATION		
❑ Import programs from Sun to HP-UX platform	2	$ 1 100.
❑ Update the Sun "makefile" environment in the new environment	6	$ 3 300.
❑ Compile and test the whole application	4	$ 2 200.
TOTAL	**12**	**$ 6 600. (Cdn)**

Table 2. Integration costs to support the DREV R&D activity

The cost estimates for the DREV R&D activity do not include the following:

- The effort to be spent to populate the Message Formats Database with the new required formats such as, for example, TACELINT, LOCATOR, PURPLE, and some OTH T Gold message formats such as GOLD, JMCIS GOLD). These formats can be easily populated as part of the R&D activity itself by using the Message Format Editor of the MHT application.

- The effort to be spent to program the required client applications (as shown in the example C++ program - Annex A). The client applications need to make the appropriate calls to API methods for querying the decomposition file and retrieving the information of interest. These elements of information contained in different messages are then merged (fusion process) by using the recompose function and become a new OTH T Gold message. The C++ program in the example contained in Annex A is a framework for the client application, but needs to be modified and upgraded to include the specific code of the targeted application.

3.4.2 Summary of Costs

The costs involved to modify the MHT in order to support the DREV R&D activity are estimated to $64.6K (Cdn), which include $33K to add the OTH T Gold capability, $25K for HP-UX software acquisition, and $6.6K for integration.

Additional costs will nevertheless be involved to populate the message Format Database with the formats required by the R&D project and to develop a certain number of client applications, but such activities are normally consider as being part of the project itself.

One should note that the costs to add the OTH T Gold capability ($33K) could be shared with project MCOIN III, which is also interested in such a feature.

4. CONCLUSION

This study has demonstrated that the integration of the MHT in a R&D activity can be done at a reasonably low costs. Having integrated the MHT in the targetted DREV R&D activity (Integration of heterogenous sources providing non real-time information for the Global Command and Control System), the integration costs to meet the requirements of other R&D activities would afterwards become minimal. This because the MHT will already support the three main families of message format (AdatP-3, USMTF and OTH T Gold) and will be running on most probably three different platforms (Sun Solaris, HP-UX and PC Windows NT). For most of the R&D activities, the only price to pay would be the one to populate the Message Format Database with the appropriate formats (very few for a given R&D activity) and to develop the required client applications using the appropriate MHT API methods.

One of the reason that makes the MHT attractive for R&D projects is that it allows to build personalized message formats (not pre-defined, but for the purposes of a particular need) to meet specific requirements. These messages must nevertheless be built using the same rules as for ADatP-3, or USMTF or OTH T Gold.

5. ACRONYMS AND DEFINITIONS

ADatP-3	Allied Data Publication 3
API	Application Programming Interface
CCIS	Command & Control Information System
COTS	Commercial Off The Shelf
DB	Database
DBMS	Database Management System
DFD	Data Fusion Demonstrator
DREV	Defence Research Establishment Valcartier
GCCS	Global Command and Control System
MHT	Message Handling Toolkit
MTF	Message Text Format
MTFID	Message Text Format Identifier
OTH T GOLD	Over-The-Horizon Targeting Gold
R&D	Research and Development
USMTF	United States Message Text Format

ANNEX A

Use of some MHT library functions through an example

Example of a typical program that uses some MHT library functions (validation, decomposition, API query methods, recomposition)

```
// Standard
#include <fstream.h>

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

//Rogue Wave
#include <rw/pstream.h>
#include <rw/slistcol.h>
#include <rw/compiler.h>
#undef RW_COMPILE_INSTANTIATE
#include <rw/tpslist.h>

// api classes
#include "apiField.hh"
#include "apiMessage.hh"
#include "apiMessageElement.hh"
#include "apiMessageList.hh"
#include "apiSet.hh"
#include "apiSegment.hh"

// parser classes
#include "parserField.hh"
#include "parserFieldGroup.hh"
#include "parserSet.hh"
#include "parserSegment.hh"
#include "parserMessage.hh"
#include "mfError.hh"
#include "mfField.hh"
#include "mfFieldGroup.hh"
#include "mfMessageFormatDatabase.hh"
#include "mfMessageTextFormat.hh"
#include "mfSegment.hh"
#include "mfColumnarSet.hh"
#include "mfLinearSet.hh"
#include "mfTypes.hh"
#include "decompMessage.hh"

#include "apiMessageList.hh"


void dummy_forcelink()
{
  parserField parser_field;
  parserFieldGroup parser_field_group( 1, 0 );
  parserSet parser_set;
  parserSegment parser_segment( 1 );
  parserMessage parser_message;

  mfField mf_field( MF_OC_MANDATORY, 0 );
  mfFieldGroup mf_field_group;
  mfLinearSet mf_linear_set( MF_OC_MANDATORY, 0 );
  mfColumnarSet mf_columnar_set( MF_OC_MANDATORY, 0 );
  mfSegment mf_segment( MF_OC_MANDATORY, 0 );
}

main( int argc, char *argv[] )
{
  apiMessageList *message_template_list;
  apiMessage *message_template;
  apiMessage *api_message;
  apiSet *api_set;
  apiField *api_field;
  int set_number;
  int field_number;
  decompMessage *decomposed_message;
  mfMessageFormatDatabase messageFormatDatabase;
```

```
mfStandard standard;  // Standard of a message after a validation (e.g. ADATP-3)
RWCString format;     // Format of a message after a validation (e.g. "ENEMY SITREP")
RWCString diagnostic; // Diagnostic message after a validation
int valid_message;    // Validation status result
RWTPtrSlist<apiSet> location_set_list;
RWTPtrSlist<apiField> field_list;
RWSlistCollectables *rw_list;


// Contents of an enemy situation report message (example)
char *message_string ="\
MSGID/ENEMY SITREP/RADGS/028//\n\
RPTTYPE/ENSIT//\n\
EFDT/040706Z/JUL//\n\
EGROUP/U0017/ORC//\n\
LOCATION/REAL/HEAD/-/-/-/POINT/32UQC018562//\n\
SOURCE/C3/AIRRECCE//\n\
TIME/AT/040700ZJUL//\n\
LOCATION/REAL/END/-/-/-/POINT/32UPD039724/32UQC025563//\n\
ACTIVITY/REAL/MOVE//\n";

// print the message to standard output
cout << endl << message_string << endl;

// open the message formats database
messageFormatDatabase.setDatabaseFilePath( "./MessageFormats.osdb" );
messageFormatDatabase.open();

// check if the database has been correctly opened
if( !messageFormatDatabase.isOpen() )
  {
    // The database has not been opened. Exit the application
    cout << "Unable to open \"./MessageFormats.osdb\"." << endl;
    exit( -1 );
  }

// validate and decompose a message
valid_message =
messageFormatDatabase.validateMessage( message_string, standard, format, diagnostic,
                                       &decomposed_message );

// print the diagnostic message to the standard output
cout << diagnostic << endl;

// check if the message is valid
if( valid_message )
  {
    // The message is valid and has been correctly decomposed
    api_message = decomposed_message->decompToApi();
    delete decomposed_message;  // delete it since no longer needed

     // Execute any required operation (calculation,fusion,etc.) with the api_message
     // object
     // For example, the following lines extracts all coordinates in the message
     // and prints them to the standard output.

    location_set_list.clear();
    cout << "GRID LOCATIONS: ";

    // Get all the LOCATION sets in the message
    api_message->getSet( location_set_list, "LOCATION" );

    // Iterate over each LOCATION set to get the coordinates
    for( set_number = 0; set_number < location_set_list.entries(); set_number++ )
      {
        // get the LOCATION set at position set_number
        api_set = location_set_list.at( set_number );

        // empty the field list
        field_list.clear();

        // get all fields located at position 7, actually the coordinate field
        api_set->getField( field_list, 7 );
```

```
        // Iterate over each coordinate field
        for( field_number = 0; field_number < field_list.entries(); field_number++ )
          {
            // get the coordinate field at position field_number
            api_field = field_list.at( field_number );

            // print the field content to the standard output
            cout << api_field->getFieldString() << " ";
          }
      }

    cout << endl;

    // The following step is to build a new message from the fusion results.

    // First, get the template messages stored in a templates decomposition file
    message_template_list = apiMessageList::readFromFile( "templates.df" );

    if( !message_template_list )
      {
        cout << "The file \"templates.df\" doesn't exist" << endl;
        exit( -1 );
      }

    // get the actual rogue wave list object inside the message_template_list
    rw_list = message_template_list->getMessageList();

    // get the appropriate message template from the list
    message_template = ( apiMessage* )rw_list->at( 0 );

    // modify fields as needed inside the message template
    // by using the api objects methods

    // recompose the string of the message template
    message_template->recompose();

    // do any further processing with the newly built message
}

  return 0;
}
```

ANNEX B

Specifications on the OTH T Gold formatting rules

Major syntactical differences between the OTH T GOLD (OTG) formatting rules and the ADatP-3 or USMTF formatting rules:

- The <Message Text Format Identifier> field included in the set MSGID is located at position number 2 instead of position number 1 as it is in ADatP-3 rules.
- An OTG message always starts with a MSGID set and finishes with an ENDAT set.
- A set name has 3 to 5 letters while a set name for ADatP-3 or USMTF has 3 to 8 letters.
- The only free-text sets are RMKS and NARR and they may be used only where specifically permitted in the format.
- An empty field must not have an hyphen character "-" in it.
- In specific situations, mandatory fields and field markers may be omitted. Such mandatory fields have a "no data" mention in the set specification.
- The hyphen character is used only as an internal field delimiter in field 2 of the "CTC" set and may not be used anywhere else.
- The RMKS set may be used at any location in a message instead of appearing only at the end of the message as it is the case for ADatP-3 rules.
- There may be checksums at the end of a field or a sub-field.
- The instructions for each field will indicate whether checksums are required or permitted (eg. 020230Z7, 7 represents the checksum of the preceding characters).
- Special handling instructions may be included in the security classification set.
- The maximum length for a message is one hundred (100) lines of sixty-nine(69) characters each.
- The ordering of sets within a segment may not be fixed if expressly stated in the message format.

ANNEX C

COTS Products – Price List

ILOG Quotation

To: Gaetane Harvey
CGI

Phone: (418) 844-4675
Fax: (418) 844-4538

Fr: Craig Schumpert
ILOG Inc.
2525 Wilson Blvd.
Arlington, VA 22201

Phone: (703) 351-9005
Fax: (703) 351-7775

Quote # 970220
Date: 2/20/98

Item:	Product	Qty	Unit Price	Cost
	ILOG License			
1	*ILOG Views Development License - Win 95 , NT*	1	$6,500	$6,500.00
2	*Annual Maintenance / Technical Support*	1	$975	$975.00
3	*ILOG Views Development License - HP UX*	1	$10,000	$10,000.00
4	*Annual Maintenance / Technical Support*	1	$1,500	$1,500.00
			Total	$18,975.00

Terms and Conditions

- Price Quotation is valid for 30 days.
- Terms are Net 30 days.
- Prices are FOB Mountain View, California
- All prices are in US Dollars
- Acceptance of Shrinkwrap Software License for each ILOG software product purchased.
- Maintenance includes hotline support, email, and minor upgrades.
- Annual maintenance fee is 15% per development license list price.
- All UNIX Development licenses are floating except for ILOG Server which is priced per CPU.

ILOG C++ Software Components, Training, Maintenance, Consulting

Build interactive graphic-intensive and portable interfaces	VIEWS
Develop distributed applications	BROKER
Build real-time groupware applications	SERVER
Connect RDBMSs to C++ applications	DB LINK
Monitor data in a real-time environment	RULES
Solve resource management problems with constraint procesing	SOLVER
Solve scheduling applications	SCHEDULE

Object Design, Inc. Order Supplement

25 Mall Road Burlington, MA 01803
EL: (781) 674-5000
ax: (781) 674-5461
Brian Murphy 781-674-5116

Agreement #: Shrink Wrap
*Please reference Agreement# on customer P.O.
Quote Date: 20-Feb-98

Bill-to Address:
CGI-DREV

Ship-to Address:
CGI-DREV

Contact Name: Accounts Payable
Telephone:
Fax:
Terms: NET 30

Contact Name: Gaetane Harvey
Telephone:
Fax:
EMAIL:

Taxable? Y N (Circle one)
(Attach tax exempt certificate)
County:
Quote Exp. Date: 20-Mar-98

Sales Agent: Brian Murphy
Territory: 1-NE
Sales Admin.: Caroline B.
Customer PO #:

PRODUCT NUMBER	PRODUCT DESCRIPTION	QTY	UNIT PRICE	EXTENDED PRICE
SW-OS-NT9-NSS	On-line Runtime Server/NT	30	$4,950.00	$148,500.00
DISCOUNT	Volume Discount of 18%	1	18%	($26,700.00)
SW-OS-NT9-RTC	ObjectStore Runtime Client/NT	625	$825.00	$515,625.00
DISCOUNT	Volume Discount of 23%	1	23%	($118,593.75)
SW-OS-NT9-DVC	Development Client/NT	1	$3,050.00	$3,050.00
SW-OS-HPU-DVC	Development Client/HP	1	$4,950.00	$4,950.00
MN-ST	Hotlines & Updates for 1 Year	1	$120,982.50	$120,982.50
				$647,813.75

Special Instructions: Sofware Maintenance is priced at 15% of List Price. Participation in the Systems Integrator program will allow for discounts to be increased an additional 35% based on these volumes, which is a very common practice among our US Government Contractors. Additionally, a site license would be most cost effective and provide the government the most flexibility with respec to number of licenses and platforms for deploying the application. If this is of interest, Object Design can provide futher details when the need arises.

Object Design and the Customer agree that the products listed above are provided by Object Design to the Customer subject to the terms and conditions of the above-referenced Agreement.

Customer Acceptance
Signature
Name
Title
Date

Object Design Acceptance
Signature
Name Kevin F. Colliton
Title Director of Sales Operations
Date

North American Price List

If you are outside of our North American pricing area, please use our international price list.

Customized display, including:
Product Lines: Foundation
Platforms: Unix, OS/2, Windows, Full Platform,
Languages: C++, Java

Prices are per developer per platform.
Unless otherwise noted, support price includes product updates, and telephone customer support for one year.

Products are source code unless otherwise noted. Software is available on CD, $3\frac{1}{2}$" disks, or via the Internet. Other media may be available upon request. Products are shipped with online documentation; printed manuals are available at additional cost. Products that require Tools.h++ must be compiled with Tools.h++ source code.

ONLINE ORDER HERE DELIVERY

Foundation Products

Product	Platform	Type	Version	Item No.	Price (US$)	Notes
Tools.h++	Unix	Software	7.0.7	21101	$495	
		Support		21301	$195	
	OS/2	Software	7.0.3	22101	$495	
		Support		22301	$195	
	Windows	Software	7.0.7	24101	$495	3.1/NT/'95
		Support		24301	$195	
	Full Platform	Software	7.0.7	25101	$990	Includes all platforms, licensed to a single developer.
		Support		25301	$390	
	Printed Manual			26601	$45	

Product	Platform	Type	Version	Item No.	Price (US$)	Notes
Tools.h++ Professional Includes Tools.h++	Unix	Software	Unknown	21112	$1195	
		Support		21312	$395	
	Windows	Software	Unknown	24112	$1195	
		Support		24312	$395	
	Full Platform	Software	Unknown	25112	$2030	Includes all platforms, licensed to a single developer.
		Support		25312	$670	
	Printed Manual			26612	$75	

Product	Platform	Type	Version	Item No.	Price (US$)	Notes
	Unix	Software	1.2.2	21103	$295	
		Support		21303	$195	

#507695