RAND CORPORATION

Research Report

JAMES RYSEFF, DAVID ZHANG, PHILIP S. ANTON, WILLIAM SHELTON, SAMANTHA COHEN

Contractor Risk and Performance Indicators

Prototype Functional Specification and Developer Document

For more information on this publication, visit **www.rand.org/t/RRA542-2**.

Published by the RAND Corporation, Santa Monica, Calif.

Library of Congress Cataloging-in-Publication Data is available for this publication.

ISBN: 978-1-9774-0955-3

Cover: Senior Airman Beaux Hebert/U.S. Air Force; denisismagilov/Adobe Stock.

About This Report

Early indication of potential contractor performance risks and execution issues is critical for proactive acquisition management. When contractors are in danger of not meeting contractual performance goals, Department of the Air Force (DAF) acquisition management may not be fully aware of the shortfall until, for example, a schedule deadline is missed, government testing indicates poor performance, or costs exceed expectations. In response, we developed a new way to apply data science to a variety of government and external data sources to assess the relative contractor performance risks and early indicators of performance issues in DAF acquisition contracts and programs. This method seeks to produce risk and performance indicators earlier than current information sources and metrics do.

This report presents the technical design and architecture for a research software prototype that implements the methods described in two companion reports:

1. Philip S. Anton, William Shelton, James Ryseff, Samantha Cohen, Grant Johnson, Stephen B. Joplin, David Kravitz, Megan McKernan, and Alejandro Vigo Camargo, *Early Predictive Indicators of Contractor Performance: A Data-Analytic Approach*, RR-A542-1, 2022
2. Philip S. Anton, William Shelton, James Ryseff, Stephen B. Joplin, Megan McKernan, Chad J. R. Ohlandt, and Samantha Cohen, *Relative Contractor Risks: A Data-Analytic Approach to Early Identification*, RR-A433-1, 2022.

This report should be of interest to software developers who want to know about the details of the research prototype and its software architecture.

The research reported here was commissioned by the Deputy Assistant Secretary for Acquisition Integration, Office of the Assistant Secretary of the Air Force for Acquisition, Technology, and Logistics, and conducted within the Resource Management Program of RAND Project AIR FORCE as part of a fiscal year 2019 project on *Mining Program Execution Performance Problems Using Advanced Analytics* (PA19D-R9A1). It built on a project on the *Early Indicators of Relative Contractor Performance Risk for Air Force Acquisition* (PA18H-R8A4). Research for these two projects was conducted from February 2018 through September 2020.

RAND Project AIR FORCE

RAND Project AIR FORCE (PAF), a division of the RAND Corporation, is the DAF's federally funded research and development center for studies and analyses, supporting both the United States Air Force and the United States Space Force. PAF provides the DAF with independent analyses of policy alternatives affecting the development, employment, combat readiness, and support of current and future air, space, and cyber forces. Research is conducted

in four programs: Strategy and Doctrine; Force Modernization and Employment; Resource Management; and Workforce, Development, and Health. The research reported here was prepared under contract FA7014-16-D-1000.

Additional information about PAF is available on our website: www.rand.org/paf/

This report documents work originally shared with the DAF on October 7, 2020. The draft report, issued in December 2020**,** was reviewed by formal peer reviewers and DAF subject-matter experts.

Acknowledgments

Many thanks go to our DAF sponsors John Miller, Senior Executive Service; Mark Murphy, Senior Executive Service; and William D. Bailey, Senior Executive Service; and our action officers Brian Knight and Joseph A. Alfano for their support, encouragement, and guidance in this research.

We also acknowledge the valuable contributions of the other project members who contributed to the design and implementation of this methodology: Edward Balkovich, Jonathan Lee Brosmer, Lisa Colabella, Sherban Drulea, Jessica Duke, Suzanne Genc, Justin Grana, Elizabeth Hammes, Grant Johnson, Stephen B. Joplin, Amanda Kadlec, Nirabh Koirala, Gabriel Lesnick, Megan McKernan, Chad Ohlandt, Edward Parker, Cole Sutera, Zoltan Szalay, and Tim Webb.

Our thanks also to Obaid Younossi, Jim Powers, Patrick Mills, and Anu Narayanan at RAND for their support and encouragement. Our administrative assistants—Jordan Bresnahan, Silas Dustin, Maria Falvo, and Tandrea Parrott—kept us sane and functioning.

The authors claim any errors.

Summary

We developed a data analytic approach to integrate a variety of government and commercial data sources to assess the relative contractor performance risks for the Department of the Air Force (DAF) acquisition contracts and the programs they support. To test our approach, we developed a software prototype to implement those relative risk measures for which we could readily obtain data and for which the risk methodologies were amenable to available resources for the project.

In this report, we outline the functional software specification and basic design of that prototype, its architecture, and the core software components within it. We focus on the technical details of the prototype along with summary context on the basic function of the approach. Theoretical details are available in the main companion report (Anton, Shelton, Ryseff, Cohen, et al., 2022).

Contents

Figures

1. Overview

In this report, we describe the design and engineering decisions made when implementing a prototype version of the relative contractor risk application. The report is intended to assist in any future transition plan for this application from its prototype development at the RAND Corporation to a sustained ownership by an appropriate engineering team.

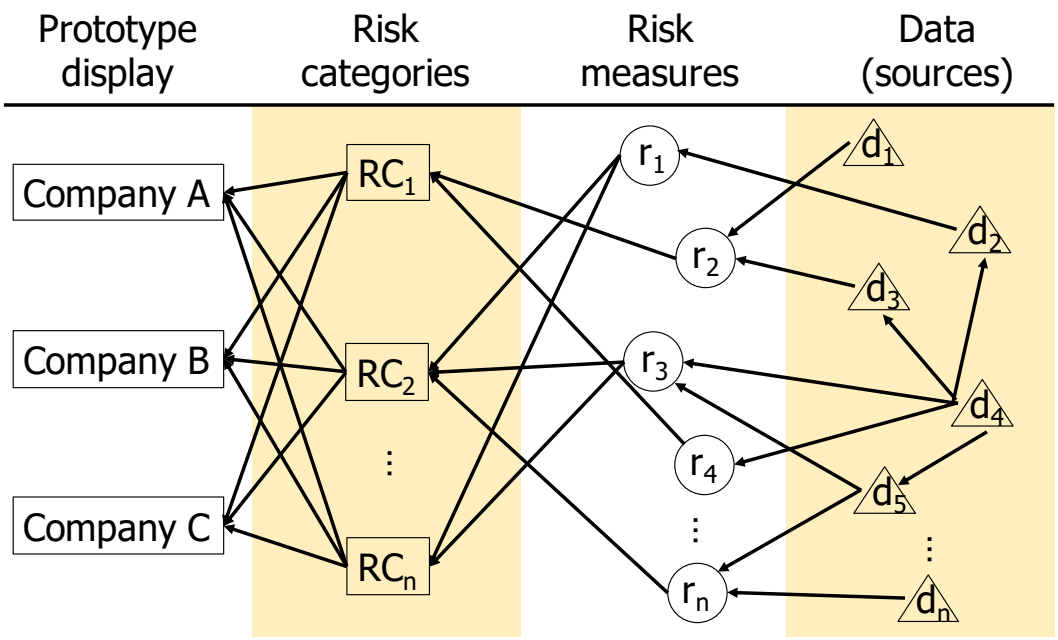
The application as initially implemented was a web application run on Microsoft's Internet Information Services (IIS) server. The web front end was implemented in C# using a model-view-controller pattern. For permanent storage, the application uses a MySQL database. The definitions for the database tables can be found in the python/templates/initialization.sql file, which originally creates the tables in a clean database. Finally, data were fed into the MySQL database by a pipeline of Extract, Transform, and Load (ETL) tasks. These tasks have been implemented in a mixture of C# and python tasks. Each task is atomic and communicates with other tasks only through the outputs it stores in the MySQL database. These tasks will be described in more detail later in this report.

2. The Prototype Architecture

Identifying Relative Risks Through Data Integration and Analysis

This prototype version of the relative contractor risk application builds on an earlier proof-of-concept effort to automatically ingest and analyze unstructured public financial Securities and Exchange Commission (SEC) filings and Department of the Air Force (DAF) Monthly Acquisition Reports to identify important facts, cross-linkages, and situations that could form the basis for subsequent risk analysis. Our approach is to build a taxonomy of contractor performance risks that could be measured by integrating internal DAF and other U.S. Department of Defense (DoD) data (including proprietary and other sensitive data), public government and commercial data, and licensed commercial data. We then developed and employed analytic techniques and algorithms to combine these structured and unstructured data to identify relative contractor performance risks.[1] The general concept is illustrated in Figure 2.1.

Figure 2.1. The Concept of Integrating Data to Measure Contractor Risks



SOURCE: Anton, Shelton, Ryseff, Joplin, et al., 2022.

[1] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

Intended Uses

The software prototype designed and implemented for this project was intended to serve two primary categories of users and their business needs. The first user category consists of DAF program managers or portfolio managers who oversee one or more DAF programs. The second user category consists of DAF analysts or other personnel assigned to the headquarters component who must keep track of the overall state of all DAF acquisition programs. It is important to note that the relative risks estimated in the prototype are not an absolute risk; just because one contractor or program is identified as having riskier characteristics than another does not mean that that contractor will definitely fail or encounter serious problems. Such an estimate means only that a contractor has a higher probability of doing so for the risk conditions from the relevant and available data evaluated by the application.[2]

Given these two categories of users, the relative contractor risk application has been primarily designed around two common uses for these personnel. In the first, an analyst wishes to quickly determine the overall state of DAF contractors and discover whether there are any new problems for closer examination. After logging in, this analyst can navigate to the All Contractors page. This page lists contractors whose relative risk level has increased or decreased since the previous evaluation period. This allows the analyst to quickly discover contractors with new risks or points of concern that might require further investigation to understand. After this, the application lists all of the known contractors from the riskiest to the least risky, allowing an analyst to quickly determine which DAF contractors present the greatest relative risk at that time. The analyst can click on any of the listed contractors to see more information about the nature of the risks associated with that contractor. The analyst can also search for a particular contractor of interest by typing a partial match for its name into a search box.[3]

In the second usage, an acquisition professional with responsibility for DAF programs wishes to understand the state of the overall acquisition portfolio. Two types of pages facilitate this usage. The Single Program page lists all of the information—compiled across various data sources—that this software prototype application contains about an individual program. In particular, this includes available information on any contractors that work on the program, any subcontractors that are known to work on the program, the relative risk of those contractors, and any recent Defense Contract Management Agency Performance Assessment Report (PAR) evaluations for the program. The page also lists recent news stories about the program that are relevant to acquisition personnel. If one of the contractors on a program has an elevated level of risk, the user can see details about the identified risks for that contractor by clicking on its name and reaching the associated Single Contractor page. This page lists the overall risk for the

[2] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

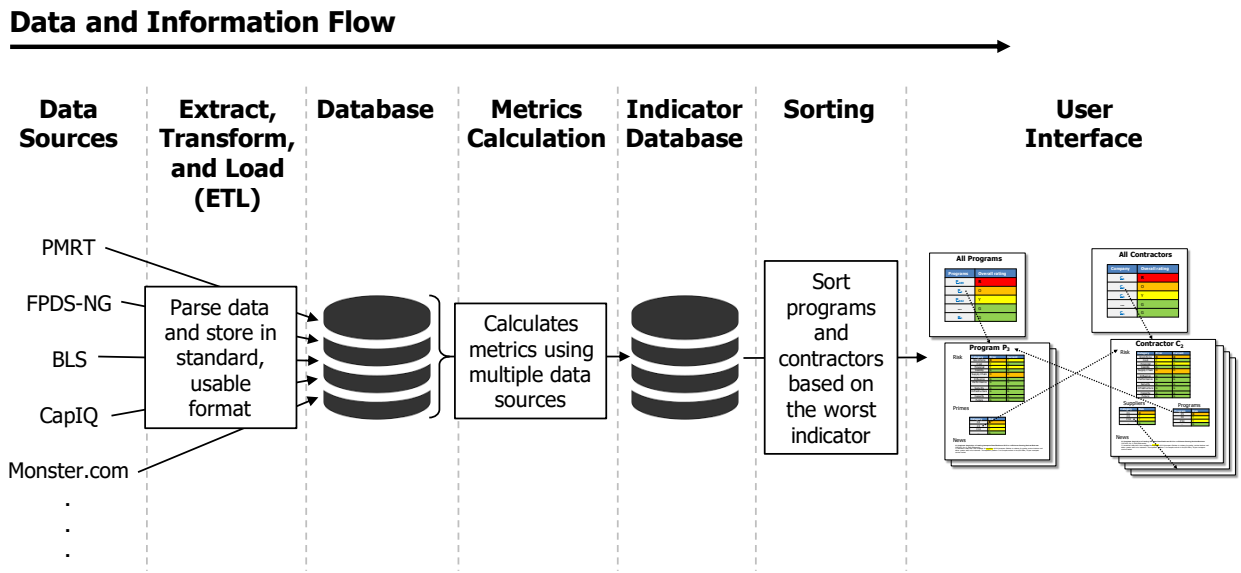
[3] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

contractor in each category of risk and explain the specific risk conditions that have been detected for this contractor. It also lists all known subcontractors for this contractor (regardless of which program those subcontractors work on) and any recent news stories about this contractor that are relevant to acquisition personnel. The Single Program page and the Single Contractor page are both reachable from the search box in the application or by clicking on any reference to the contractor or program from the All Contractors page or the All Programs page.[4]

Architecture Overview

To support these intended uses, the prototype consumes data from a variety of sources, including restricted government sources (e.g., the Project Management Resource Tools [PMRT]), open government sources (e.g., the Federal Procurement Data System–Next Generation [FPDS-NG], the Bureau of Labor Statistics [BLS] website, and USAspending.gov), and open or for-fee commercial sources (e.g., job posting websites and aggregators of financial data). Each of these sources feeds a data and information pipeline in the prototype architecture (see Figure 2.2). The pipeline has two primary components: data parsers and risk calculators.[5]

Figure 2.2. The Prototype's Architecture



NOTE: CapIQ = Capital IQ

[4] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

[5] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

ETL. As a first step in pipelines, parsers specific to each data source clean and normalize the data into standard formats. This serves several purposes. To begin with, parsing removes data that may be missing critical fields or that fail other validity checks. Parsing also matches data that originate from different sources to facilitate shared data constructs. For example, correctly identifying those data from different sources to describe the same company or DAF program can be a challenge when (as is often the case) the input data lack a unique identifier to confirm the match. Here, parsers look for near-matches between the input data and known companies or programs to determine whether the input data represent something new or can be matched to a known entity. Finally, parsers allow the prototype to add, remove, or change data sources that provide similar types of information without the need to alter or rewrite any part of the prototype software application other than the data parser specific to the data source in question.[6]

Metrics calculation. Once the data have been parsed and cleaned, a different set of software code examines the data to assess all indicators that can be calculated using the available data in the pipeline's second step. These indicators use data from one or more data sources, depending on the specific nature of the metric. Data pipelines are not run continuously; they run only when new data are received. This varies based on the nature of the data. For example, financial data are typically updated quarterly (when SEC requirements require corporations to update their investors and potential investors as to the state of their business). Other data, such as the posting of open job positions, could be updated on a monthly or daily basis if so desired. Once complete, the indications generated by these pipelines are stored in a MySQL database that other portions of the application can reference at any time.[7]

Sorting. Once all of the indicator modules have completed and any newly identified potential concerns have been stored, a final software job calculates the performance indicators (described earlier in this report); an overall indicator rating for each contracting company; the relative risks of all known DAF contractors within each of the 11 risk categories (as described in Anton et al., 2022); and an overall indicator rating for each contracting company.[8]

User interface. Each of these data pipelines and risk calculators operates independently from the final element of the prototype within the user interface. Users can interact with the prototype through a standard web browser with a common access card (CAC) reader. The research prototype's user interface has been built using the model-view-controller paradigm[9] and can be deployed on a standard Microsoft IIS web server.[10]

---

[6] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

[7] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

[8] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

[9] For more information, see, for example, Reenskaug, 1979a, 1979b, and 2003; and Burbeck, 1987.

[10] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

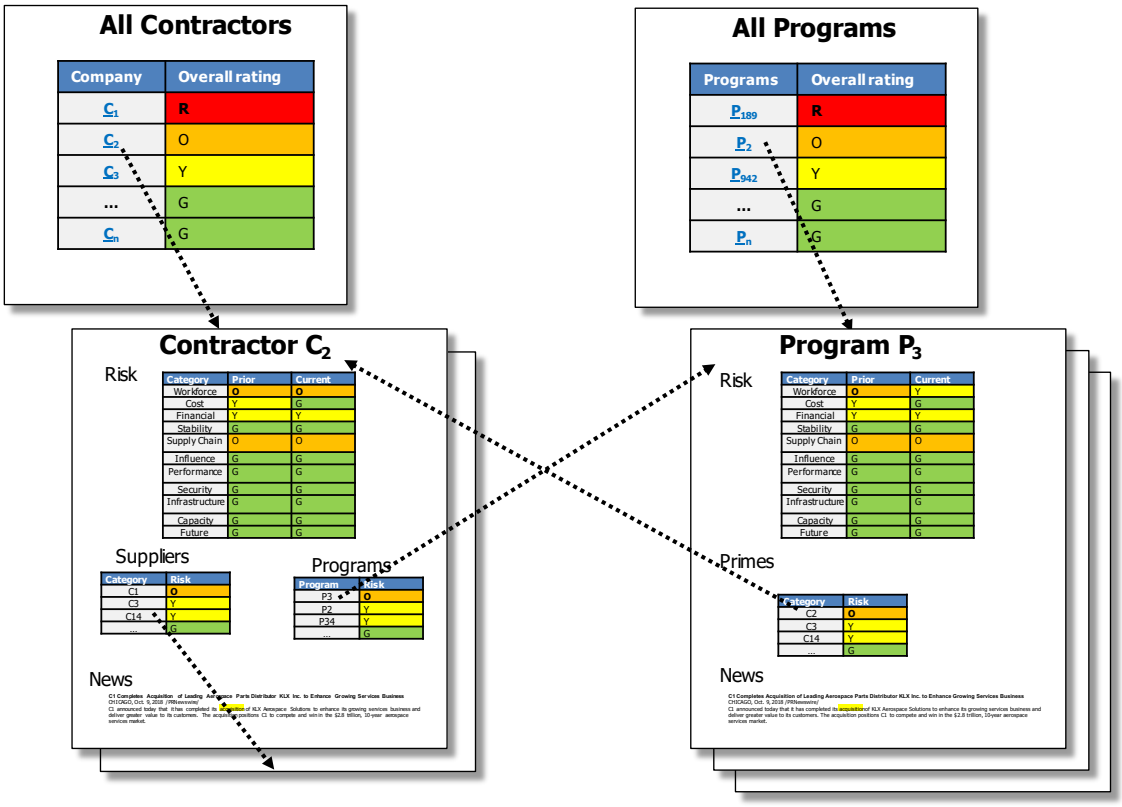
User Interface

Users of the relative contractor risk application interact with the data and results through a CAC-enabled web interface. As discussed earlier in this report, it provides four primary views:

1. an All Contractors page
2. numerous Single Contractor pages
3. an All Programs page
4. numerous Single Program pages.

Information on each of these pages is often linked: For example, the Single Contractor page displays information about (1) the specific contractor of focus for the page, (2) other contractors with which this contractor has some kind of relationship, and (3) DAF programs the contractor works on.[11] Users can easily navigate among these pages, as shown in Figure 2.3.

Figure 2.3. User Interface Structure for the Prototype: Contractor and Program Views



SOURCE: Anton et al., 2022.
NOTES: The arrows illustrate links in the prototype between views. Thus, the Program P_3 element on the Contractor C_2 page points to the Program P_3 page. The arrow in the lower left from Supplier C_{14} points to the Contractor C_{14} page lower in the stack. G = green; R = red; O = orange; Y = yellow.

[11] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

Additionally, every page in the web application has some global controls for navigating the prototype. Each page has a link to the All Contractors page and the All Programs page for the user to quickly return to an overview of all contractors or programs from a more-specific search. Also, every page has a search box for a user to navigate to the webpage for a specific contractor or specific program. The search box displays suggestions for companies and programs as the user types based on the text provided. Once a user sees a topic of interest, the user should select the appropriate option in the list and wait for the program to navigate to the webpage.[12]

Further discussion of each of these interface pages can be found, along with screenshots from the prototype, in Chapter 3 of the companion report (Anton, Shelton, Ryseff, Cohen, et al., 2022).

[12] Material in this section draws from Anton, Shelton, Ryseff, Joplin, et al., 2022.

3. Data Ingestion and Processing

Although the user interface of the relative contractor risk application is the primary means by which DAF personnel interact with the application, the user interface is dependent on a suite of other software programs that gather and curate[13] the information on which it depends. Most tasks have been designed to complete a single purpose. This keeps individual tasks simple and allows software engineers to quickly identify where they may need to make future changes. It also allows for a future transition of this prototype application to a cloud computing environment, where these tasks could be run in parallel.

Specifics of Each Software Task

The following sections describe the main program tasks in the system.

DUNS_Parser

One way that a company can be uniquely identified is through its Data Universal Numbering System (DUNS) number. Matching through a DUNS number is more reliable than matching on the company name, even though some data sources ingested by this application do not have a DUNS number. The parser reads a file downloaded from DUNS and establishes a relationship between entries in the companies table and the company_duns table (because a company can have more than one DUNS number). The parser verifies that each line of the file contains information about a valid company; many entries in the file have "Company Unknown" in the company name column and are skipped.

FPDS_ETL

This ETL task parses data originating from the Federal Procurement Data System (FPDS). It reads data about companies and the specific contracts they have been awarded by the DAF. First, it looks up each company by its DUNS number. If the DUNS number is unknown, it then attempts to look up the company by its name. If neither match, it adds the company to the companies table in the database. It also adds the DUNS number if that was previously unknown. Finally, it records useful information about the contracts that this company has been awarded, such as the place of performance, product and service codes (PSCs), and federal obligations for the contract.

[13] *Data curation* is the process of transforming data when necessary to integrate these date with all of the other sources used.

PMRT_Parser

The PMRT parser is the primary way that relationships are established between companies and the programs they use, because each row contains both the name of the prime contractor and the program name it uses. This parser assumes that the FPDS-NG parser has already run. It attempts to match a company to a known entity created by FPDS-NG using the company name. If this is unsuccessful, it creates a new company based on the name in PMRT. Associations between companies and programs are made based on the contract number.

USA_Spend_Parser

USAspending.gov is one of two places from which the application consumes information about subcontractors (the other being CapIQ). The application reads all files from a specified directory where files downloaded from USAspending.gov contain information identifying which subcontractors are working for which prime contractor on which specific DAF contracts. For each prime contractor and subcontractor in the file, the parser first attempts to match the company based on its DUNS number. If that fails, it attempts to match based on the company name. It matches the contractor or subcontractor to a program based on the contract number in the data.

CapIQ_ETL

This task reads a variety of useful facts about DAF contractors and stores them for analysis. Unlike the previous data sources, CapIQ does not include unique identifiers, such as a DUNS number for its companies; instead, matches must be made based on the company name. If the company is publicly traded, CapIQ does provide a new unique identifier in the stock symbol for the exchange where the company is listed. It also provides information about subcontractor-supplier relationships between companies, financial information about the company's performance, and the overall health of the company.

BLS_ETL

This task reads unemployment information collected by the BLS for each county in the United States and stores it to be used in risk calculations later in the task pipeline. It does not match these data to anything in the parser step, because the information is specific to locations as opposed to companies or programs.

PAR_Parser

This task parses PAR files in Portable Document Format (pdf) to extract the evaluations of each program over time. It matches to the program based on the program name.

News_fetcher

This task periodically updates the news stories associated with the contractors and programs stored by the application. It depends on a subscription to the CapIQ daily news feed. This feed periodically emails subscribers with news stories relevant to an individual company or companies that they have indicated an interest in. Although CapIQ is the source that is currently available to this project, there are a variety of alternative news feeds that could easily be substituted. The primary assumption made by this task is that it does not have to determine which company a news story covers because it relies on preprocessing by some other task or source for that functionality.

CapIQ emails these stories to a specified email account. This task downloads the news stories from that email account and filters them for relevance using a list of keywords indicating whether the story might be related to acquisition activity and contractor stability. Finally, it stores the relevant news stories in the database and associates them with the company.

Evm_LeadingIndicatorsRisks

This task uses data from the Earned-Value Management–Central Repository (EVM-CR) to look for risks specific to a particular program and contract vehicle. The specific risks detected are based on formulas published by the National Defense Industrial Association. The task first loads data downloaded from EVM-CR by iterating through the directory where the appropriate files have been saved and loading each file's data individually. New files can be added to this directory over time and incorporated into the final results. After this, nine separate risk conditions are calculated for each contract in the data. The risk types span a range of categories, including schedule risks, cost risks, performance risks, and technical risks. Data from EVM-CR are associated with other data about U.S. Air Force programs through either the contract identifier or the program name.

Financial Ratio Risks

This task looks for U.S. Air Force contractors in poor financial health. It relies on a feed of data from CapIQ that details a variety of financial metrics, including the current ratio, the earnings before interest, taxes, depreciation, and amortization (EBITDA), the net debt held by the company, the EBITDA interest expenses, and the total debt equity for the company. Given this information, it determines the risk level of each company using these parameters and saves risks to the database accordingly.

SAM_Exclusions_Parser

This task reads information about companies listed in the System for Award Management (SAM) that have been excluded from being awarded federal contracts. It then attempts to look up the company by its DUNS number and creates a risk rating if it identifies any such company.

InfluenceRisk

This task uses the data from FPDS-NG to calculate the total revenue that each prime contractor derived from the DAF in fiscal year 2018. It then compares that amount with the contractor's annual revenue in that year (taken from CapIQ). Prime contractors who derive less than 10 percent of their overall revenue from the government are marked as having a higher risk.

MonsterJobs_Parser

This task reads data received from Monster.com about job openings around the United States. It first calculates the national average and standard deviation for how long each type of job has been open. It then looks at each job opening to determine which company has posted the job. If the company is a defense contractor and one or more jobs have been open for an atypically long period, it stores a risk describing the location and details of one of the most–difficult-to-fill positions and how many job openings in total this contractor is having difficulty filling.

MergerRiskCalculator

This task loads a file acquired from CapIQ with all known corporate mergers that have occurred in the past two years. It then matches these companies to known publicly traded companies based on their stock symbols. If the company in question is a defense contractor, it saves a "Stability" risk because of its mergers and acquisitions activity.

Risks.py/Calcs.py

This task detects several risk conditions from data ingested by tasks earlier in the pipeline. The first is the Revenue Decrease risk, which is based on data ingested from CapIQ regarding the overall revenue taken in by the company. Next, there is the credit health risk, also based on CapIQ data about the overall debt load for the company and other factors. Third, there is the unemployment risk, which is derived from the BLS data on unemployment rates by county across the country. Finally, the new contractor risk and the prior experience risk are both based on PSCs and company obligations ingested from FPDS-NG. The specifics of the risk calculations can be found in Chapter 3 of the companion report (Anton, Shelton, Ryseff, Cohen, et al., 2022).

Rank_Contractors

This task runs last in the overall task pipeline. As its first step, it deletes all rows from the previous_contractor_risk_level and previous_contractor_risk_category_level tables. It then transfers all risks in the current_contractor_risk_level and current_contractor_risk_category_level tables into the previous tables, storing them for comparison with the values it is about to calculate. It then aggregates risks, first within categories for each company and then for each company overall. These overall risks are stored in the current risk tables for use in the user interface component. All risk levels are also permanently saved in the historical_contractor_risk_level and historical_contractor_risk_category_level tables.

4. Future Steps

This report describes the overall design of the prototype relative contractor risk application and an implementation of that design. The current implementation has been deployed in RAND's information technology infrastructure. Consequently, it uses the MySQL database for storage and a mix of python and C# software programs to instantiate the individual tasks in the data ingestion pipeline. However, these implementation details could easily be altered in a production environment. This chapter identifies recommendations for future steps for transitioning and updating the prototype tool for use in this environment.

The primary guideline for extending this prototype design is to ensure that individual tasks within the data ingestion pipeline should be able to run without interfering with any other tasks in the pipeline. Tasks communicate solely through their intermediate outputs; a task stores data in a database table as it completes its computation and allow tasks farther down the pipeline to operate on its outputs. This isolation allows individual components of the pipeline to improve or alter data without affecting any other tasks within the pipeline, isolate any potential problems, and ensure that individual tasks remain simple and straightforward.

The framework developed here could be extended with minimal effort in two ways. First, as additional risk criteria are identified, new risks can be added without changing the existing tasks or risk criteria. These risks roll up into the overall pipeline and risk calculations just as all of the existing risks already do. Second, as additional relevant data sources are identified or purchased, these data sources can be used as inputs into the appropriate risk calculations. This might not even require changes to the code that determines whether a risk condition exists, as long as the new data sources are correctly transformed into the appropriate intermediate inputs for the appropriate risk calculation job. However, future developers should carefully review the full set of tasks that depend on these intermediate inputs to determine whether the risk calculation needs to change. Some calculations might require an update to work correctly with additional input data.

Additionally, this application has been designed with an eye toward future improvements in interoperability between DoD software applications. In particular, the relative contractor risk application would benefit from a DoD-wide software architecture based on service-oriented architecture principles, also referred to as *microservices*. For example, the relative contractor risk application needs a way to uniquely identify all the contractors that are currently working on or previously worked on DoD acquisition projects. Unfortunately, there is not a universal way to refer to a unique company; instead, different data sources use their own keys to refer to the same set of companies. Similarly, the relative contractor risk application tracks the relationship between prime contractors and their network of subcontractors and suppliers. Neither of these problems are unique to the relative contractor risk application. Many parts of DoD would benefit

from having a single service that serves as the authoritative source for these facts and supports an interactive application programming interface that other applications could rely on to provide timely responses to these types of queries. If such a service is ever built, relatively minor updates to the relative contractor risk application would allow it to take advantage of these common components.

Finally, the data ingestion pipeline has been designed in a way that it could be migrated to a cloud computing environment and be run as a series of scheduled tasks with dependencies between those tasks. Amazon Web Services Batch or Azure Logic Apps would easily support the ecosystem of tasks required to periodically update the risks and the data on which those risk conditions are based. A cloud environment would also allow for a simple instantiation of the contractor's database of choice to provide permanent storage; nothing about the current implementation depends on atypical features of the MySQL database, and the storage solution could be quickly migrated to many other platforms, such as PostgreSQL, SQL Server, Oracle Database, or cloud-specific storage solutions (e.g., Aurora or Azure Cosmos Database). Any transition of this application into a cloud environment should be coordinated with a larger transition of DoD applications into the cloud to ensure interoperability with other software applications.

Abbreviations

BLS	Bureau of Labor Statistics
CAC	common access card
CapIQ	Capital IQ
DAF	Department of the Air Force
DoD	U.S. Department of Defense
DUNS	Data Universal Numbering System
ETL	Extract, Transform, and Load
FPDS	Federal Procurement Data System
FPDS-NG	Federal Procurement Data System–Next Generation
IIS	Internet Information Services (Microsoft product)
PAR	Performance Assessment Report
PMRT	Project Management Resource Tools
PSC	product and service code
SEC	Securities and Exchange Commission

References

Anton, Philip S., William Shelton, James Ryseff, Samantha Cohen, Grant Johnson, Stephen B. Joplin, David Kravitz, Megan McKernan, and Alejandro Vigo Camargo, *Early Predictive Indicators of Contractor Performance: A Data-Analytic Approach*, Santa Monica, Calif.: RAND Corporation, RR-A542-1, 2022.

Anton, Philip S., William Shelton, James Ryseff, Stephen B. Joplin, Megan McKernan, Chad J. R. Ohlandt, and Samantha Cohen, *Relative Contractor Risks: A Data-Analytic Approach to Early Identification*, Santa Monica, Calif.: RAND Corporation, RR-A433-1, 2022.

Burbeck, Steve, "Applications Programming in Smalltalk-80: How to Use Model-View-Controller (MVC)," 1987. As of September 1, 2020: https://web.archive.org/web/20120729161926/http://st-www.cs.illinois.edu/users/smarch/st-docs/mvc.html

Reenskaug, Trygve, "THING-MODEL-VIEW-EDITOR: An Example from a Planningsystem," May 12, 1979a. As of September 1, 2020: http://heim.ifi.uio.no/~trygver/1979/mvc-1/1979-05-MVC.pdf

———, "MODELS – VIEWS – CONTROLLERS," December 10, 1979b. As of September 1, 2020: http://heim.ifi.uio.no/~trygver/1979/mvc-2/1979-12-MVC.pdf

———, "The Model-View-Controller (MVC): Its Past and Present," August 20, 2003. As of September 1, 2020: http://heim.ifi.uio.no/~trygver/2003/javazone-jaoo/MVC_pattern.pdf

PROJECT AIR FORCE

A recurring challenge to successful Department of the Air Force (DAF) acquisition program execution is poor contractor performance. Early indication of potential contractor performance risks and execution issues is critical for proactive acquisition management. When contractors are in danger of not meeting contractual performance goals, DAF acquisition management may not be fully aware of the shortfall until, for example, a schedule deadline is missed, government testing indicates poor performance, or costs exceed expectations. In response, the authors developed a new way to apply data science to a variety of government and external data sources to assess the relative contractor performance risks and early indicators of performance issues in DAF acquisition contracts and programs. This method seeks to produce risk and performance indicators earlier than do current information sources and metrics.

In this report, the authors outline the functional software specification and basic design of that prototype, its architecture, and the core software components within it. The authors focus on the technical details of the prototype, along with summary context on the basic function of the approach.

$15.00

ISBN-10 1-9774-0955-5
ISBN-13 978-1-9774-0955-3

www.rand.org

RR-A542-2