

SAND2016-4960
Unlimited Release
Printed May 2016

Contingency Contractor Optimization Phase 3 Sustainment, Software Design Document Contingency Contractor Optimization Tool - Prototype

Justin D. Durfee, Christopher R. Frazier, Alisa Bandlow, Katherine A. Jones

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.



U.S. DEPARTMENT OF
ENERGY



Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: <http://www.osti.gov/scitech>

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: <http://www.ntis.gov/search>



Contingency Contractor Optimization Phase 3 Sustainment, Software Design Document Contingency Contractor Optimization Tool - Prototype

Justin D. Durfee, Christopher R. Frazier, Alisa Bandlow, Katherine A. Jones
Operations Research and Computational Analysis
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-MS1138

Abstract

This document describes the final software design of the Contingency Contractor Optimization Tool – Prototype. Its purpose is to provide the overall architecture of the software and the logic behind this architecture. Documentation for the individual classes is provided in the application Javadoc.

The Contingency Contractor Optimization project is intended to address Department of Defense mandates by delivering a centralized strategic planning tool that allows senior decision makers to quickly and accurately assess the impacts, risks, and mitigation strategies associated with utilizing contract support.

The Contingency Contractor Optimization Tool - Prototype was developed in Phase 3 of the OSD ATL Contingency Contractor Optimization project to support strategic planning for contingency contractors. The planning tool uses a model to optimize the Total Force mix by minimizing the combined total costs for selected mission scenarios. The model optimizes the match of personnel types (military, DoD civilian, and contractors) and capabilities to meet mission requirements as effectively as possible, based on risk, cost, and other requirements.

Contents

- 1. Introduction5
- 2. System Architecture5
- 3. Software Design7
 - 3.1 CCOT-P Server7
 - 3.1.1 Third-Party Libraries7
 - 3.1.2 Configuration Files7
 - 3.1.3 Database Access8
 - 3.1.4 Client9
 - 3.1.5 Servlet9
 - 3.2 Solver10
 - 3.2.1 Third-Party Libraries11
 - 3.3 Polling Application11
- References13
- Distribution15

1. INTRODUCTION

The Contingency Contractor Optimization Tool – Prototype (CCOT-P) is a web-based force planning tool. See “Contingency Contractor Optimization Tool Phase 3, Platform Requirements Document [1],” and “Contingency Contractor Optimization Tool Phase 3, Requirements Document [2],” for the system description, overview, and requirements.

2. SYSTEM ARCHITECTURE

The overall CCOT-P system architecture is shown below in Figure 1.

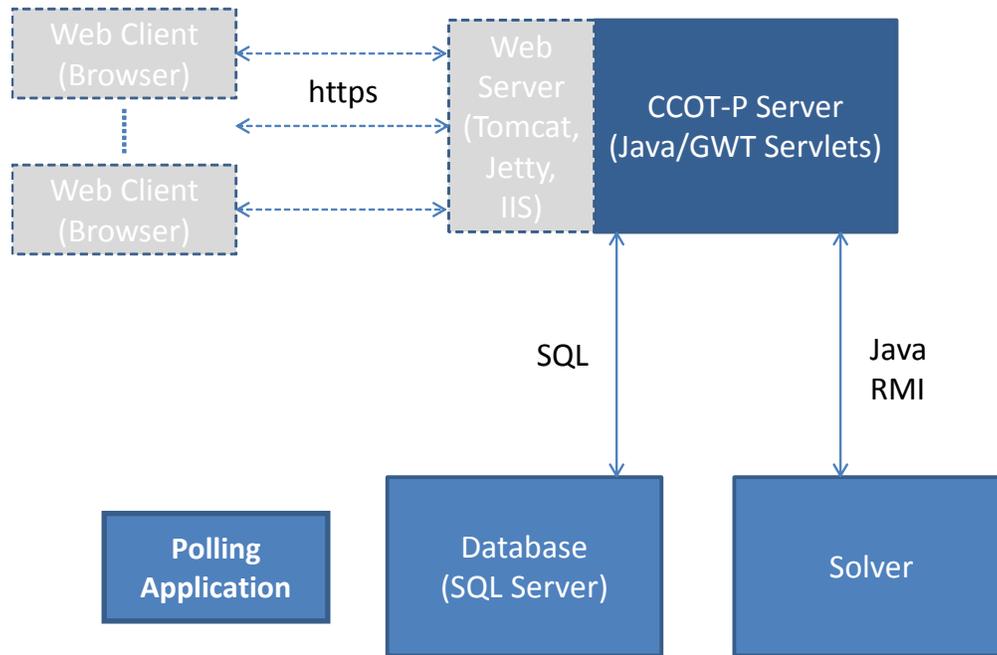


Figure 1: CCOT-P Architecture

The CCOT-P Server application, Solver, and Polling Application were developed for installation on Windows computers. The CCOT-P Server is a Java Google Web Toolkit (GWT) servlet requiring a Java Servlet Container web server to run. This server must pass the user’s authentication (login name) to the servlet. As of this writing, the OUSD(AT&L) eBusiness Center’s installation uses Apache Tomcat v7. The Solver is an executable .jar file. SQL Server 2008 or later is required for the Database. The Polling Application periodically polls the CCOT-P Server, the Solver, and the Database to provide a simple web page showing their states.

3. SOFTWARE DESIGN

3.1. CCOT-P Server

The CCOT-P Server is built using the Google Web Toolkit (GWT) [1]. GWT is a development toolkit for building web browser-based applications in Java. In all GWT web applications, there is a client side and a servlet side. The servlet side comprises one or more GWT Servlets, which are extensions of Java Servlets.

The client side is also written in Java, though using a subset provided by GWT. When the application is built, GWT translates the client-side Java into HTML/JavaScript. When a client connects with a browser, a default html file (CCOT.html) is sent to the client, which refers to the generated HTML/JavaScript. That script then runs in the context of the client browser and invokes servlet methods using Remote Procedure Calls (RPC).

For explanations of login roles and application flow, consult the “Contingency Contractor Optimization Phase 3, User Manual [3],” and “Contingency Contractor Optimization Phase 3, Quick Start Guide [4].”

3.1.1. *Third-Party Libraries*

In addition to the code developed by Sandia National Laboratories, third-party packages are used to extend the web application’s capabilities. These are detailed in *CCOT-P Third Party Software List [6]*.

3.1.2. *Configuration Files*

CCOT-P utilizes several configuration files to support multiple logical and physical arrangements:

OCS.gwt.xml: (gov.sandia.orca.ocs.web) This file is used by GWT to load GWT and GXT packages, define certain GWT and GXT parameters, and designate the entry point class for the web application. It should normally not need modification.

web.xml: (war/WEB-INF/) This file is used by the web server (Tomcat, Jetty, etc.) to define all servlets to be started, as well as the startup web page. Currently, it lists the OCS servlet (OCSServiceImpl), the remote logging service (to receive log messages from the client), and upload/download servlets for TPFDDs and Annex W. It should normally not need modification.

CCOT.css: (war/) This file defines the styles for widgets (text, buttons, etc.) in the web application. It should normally not need modification.

hibernate.cfg.xml: This file provides the necessary parameters for Hibernate: database dialect (SQL Server, MySQL, etc.), URL, username, password, and links to all of the database class mapping files. (Every database class has a mapping file, so they are not listed here.)

log4j.properties: This file defines logging parameters (log levels, file names, etc.) for Log4J.

solver.properties: This file defines the address and port on which the solver RMI (Remote Method Invocation) server sits.

uncertainty.properties: This file defines parameters for solving uncertainty models. This file is loaded with baseline values which can be modified based on analysis needs. This file should normally not need modification.

display.properties: This file is used to configure the classification banner and initial pop-up text. For DoD systems, the pop-up text would contain the standard Notice of Consent.

3.1.3. Database Access

The database design is detailed in *CCOT-P Database Design Document [5]*. The Hibernate API is used for all database interaction. An instance of a Java class corresponds to an entry in a database table. Each Hibernate class has an ID. If a table has a simple primary key, then the `getID()/setID()` methods for the class use a primitive. An example of this is `gov.sandia.orca.ocs.web.server.domain.dto.admin.Policy`. If a table uses a compound primary key, then the ID for that table is also a class. An example is `gov.sandia.orca.ocs.web.server.domain.dto.admin.AnnualCost`, which uses an `AnnualCostID` class as its identifier. For each Hibernate database class, a mapping file provides the mapping between the class and the database table.

It is the servlet – not the client – which interacts directly with the database. Also, Hibernate classes which are connected to the database do not have GWT JavaScript counterparts and are not directly serializable. All data exchanged between the servlet and client must be serializable. Therefore, all of the Hibernate classes are in the server packages (`gov.sandia.orca.ocs.web.server.domain`). Data transfer object (DTO) classes (`gov.sandia.orca.ocs.web.server.domain.dto`) are the classes which correspond directly to database table/view entries. A base data access object (DAO) class, `BaseHibernateDAO`, is used in conjunction with the DTO classes to interact with the database, providing methods for saving, updating, and deleting entries.

Finally, since the content of some database objects needs to be passed between clients and the servlet, several interfaces are defined in the `gov.sandia.orca.ocs.web.client.domain` package. These are the interfaces which are implemented by the servlet DTO classes. However, because data from these classes are used in GXT charts, additional classes which extend the GXT `BaseModel` class are used in exchanging data between the client and servlet. For example, an instance of the `Hostility` class is read from the database. When the client requests this data, an instance of `HostilityModel` (extending `BaseModel`) is populated and returned to the client. If the user changes this data, the `HostilityModel` is sent as an argument to the servlet, and an instance of the `Hostility` class on the servlet is populated with data from the `HostilityModel`, then written back to the database.

3.1.4. Client

The CCOT-P client classes are under the package `gov.sandia.orca.ocs.web.client`. These are the classes which will be translated into JavaScript and run in the context of a client browser. The client contains all of the classes which involve viewing and interacting with data, additional classes needed for transferring Hibernate data to and from the client, and all classes which are used by both the client and the servlet. In GWT, client classes may be used by the servlet, but not the other way around.

The following classes and interfaces are in the main client package:

- OCS
 - Implements the GWT EntryPoint interface.
 - Provides starting point for client (`onModuleLoad()`).
 - Instantiates OCSService for invoking servlet methods.
- OCSService: Interface which defines the methods OCS servlet (`OCSServiceImpl`) must implement.
- OCSServiceAsync
 - Interface which provides asynchronous (non-blocking) versions of OCSService methods.
 - These are the methods called by the OCS client.
 - Each method is type “void” with an additional `AsyncCallback` argument to handle success and failure return conditions.
- ClientModelRunner: Used for invoking servlet calls to run analyses.
- CommonData: Class containing common data used throughout the client.

The remaining classes are contained in the following packages:

- domain
 - Classes used for database data exchanged with servlet.
 - Classed used for putting data in form usable by GXT charts.
- exception: Custom Java Exceptions used by both client and server.
- utils: Utility classes used throughout the client.
- view: Classes for viewing and interacting with data.
 - admin: Classes specific to Administrator role.
 - superuser: Classes specific to Planning Manager role.
 - user: Classes specific to Analyst role.
 - scenarios: Classes for panels involving scenario data.
 - layout: Table layout classes for all login roles.
 - charts: Base classes for several charts and their components.
 - general: Other view classes used by all login roles.

3.1.5. Servlet

While the client code is mainly concerned with the user interface, the servlet takes care of most of the lower-level work. It interacts with the database, handles baseline, scenario, and analysis creation, and runs analyses. These classes are located in `gov.sandia.orca.ocs.web.server`.

The main servlet package contains the following classes:

- OCSServiceImpl

- Main servlet class.
- Extends GWT's RemoteServiceServlet and implements the client's OCSService interface.
- Web server creates a single instance of this class used for all clients.
- Clients invoke these methods using RPC.
- AnnexWDownloadServlet: Provides a Java HttpServlet used by the client to download an Annex W spreadsheet after it is produced.
- TPFDDImportServlet: Extends HttpServlet to allow the client to upload TPFDD spreadsheets
- BaselineCreator, ModelRunCreator, ScenarioAdder: utility classes to create new baselines, analysis runs, and scenarios.
- CleanupTool: Utility class which allows the Administrator login role to remove old model runs flagged for deletion from the database.
- HibernateSessionFactory: Utility class used for configuring and managing Hibernate database connections.
- OcsRemoteLoggingServlet: Extends GWT's RemoteServiceServlet to allow log entries to be received from the client and stored in the application log.

Four additional packages exist under the main servlet package:

- domain: Contains all database-related classes – DAOs, DTOs, and Hibernate mapping files.
- model: Contains utility classes for generating the necessary initial packages for an analysis.
- modelrun: Contains utility classes for actually invoking the solver and writing the results.
- util: Contains classes needed for uploading files, reading and writing Microsoft Excel sheets, and generating the Annex W sheet.

3.2. Solver

The CCOT-P Solver provides a simple Java Remote Method Invocation (RMI) server used to invoke the CPLEX solver to solve linear programs created by the CCOT web application.

Classes in CcotSolver are (under the gov.sandia.orca.ocs.solver package) as follows:

- IRemoteSolver
 - RMI interface definition.
 - The CCOT-P web servlet receives an instance of this interface when it connects.
 - Defines methods invoked by the CCOT-P web servlet to solve a matrix for a single package.
- CplexRemoteSolver
 - Implementation of IRemoteSolver interface.
 - Takes the SparseInputMatrix; invokes CPLEX using its Java API
- SparseInputMatrix
 - Class used for values of input matrix.
 - Contains only the nonzero values and their positions in the input matrix.
- SparseMatrixRow: Class used within SparseInputMatrix.
- ColumnSolution
 - Class used for solution.

- Contains column solution values, as well as the objective value and other pertinent attributes.
- SolverRmiService
 - Sets up RMI server and registers CplexRemoteSolver for use.
 - Provides additional methods used by Apache Commons Procrun to install as a Windows service.

3.2.1. Third-Party Libraries

In addition to the code developed by Sandia National Laboratories, third-party packages are used to extend the application's capabilities. These are detailed in *CCOT-P Third Party Software List [6]*.

3.3. Polling Application

The Polling Application is a Java project, and while not necessary for CCOT-P to function, provides a form of status monitoring for the system. It periodically polls the web application site, the database, and solver, and provides a basic web site to view the status. The Polling Application uses a single configuration file ("polling.properties"), which designates the following:

- Web page port.
- Polling interval (defaults to 2 minutes).
- CCOT-P Server URL.
- Database driver (defaults to SQL Server), URL, user name, and password.
- Solver host, port, and RMI registry entry.

The Polling Application requires a third-party library for polling the database. If using SQL Server, this will be Microsoft's sqljdbc4.jar library. If installed as a Windows service, the Polling Application uses the Apache Commons Procrun package.

The Polling Application has the following Java classes, under the gov.sandia.orca.ocs.polling package:

- Poller
 - Top level class with 'main' method.
 - Sets up FileHandler and ArrayLogHandler for logging.
 - Creates instances of DatabasePoller, WebAppPoller, and SolverPoller.
 - Instantiates WebPage.
 - Starts polling thread.
- PollerProcRunService: Provides extra methods for Procrun to facilitate installation as a Windows service, if desired.
- ArrayLogHandler
 - Extends logging Handler to store log entries in an array.
 - Allows for easy access to last N log entries (shown on web page).
- DatabasePoller: Uses JDBC to poll database.
- SolverPoller: Polls solver by connecting to RMI registry and checking for the solver's entry.
- WebAppPoller: Polls web app by using a Java HttpURLConnection (which includes https) and checking for a response.

- WebPage: Uses a simple Java `HttpHandler` to show the number of polling successes and failures, and the 20 most recent log entries.

While the web page shows overall statistics and the most recent log entries, the user may also look at the log files themselves, which are named with the form “polling.log.X,” where X are sequential integers starting with 0.

REFERENCES

- [1] J. D. Durfee, C. R. Frazier, A. Bandlow, J. L. Gearhart, K. A. Jones, "Contingency Contractor Optimization Tool Phase 3 Sustainment, Platform Requirements - Contingency Contractor Optimization Tool - Prototype," Sandia National Laboratories, May 2016.
- [2] A. Bandlow, J. D. Durfee, C. R. Frazier, K. A. Jones, J. L. Gearhart, "Contingency Contractor Optimization Tool Phase 3 Sustainment, Requirements Document - Contingency Contractor Optimization Tool - Prototype," Sandia National Laboratories, May 2016.
- [3] A. Bandlow, K. L. Adair, T. R. Brounstein, J. D. Durfee, J. L. Gearhart, K. A. Jones, N. Martin, L. K. Nozick, "Contingency Contractor Optimization Phase 3 Sustainment, User Manual - Contingency Contractor Optimization Tool, Engineering Prototype - Release 2.3," Sandia National Laboratories, July 2015.
- [4] A. Bandlow, K. L. Adair, J. D. Durfee, C.R. Frazier, J. L. Gearhart, L. K. Nozick, "Contingency Contractor Optimization Phase 3 Sustainment, Quick Start Guide - Contingency Contractor Optimization Tool, Engineering Prototype - Release 2.3," Sandia National Laboratories, July 2015.
- [5] C.R. Frazier, J. D. Durfee, A. Bandlow, J. L. Gearhart and K. A. Jones, "Contingency Contractor Optimization Phase 3 Sustainment, Database Design Document - Contingency Contractor Optimization Tool - Prototype," Sandia National Laboratories, May 2016.
- [6] J. D. Durfee, C. R. Frazier, A. Bandlow, "Contingency Contractor Optimization Phase 3 Sustainment, Third Party Software List - Contingency Contractor Optimization Tool - Prototype," Sandia National Laboratories, May 2016.

DISTRIBUTION

1 MS0899 Technical Library 9536

