ORNL/SPR-2019/1351

Real-Time Automated Health Information Technology Hazard Detection

Teja Kuruganti
Olufemi A. Omitaomu
Ozgur Ozmen
Laura Pullum
Hilda B. Klasky
Mark Martin
Mohammed Olama
Rajasekar Karthik
Greg Watson
Robert Smith
Alex Moore
Lance Drane
Caleb Cooper
James Nutaro
Helia Zandi
Yunhe Feng
Qing Charles Cao
Jordan Pellett
Jay Jay Billings
Eileen McAllister
Jeremy Cohen

September 2019

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE US DEPARTMENT OF ENERGY

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, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents 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 or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ORNL/SPR-2019/1351

Computational Sciences and Engineering Division

REAL-TIME AUTOMATED HEALTH INFORMATION TECHNOLOGY HAZARD DETECTION

Teja Kuruganti
Olufemi A. Omitaomu
Ozgur Ozmen
Laura Pullum
Hilda B. Klasky
Mark Martin
Mohammed Olama
Rajasekar Karthik
Greg Watson
Robert Smith
Alex Moore
Lance Drane
Caleb Cooper
James Nutaro
Helia Zandi
Yunhe Feng
Qing Charles Cao
Jordan Pellett
Jay Jay Billings
Eileen McAllister
Jeremy Cohen

Date Published: September 2019

Prepared by
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, TN 37831-6283
managed by
UT-BATTELLE, LLC
for the
US DEPARTMENT OF ENERGY
under contract DE-AC05-00OR22725

DOCUMENT VERSION CONTROL

Version	Date Changed	Description of Changes	Owner

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AOSVR	accurate online support vector regression
API	application programming interface
BCPD	Bayesian change point detection
BOCPD	Bayesian online change point detection
CDW	Corporate Data Warehouse
CPD	change point detection
DFD	diagnosis flow diagram
DTMC	discrete-time Markov chain
EHR	electronic health record
ETL	extract, transform, and load
HD	hazard detector
HIT	health information technology
HITHD	Health Information Technology Hazard Detection (database)
IHD	Ischemic Heart Disease
LDA	latent Dirichlet analysis
MC	Markov chain
ORNL	Oak Ridge National Laboratory
OSEHRA	Open Source Electronic Health Records Alliance
PA	perturbation algorithm
PD	probability distribution
PDF	probability density function
PHI	Private Health Information
REST	Representational State Transfer
SME	subject matter expert
SPC	statistical process control
std	standard deviation
STM	system transition matrix
SVR	support vector regression
TP	transition probability
TPM	transmission process model
VA	US Department of Veterans Affairs
VHA	Veterans Health Administration
VistA	Veterans Information System and Technology Architecture
V&V	verification and validation

ACKNOWLEDGMENTS

The authors wish to thank the following people for their technical analysis, technical coordination, and/or feedback during all aspects of this project:

Jonathan R. Nebeker, MD, MS
Chief Informatics Patient Safety
Clinical Informatics and Data Management Office
US Department of Veterans Affairs
Washington, DC

Jeanie Scott, MS
Director, Informatics Patient Safety
Clinical Informatics and Data Management Office
US Department of Veterans Affairs
Washington, DC

Merry Ward, PhD
Research and Innovation Manager
Clinical Informatics and Data Management Office
US Department of Veterans Affairs
Washington, DC

Angela L. Laurio, DPH
Statistician
Informatics Patient Safety
Clinical Informatics and Data Management Office
US Department of Veterans Affairs
Washington, DC

Frank Drews, PhD
Cognitive Scientist
Clinical Informatics and Data Manaement Office
US Department of Veterans Affairs
Washington, DC

Makoto Jones, MD
VA Infectious Disease Physician
US Department of Veterans Affairs
Washington, DC

EXECUTIVE SUMMARY

The objective of this project is to develop an automated surveillance tool that will help the US Department of Veterans Affairs (VA) to detect potential hazards in health information technology (HIT) systems. To achieve this objective, Oak Ridge National Laboratory (ORNL) researchers have developed computational methods and a surveillance tool that will detect safety concerns and identify opportunities to improve HIT and safety within the VA system using Corporate Data Warehouse (CDW) data. This work is limited to the safe operation and safe use of HIT. It does not address the use of HIT to make health care safer. This project leveraged ORNL expertise in computational sciences and its computing resources accordingly.

This report documents the FY 2019 accomplishments on this project. There were three major tasks. The first task developed data extraction approaches, which helped provide a better overall understanding of the important data elements in CDW. The outputs of this task served as inputs into the second task, which is the hazard detection and characterization task. This second task involved development of three methods that can be categorized into probabilistic model, event-based hazards detection, and system-based reliability performance evaluation. The third task dealt with the design, development, and demonstration of the automated surveillance tool. The tool currently resides in ORNL's Private Health Information enclave at Oak Ridge, Tennessee, and operates on the VA's CDW data. The developed capabilities and tool have been tested on CDW data from different clinical domains including Consult, Radiology, Laboratory, and Medication.

1. INTRODUCTION AND BACKGROUND

The health and safety of veterans is one of the highest national priorities; and the US Department of Veterans Affairs (VA) is committed to providing seamless care for veterans, including access to a complete electronic health record (EHR) and shared, transparent care pathways. To ensure seamless care for Veterans, the VA will move toward seamless US Department of Defense–VA health information technology (HIT) systems. The Veteran's Health Administration (VHA) expects that during adaptation, adoption, and post-deployment, there will be a need for an automated, real-time prospective hazard detection approach to detect hazards and monitor the safety of HIT, the safe use of HIT, and the impact of HIT on safety. To meet these goals, the health care system will require systems and methods to monitor the implementation and maintenance process. The VA seeks to develop monitoring and hazard detection for new types of safety problems that have not been encountered before, and this will require a new approach for data and event streaming to be developed as well as an understanding of the requirements for a rapid response approach to issues identified with the monitoring and event streaming approach.

The goal of this project is to develop evidence-based models and frameworks that work towards automated, near real-time and real-time systems that will reveal safety concerns, predictable and unpredictable unintended consequences, and opportunities to improve HIT and safety. The objectives of this work are to leverage Oak Ridge National Laboratory (ORNL) expertise in computational sciences and its computing resources to improve the quality and safety of HIT systems by developing a proof-of-concept system to detect and manage hazards in near-real-time to promote the safe operation and use of HIT. This work is limited to the safe operation use of HIT. It does not address the use of HIT to make health care safer.

At the end of this project, a HIT safety manager will be able to use the proof-of-concept solution to monitor HIT systems for a variety of hazard signals, diagnose with computer decision support whether the hazard is real, and manage the remedy for the hazard. Depending on the timing of availability of data from the EHR system, the proof of concept should work for the Veterans Information System and Technology Architecture (VistA), the new EHR, or both.

Consequently, ORNL researchers developed an end-to-end system for monitoring and detecting hazards in HIT. The schematic model for the proposed end-to-end HIT hazard detection system is shown in Figure 1. The model consists of three components:

1. data extraction and preprocessing,
2. hazards monitoring and detection, and
3. report generation and feedback.

The hazards monitoring and detection component, in turn, comprises two layers: event-based hazards detection and system-based reliability performance evaluation.

The proposed system model has the following capabilities:

- Data-driven hazard characterization and detection methods to structurally query EHR data and detect anomalous patterns that might constitute hazards. These methods are implemented as a near real-time detection tool that is expected to run hourly/daily on an EHR database.
- Analytical methods that evaluate the system-level performance of HIT systems to detect emerging behavior in the system that might result in unfavorable outcomes in patient safety. This part is implemented as a surveillance tool that is expected to run weekly/monthly on an EHR database.

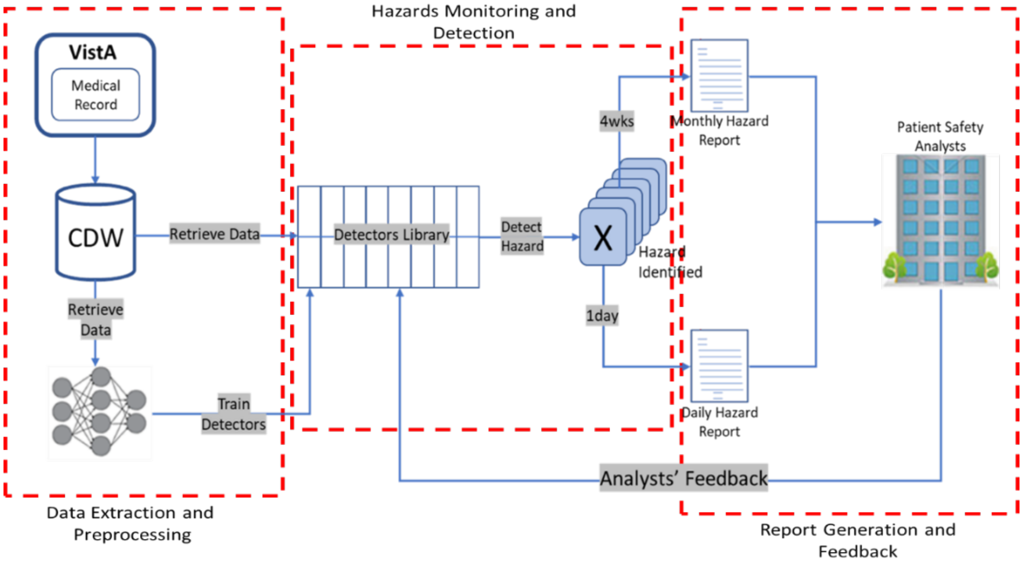


Figure 1. Components of an end-to-end HIT hazard detection system.

The EHR systems used to test the proposed framework are the VA's HIT system, named VistA,[1] and Corporate Data Warehouse (CDW)[2] systems. This report describes some of the initial solutions developed as part of this proposed end-to-end system. The data extraction and preprocessing capabilities are covered in Section 2; the hazards monitoring and detection capabilities are covered in Section 3; and the report generation and feedback capabilities are covered in Section 4.

2. DATA ACCESS, ANALYSIS, AND REQUIREMENT DEFINITION

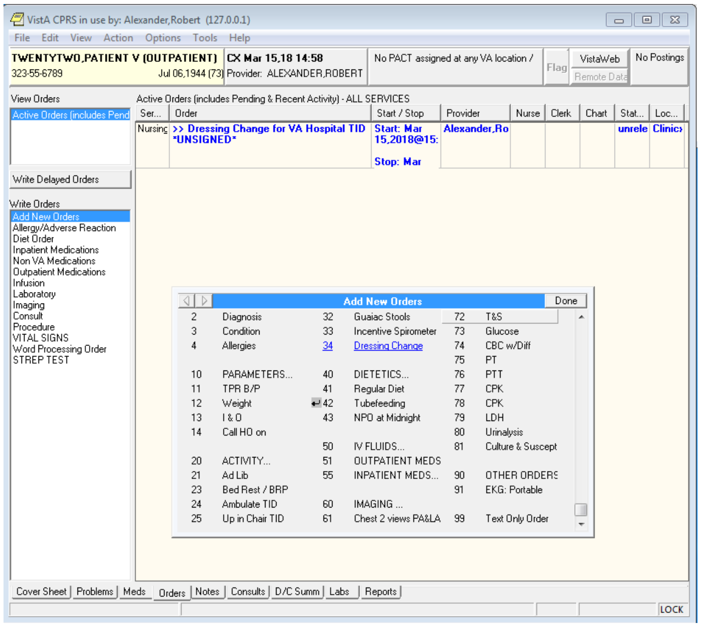
This task was initially focused on extracting relevant data fields that are involved in the software workflow (of the VistA system) to execute a diagnosis flow or decision-making framework. We selected the Ischemic Heart Disease (IHD) diagnosis flow diagram (DFD) that was well studied in another project as part of the ORNL and VHA collaboration. We analyzed VistA software packages and subsequently identified relevant data fields and data flows regarding the diagnosis flow diagrams. This analysis could be possible through three mappings: *(i) Diagnosis Flow Diagram and VistA system menus/screens, (ii) VistA screens and VA FileMan files,* and *(iii) VA FileMan files and CDW fields.* The goal of the task was to develop a data extraction approach and feed extracted information to the probabilistic model component and develop understanding of the important data elements in this process.

2.1 MAPPING DIAGNOSIS FLOW DIAGRAM AND VISTA SYSTEM

Our first approach to mapping focused on the generic transactions and VistA packages that are used to realize the IHD DFD rather than focusing on each individual clinical process/procedure or decision point (node) in IHD DFD. This approach allowed us to find a more generic way of representing disease agnostic VistA processes and patterns that are commonly shared by any DFD. We identified that IHD DFD includes four major clinical order processes: *(i) Consult, (ii) Radiology, (iii) Laboratory, and (iv) Medication.* Initially, we focused solely on clinical *Consult* orders to understand the order process in VistA and the transactions an order goes through. Iteratively, we extended this approach to *Laboratory*, *Radiology*, and *Medications* orders.

[1] https://www.data.va.gov/dataset/veterans-health-information-systems-and-technology-architecture-vista.
[2] https://www.data.va.gov/dataset/corporate-data-warehouse-cdw.

To gain insights into the clinical user interactions and the data flow through the VistA system, we first stood up a local VistA server. The Open Source Electronic Health Records Alliance (OSHERA) has released demo versions of CPRS, BCMA, and Vitals applications. OSEHRA also provides support for Docker containers that stand up a VistA server instance (see installation instructions[3]). The team installed a VistA server (specifically, VEHU) instance and the demo applications that work with the VEHU (version 1.30.75). Figure 2 is the general layout of the CPRS client, and Figure 3 shows the terminal of the VistA server. Please note that BCMA and Vitals clients are not supported by OSEHRA and could not be connected to the VistA server.



TWENTYTWO,PATIENT V (OUTPATIENT)
323-55-6789 Jul 06,1944 (73)

CX Mar 15,18 14:58
Provider: ALEXANDER,ROBERT

No PACT assigned at any VA location /

No Postings

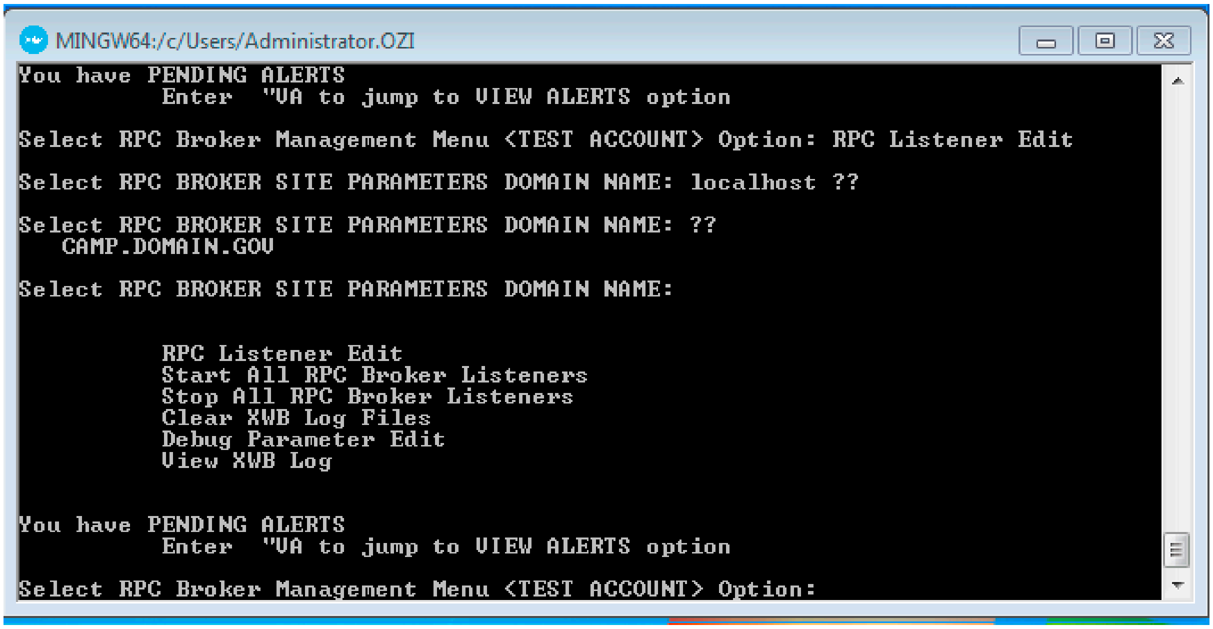
Active Orders (includes Pending & Recent Activity) - ALL SERVICES

Ser...	Order	Start / Stop	Provider	Nurse	Clerk	Chart	Stat...	Loc...
Nursing	>> Dressing Change for VA Hospital TID *UNSIGNED*	Start: Mar 15,2018@15: Stop: Mar	Alexander,Ro				unrele	Clinic:

Add New Orders

2	Diagnosis	32	Guaiac Stools	72	T&S
3	Condition	33	Incentive Spirometer	73	Glucose
4	Allergies	34	Dressing Change	74	CBC w/Diff
				75	PT
10	PARAMETERS...	40	DIETETICS...	76	PTT
11	TPR B/P	41	Regular Diet	77	CPK
12	Weight	42	Tubefeeding	78	CPK
13	I & O	43	NPO at Midnight	79	LDH
14	Call HO on			80	Urinalysis
		50	IV FLUIDS...	81	Culture & Suscept
20	ACTIVITY...	51	OUTPATIENT MEDS		
21	Ad Lib	55	INPATIENT MEDS...	90	OTHER ORDERS
23	Bed Rest / BRP			91	EKG: Portable
24	Ambulate TID	60	IMAGING ...		
25	Up in Chair TID	61	Chest 2 views PA&LA	99	Text Only Order

Figure 2. VistA CPRS client—Orders tab.



```
You have PENDING ALERTS
          Enter  "VA to jump to VIEW ALERTS option

Select RPC Broker Management Menu <TEST ACCOUNT> Option: RPC Listener Edit

Select RPC BROKER SITE PARAMETERS DOMAIN NAME: localhost ??

Select RPC BROKER SITE PARAMETERS DOMAIN NAME: ??
   CAMP.DOMAIN.GOV

Select RPC BROKER SITE PARAMETERS DOMAIN NAME:


          RPC Listener Edit
          Start All RPC Broker Listeners
          Stop All RPC Broker Listeners
          Clear XWB Log Files
          Debug Parameter Edit
          View XWB Log


You have PENDING ALERTS
          Enter  "VA to jump to VIEW ALERTS option

Select RPC Broker Management Menu <TEST ACCOUNT> Option:
```

Figure 3. VistA server on the virtual machine environment (CHUI screens).

[3] https://hub.docker.com/r/osehra/vehu/.

2.2 MAPPING *FILEMAN* FILES TO VISTA SCREENS

The team evaluated multiple options to collect data from the VistA server instance and understand how the data is manipulated on a *FileMan* database and what screens/forms users interact with. The first option was to interrupt Remote Procedure Call (RPC) Broker calls that happen between the CPRS Demo application and the VistA system, and consequently to the *FileMan* database. This option would provide command calls between two systems. Considering the difficulties in interpretation of the calls (MUMPS and DELPHI commands) and in crowd-sourcing user data for VistA use cases, the team decided to manually enter orders and follow them through a generic transactional process model that is based on expert opinion. A generic transactional process model of a clinical object (i.e., order) is determined as in Figure 4.

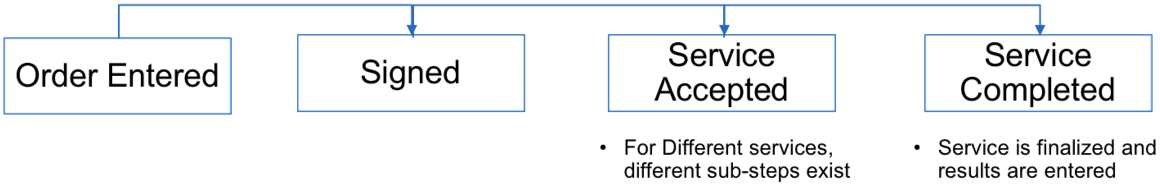

Order Entered
Signed
Service Accepted
Service Completed
• For Different services, different sub-steps exist
• Service is finalized and results are entered

Figure 4. Generic transactional process model.

This generic model can be leveraged to represent each order type with its specific subprocesses and substates. For example, a Consult has a *Receive* process that is followed by a *Schedule* process in the "Service Accepted" state in Figure 5. Once an order is accepted, its status becomes *active,* and once it is scheduled its status changes to *scheduled*. This status information varies for different order types. For instance, a laboratory order becomes *accepted* instead of being *active*. To observe an order from entry to completion, we used CPRS Demo and CHUI screens and monitored changes in the data fields on VistA *FileMan* files using the *Inquire* feature of the VistA system. Consequently, we identified the *FileMan* files that are revised by each process in the transactional process model and listed the field information that is updated. This gives us high-level information flow between files for each transactional process (Figure 5).

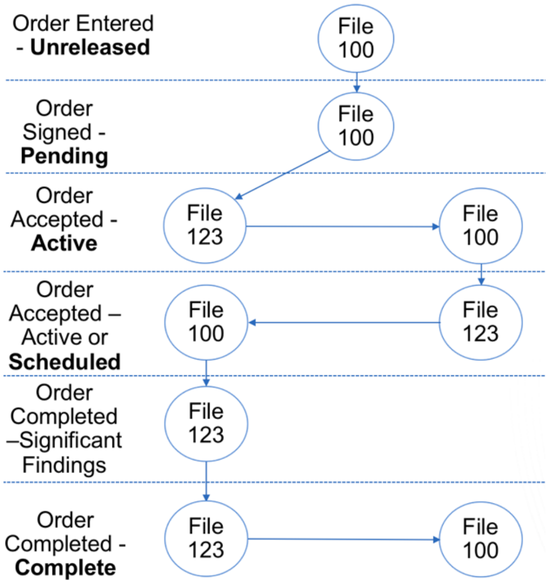
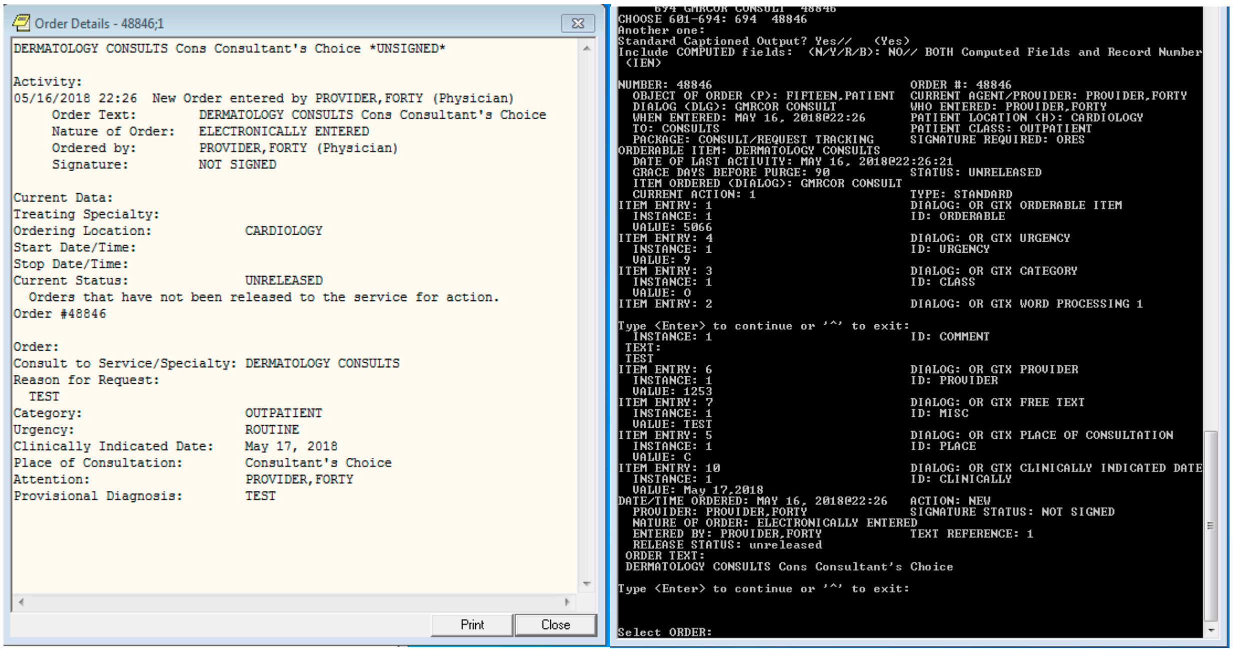

Order Entered - **Unreleased**
File 100
Order Signed - **Pending**
File 100
Order Accepted - **Active**
File 123
File 100
Order Accepted – Active or **Scheduled**
File 100
File 123
Order Completed –Significant Findings
File 123
Order Completed - **Complete**
File 123
File 100

Figure 5. Information flow between *FileMan* files for a Consult order.

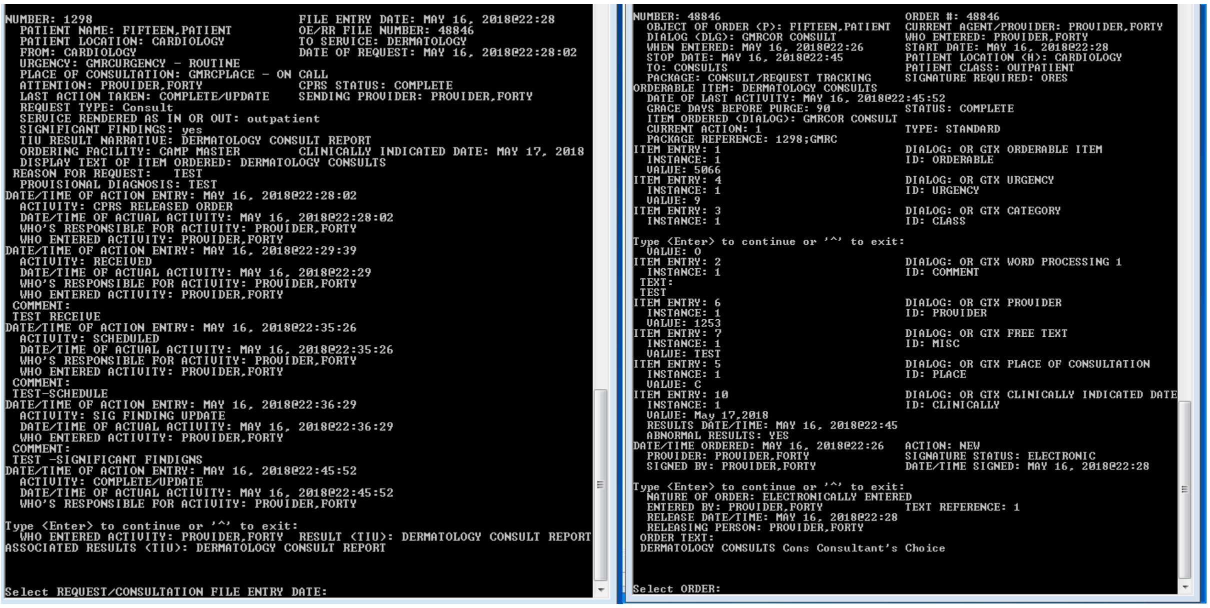
As seen in Figure 6, File 100 (order) and File 123 (Consult) are found to be manipulated constantly for a Consult order. All orders start at File 100, and when it is accepted, it becomes active and corresponding file entries are generated (File 123 record for a Consult). Although File 123 records each activity after acceptance of a Consult order, some date and status updates also occur at File 100. Figure 6 and Figure 7 exemplify the details on each file through CPRS Demo and CHUI screens after some transactions occurred. This practice allowed us to identify the majority of the fields on the *FileMan* files (File 100 and File 123 for Consults) that are revised and populated.



```
DERMATOLOGY CONSULTS Cons Consultant's Choice *UNSIGNED*

Activity:
05/16/2018 22:26  New Order entered by PROVIDER,FORTY (Physician)
     Order Text:        DERMATOLOGY CONSULTS Cons Consultant's Choice
     Nature of Order:   ELECTRONICALLY ENTERED
     Ordered by:        PROVIDER,FORTY (Physician)
     Signature:         NOT SIGNED

Current Data:
Treating Specialty:
Ordering Location:            CARDIOLOGY
Start Date/Time:
Stop Date/Time:
Current Status:               UNRELEASED
  Orders that have not been released to the service for action.
Order #48846

Order:
Consult to Service/Specialty: DERMATOLOGY CONSULTS
Reason for Request:
  TEST
Category:                     OUTPATIENT
Urgency:                      ROUTINE
Clinically Indicated Date:    May 17, 2018
Place of Consultation:        Consultant's Choice
Attention:                    PROVIDER,FORTY
Provisional Diagnosis:        TEST
```

```
     694 GMRCOR CONSULT   48846
CHOOSE 601-694: 694  48846
Another one:
Standard Captioned Output? Yes//   (Yes)
Include COMPUTED fields:  (N/Y/R/B): NO// BOTH Computed Fields and Record Number
 (IEN)

NUMBER: 48846                           ORDER #: 48846
  OBJECT OF ORDER (P): FIFTEEN,PATIENT  CURRENT AGENT/PROVIDER: PROVIDER,FORTY
  DIALOG (DLG): GMRCOR CONSULT          WHO ENTERED: PROVIDER,FORTY
  WHEN ENTERED: MAY 16, 2018@22:26      PATIENT LOCATION (H): CARDIOLOGY
  TO: CONSULTS                          PATIENT CLASS: OUTPATIENT
  PACKAGE: CONSULT/REQUEST TRACKING     SIGNATURE REQUIRED: ORES
ORDERABLE ITEM: DERMATOLOGY CONSULTS
  DATE OF LAST ACTIVITY: MAY 16, 2018@22:26:21
  GRACE DAYS BEFORE PURGE: 90           STATUS: UNRELEASED
  ITEM ORDERED (DIALOG): GMRCOR CONSULT
  CURRENT ACTION: 1                     TYPE: STANDARD
ITEM ENTRY: 1                           DIALOG: OR GTX ORDERABLE ITEM
  INSTANCE: 1                           ID: ORDERABLE
  VALUE: 5066
ITEM ENTRY: 4                           DIALOG: OR GTX URGENCY
  INSTANCE: 1                           ID: URGENCY
  VALUE: 9
ITEM ENTRY: 3                           DIALOG: OR GTX CATEGORY
  INSTANCE: 1                           ID: CLASS
  VALUE: 0
ITEM ENTRY: 2                           DIALOG: OR GTX WORD PROCESSING 1

Type <Enter> to continue or ’^’ to exit:
  INSTANCE: 1                           ID: COMMENT
 TEXT:
 TEST
ITEM ENTRY: 6                           DIALOG: OR GTX PROVIDER
  INSTANCE: 1                           ID: PROVIDER
  VALUE: 1253
ITEM ENTRY: 7                           DIALOG: OR GTX FREE TEXT
  INSTANCE: 1                           ID: MISC
  VALUE: TEST
ITEM ENTRY: 5                           DIALOG: OR GTX PLACE OF CONSULTATION
  INSTANCE: 1                           ID: PLACE
  VALUE: C
ITEM ENTRY: 10                          DIALOG: OR GTX CLINICALLY INDICATED DATE
  INSTANCE: 1                           ID: CLINICALLY
  VALUE: May 17,2018
DATE/TIME ORDERED: MAY 16, 2018@22:26   ACTION: NEW
  PROVIDER: PROVIDER,FORTY              SIGNATURE STATUS: NOT SIGNED
  NATURE OF ORDER: ELECTRONICALLY ENTERED
  ENTERED BY: PROVIDER,FORTY            TEXT REFERENCE: 1
  RELEASE STATUS: unreleased
 ORDER TEXT:
 DERMATOLOGY CONSULTS Cons Consultant’s Choice

Type <Enter> to continue or ’^’ to exit:


Select ORDER:
```

Figure 6. Order details (File 100) after order entry on CPRS Demo (left) and on CHUI screens (right).



```
NUMBER: 1298                            FILE ENTRY DATE: MAY 16, 2018@22:28
  PATIENT NAME: FIFTEEN,PATIENT         OE/RR FILE NUMBER: 48846
  PATIENT LOCATION: CARDIOLOGY          TO SERVICE: DERMATOLOGY
  FROM: CARDIOLOGY                      DATE OF REQUEST: MAY 16, 2018@22:28:02
  URGENCY: GMRCURGENCY - ROUTINE
  PLACE OF CONSULTATION: GMRCPLACE - ON CALL
  ATTENTION: PROVIDER,FORTY             CPRS STATUS: COMPLETE
  LAST ACTION TAKEN: COMPLETE/UPDATE    SENDING PROVIDER: PROVIDER,FORTY
  REQUEST TYPE: Consult
  SERVICE RENDERED AS IN OR OUT: outpatient
  SIGNIFICANT FINDINGS: yes
  TIU RESULT NARRATIVE: DERMATOLOGY CONSULT REPORT
  ORDERING FACILITY: CAMP MASTER        CLINICALLY INDICATED DATE: MAY 17, 2018
  DISPLAY TEXT OF ITEM ORDERED: DERMATOLOGY CONSULTS
 REASON FOR REQUEST:   TEST
  PROVISIONAL DIAGNOSIS: TEST
DATE/TIME OF ACTION ENTRY: MAY 16, 2018@22:28:02
  ACTIVITY: CPRS RELEASED ORDER
  DATE/TIME OF ACTUAL ACTIVITY: MAY 16, 2018@22:28:02
  WHO’S RESPONSIBLE FOR ACTIVITY: PROVIDER,FORTY
  WHO ENTERED ACTIVITY: PROVIDER,FORTY
DATE/TIME OF ACTION ENTRY: MAY 16, 2018@22:29:39
  ACTIVITY: RECEIVED
  DATE/TIME OF ACTUAL ACTIVITY: MAY 16, 2018@22:29
  WHO’S RESPONSIBLE FOR ACTIVITY: PROVIDER,FORTY
  WHO ENTERED ACTIVITY: PROVIDER,FORTY
 COMMENT:
 TEST RECEIVE
DATE/TIME OF ACTION ENTRY: MAY 16, 2018@22:35:26
  ACTIVITY: SCHEDULED
  DATE/TIME OF ACTUAL ACTIVITY: MAY 16, 2018@22:35:26
  WHO’S RESPONSIBLE FOR ACTIVITY: PROVIDER,FORTY
  WHO ENTERED ACTIVITY: PROVIDER,FORTY
 COMMENT:
 TEST-SCHEDULE
DATE/TIME OF ACTION ENTRY: MAY 16, 2018@22:36:29
  ACTIVITY: SIG FINDING UPDATE
  DATE/TIME OF ACTUAL ACTIVITY: MAY 16, 2018@22:36:29
  WHO ENTERED ACTIVITY: PROVIDER,FORTY
 COMMENT:
 TEST -SIGNIFICANT FINDIGNS
DATE/TIME OF ACTION ENTRY: MAY 16, 2018@22:45:52
  ACTIVITY: COMPLETE/UPDATE
  DATE/TIME OF ACTUAL ACTIVITY: MAY 16, 2018@22:45:52
  WHO’S RESPONSIBLE FOR ACTIVITY: PROVIDER,FORTY

Type <Enter> to continue or ’^’ to exit:
  WHO ENTERED ACTIVITY: PROVIDER,FORTY  RESULT (TIU): DERMATOLOGY CONSULT REPORT
ASSOCIATED RESULTS (TIU): DERMATOLOGY CONSULT REPORT


Select REQUEST/CONSULTATION FILE ENTRY DATE:
```

```
NUMBER: 48846                           ORDER #: 48846
  OBJECT OF ORDER (P): FIFTEEN,PATIENT  CURRENT AGENT/PROVIDER: PROVIDER,FORTY
  DIALOG (DLG): GMRCOR CONSULT          WHO ENTERED: PROVIDER,FORTY
  WHEN ENTERED: MAY 16, 2018@22:26      START DATE: MAY 16, 2018@22:28
  STOP DATE: MAY 16, 2018@22:45         PATIENT LOCATION (H): CARDIOLOGY
  TO: CONSULTS                          PATIENT CLASS: OUTPATIENT
  PACKAGE: CONSULT/REQUEST TRACKING     SIGNATURE REQUIRED: ORES
ORDERABLE ITEM: DERMATOLOGY CONSULTS
  DATE OF LAST ACTIVITY: MAY 16, 2018@22:45:52
  GRACE DAYS BEFORE PURGE: 90           STATUS: COMPLETE
  ITEM ORDERED (DIALOG): GMRCOR CONSULT
  CURRENT ACTION: 1                     TYPE: STANDARD
  PACKAGE REFERENCE: 1298;GMRC
ITEM ENTRY: 1                           DIALOG: OR GTX ORDERABLE ITEM
  INSTANCE: 1                           ID: ORDERABLE
  VALUE: 5066
ITEM ENTRY: 4                           DIALOG: OR GTX URGENCY
  INSTANCE: 1                           ID: URGENCY
  VALUE: 9
ITEM ENTRY: 3                           DIALOG: OR GTX CATEGORY
  INSTANCE: 1                           ID: CLASS

Type <Enter> to continue or ’^’ to exit:
  VALUE: 0
ITEM ENTRY: 2                           DIALOG: OR GTX WORD PROCESSING 1
  INSTANCE: 1                           ID: COMMENT
 TEXT:
 TEST
ITEM ENTRY: 6                           DIALOG: OR GTX PROVIDER
  INSTANCE: 1                           ID: PROVIDER
  VALUE: 1253
ITEM ENTRY: 7                           DIALOG: OR GTX FREE TEXT
  INSTANCE: 1                           ID: MISC
  VALUE: TEST
ITEM ENTRY: 5                           DIALOG: OR GTX PLACE OF CONSULTATION
  INSTANCE: 1                           ID: PLACE
  VALUE: C
ITEM ENTRY: 10                          DIALOG: OR GTX CLINICALLY INDICATED DATE
  INSTANCE: 1                           ID: CLINICALLY
  VALUE: May 17,2018
  RESULTS DATE/TIME: MAY 16, 2018@22:45
  ABNORMAL RESULTS: YES
DATE/TIME ORDERED: MAY 16, 2018@22:26   ACTION: NEW
  PROVIDER: PROVIDER,FORTY              SIGNATURE STATUS: ELECTRONIC
  SIGNED BY: PROVIDER,FORTY             DATE/TIME SIGNED: MAY 16, 2018@22:28

Type <Enter> to continue or ’^’ to exit:
  NATURE OF ORDER: ELECTRONICALLY ENTERED
  ENTERED BY: PROVIDER,FORTY            TEXT REFERENCE: 1
  RELEASE DATE/TIME: MAY 16, 2018@22:28
  RELEASING PERSON: PROVIDER,FORTY
 ORDER TEXT:
 DERMATOLOGY CONSULTS Cons Consultant’s Choice


Select ORDER:
```

Figure 7. Order details after completion on File 123 (left) and File 100 (right).

2.3 MAPPING VA *FILEMAN* FILES TO CDW FIELDS

The automated HIT Hazard Detection (HD) tool is designed to monitor the operational CDW. While clinical users interact with third-party applications (i.e., CPRS Demo) and the CHUI screens, the *FileMan* files are kept in local VistA instances and the regional databases. Transformation of the regional data goes through multiple filters and aggregations before the data is stored in CDW. Daily extract, transform, and load (ETL) processes (in batches) handle this data transfer. Figure 8 represents the dimensions of a VistA instance and CDW. Since the details of this transformation are not well known on the ORNL side, we treat this process as a black box for now and map the fields to columns and observe CDW for anomalies.

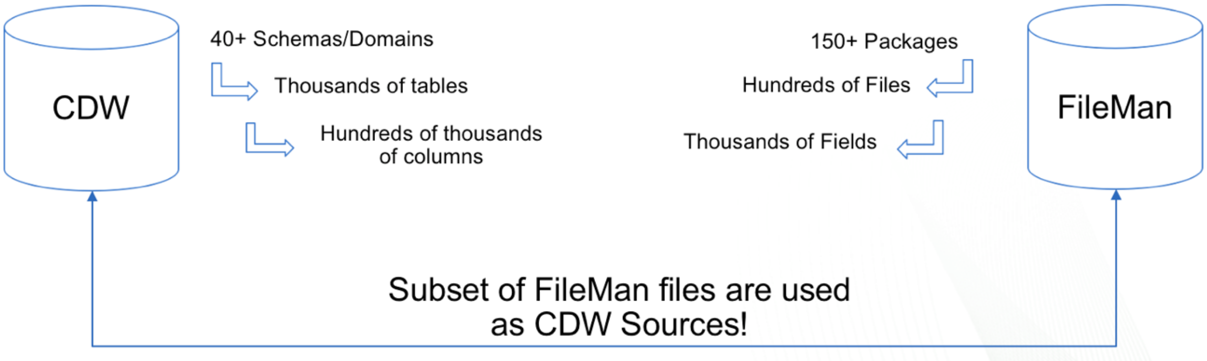


Figure 8. CDW and *FileMan* dimensions.

2.4 DATA MODELS

We gained significant insights on how the data is collected while conducting the analysis described in the previous sections. Clinical orders are found to be the most important atomic units of VistA, and they are of particular interest for safety officers in the VHA. Moreover, transactions that occur during the life cycle of an order can be indicative of adverse events and interruptions in care delivery. For this purpose, we focused our efforts on revealing significant patterns in order-related data.

2.4.1 Data Model for Count-Based Detectors

Anomalous numbers of order cancellations and rejections are found to be recently relevant and frequently occurring patterns. Based on our discussions with subject matter experts (SMEs) at the VHA and our exploratory querying of CDW, we designed a data model (aggregate table structure) that incorporates various information extracted from different tables in CDW. The table counts the number of order discontinuations per day, per VistA station (a.k.a. VHA facility), per discontinuation type (orders without *PackageReference* are considered as Rejected, the rest are Cancelled), per discontinuation reason, per discontinuation nature, and per creator (human vs. machine). Sample data structure is shown in Table 1.

Table 1. Data model for count-based hazard detection

Year	Month	Day	Sta3n	Reject	Staff	Nature	Reason	Freq	Created Date
2019	01	15	539	0	1	REJECT	OBSOLETE	120	'2019-01-16'
2019	01	15	539	1	0	OBSOLETE	FREE-TEXT	100	'2019-01-16'

In Table 1, when Reject = 1, then the count is for rejected orders, and when Staff = 1, it is for human-generated orders. Therefore, Reject = 0 is for cancelled orders, and Staff = 0 is for machine-generated orders. These frequency (Freq) numbers are calculated for each domain (Consults, Radiology, Laboratory, and Outpatient Medication) as a separate table. We have generated an analysis database called HITHD and updated these tables daily via stored procedures. Every day (at 8 am), these stored procedures are run as a job, and they basically create the counts data after the ETL processes are finished and new data is received into the CDW. We designed these stored procedures so that when the daily CDW update is not finished or there is ongoing maintenance in the CDW, the tables are kept the same. When new data arrives, the next cycle creates the counts data since the last update date. This way we continuously capture the daily counts without any gap.

2.4.2 Data Model for Event Sequences

Unfortunately, the clinical workflows and event logs may or may not reveal themselves in an easily distinguishable manner in the CDW. Usually, relevant information is scattered around the database in multiple tables and sometimes can be discerned only with examination and denormalization of the data. Thus, we first performed an arduous task identifying all date records that are assumed to represent a transaction on an order. All columns/tables that have some form of activity or status update associated with orders are also recorded. All these relevant records are captured from the *CPRSOrder* domain and the relevant domains for each order type. Then we represented them as time-sorted event sequences per order that we call raw event-sequence data. The purpose of this task is to reverse engineer VistA event logs from the CDW data columns. A sample order event sequence for the Radiology domain (pipe delimited):

```
DesiredNotGuaranteedDateTime,2017-01-03 00:00:00|OrderStartDateTime,2017-01-03
00:00:00|RequestedDateTime,2017-01-03 00:00:00|EnteredDateTime,2017-01-03 12:01:00|NW,2017-01-
03 12:01:00|ReleaseDateTime,2017-01-03 12:01:00|SignedDateTime,2017-01-03
12:01:00|COMPLETE,2017-01-03 12:01:48|VistaCreateDate,2017-01-03
12:30:22|ReportedDateTime,2017-01-09 00:00:00|EXAM ENTRY,2017-01-09
10:11:00|ExamDateTime,2017-01-09 10:11:00|WAITING FOR EXAM,2017-01-09
10:11:00|EXAMINED,2017-01-09 10:31:00|UPDATE STATUS,2017-01-09
10:31:00|COMPLETE,2017-01-09 11:13:00|LastActivityDateTime,2017-01-09
11:13:00|OrderStopDateTime,2017-01-09 11:13:00|ReportEnteredDateTime,2017-01-09
11:13:00|ResultsDateTime,2017-01-09 11:13:00|VerifiedDateTime,2017-01-09
11:13:00|VistaEditDate,2017-01-09 11:46:11
```

This raw event-sequence data is extracted in an ad hoc manner. To explore and make sense of raw data, we focused our analysis on a cohort (approximately 808,000 patients) diagnosed with IHD between January 1, 2017, and January 1, 2018. For example, for the Radiology domain, we extracted radiology orders that were started in the year 2017 for the identified cohort. This resulted in approximately 860,000 radiology orders. The data structure for radiology event sequences incorporated 36 dates, 5 actions, and 76 status updates as distinct event types. The goal is to represent these raw event sequences as well-defined event logs that will decrease the complexity and exhibit the main transactions that occurred in orders. For this purpose, we selected OASIS WS-Human Task specification as a standard to map the raw event sequences. The following section describes the OASIS standard.

2.4.2.1 OASIS WS-Human Task Specification

The WS-Human Task Specification [1] is an OASIS (https://www.oasis-open.org/org) standard. OASIS is a nonprofit consortium that drives the development, convergence, and adoption of open standards for the global information society. The concept of human tasks is used to specify work that must be accomplished by people. Typically, human tasks are part of business processes. However, those tasks can also be used to design human interactions that are invoked as services, whether as part of a process or otherwise.

We selected OASIS WS-Human Task Specification Version 1.1 [1], specifically, the Human Task Behavior and State Transitions diagram (see page 55 of [1]) to decrease the number of event types and distinguish important stages that raw event sequences exhibit. The WS-Human Task Specification introduces the definition of human tasks, including their properties, behavior, and a set of operations used to manipulate human tasks. In addition, it introduces a coordination protocol that uses web services to control the autonomy and life cycle of service-enabled human tasks in an interoperable manner. The human task behavior and state transitions are as follows:

1. Upon creation, a task goes into the *Created* state. There is no need to have a task owner in this state. The task remains in the *Created state* until it is activated and has potential owners.
2. When a task has multiple potential owners or is assigned to a work queue, it transitions into the *Ready* state, indicating that it can be claimed by one of its potential owners.
3. When the task is claimed by a single owner, it transitions into the *Reserved* state, indicating that it is assigned to a single actual owner. The current actual owner of a human task can release a task to again make it available for all potential owners.
4. Once work is started on a task that is in the *Ready* or *Reserved* state, it goes into the *InProgress* state, indicating that it is being worked on.
5. The task will go into the *Suspended* state when a suspend operation is invoked or a suspend event is received.
6. On successful completion of the work, the task transitions into the *Completed* final state (termination).
7. On unsuccessful completion of the work, the task transitions into the *Failed* final state (termination).
8. A received exit event will make the task transition into the *Exit* final state (termination).
9. A nonrecoverable error event will make the task transition into the *Error* final state (termination).
10. A skip operation invoked (and if the task is skippable) will cause a task to go to the *Obsolete* final state (termination).

The life cycle of subtasks is the same as that of the main task. The specification allows state transitions among all states except the termination states. In the next subsection, we describe how we mapped the raw data to OASIS for the Radiology domain.

2.4.2.2 Mapping Raw Data to OASIS

We first applied temporal clustering on the raw event-sequence data to extract clusters of events that occurred together in a two-minute time window. Usually, the dates have minute-level data (no seconds are recorded) and the relevant dates and columns are updated together as a result of a transaction on an order. We adopted the two-minute time window because of system lags that we seldom observe. Note that different dates in different tables might represent the same information (such as Results Date Time and Order Stop Date Time). The purpose of this activity is to decompose the raw event sequence into coherent blocks of events that are triggered by the same transaction or encounter (appointment, phone call, visit, etc.). This helped us to identify important events in each cluster that could be mapped to transitions per the OASIS standard. Figure 9 represents the raw events that are temporally clustered for the sample order presented previously.

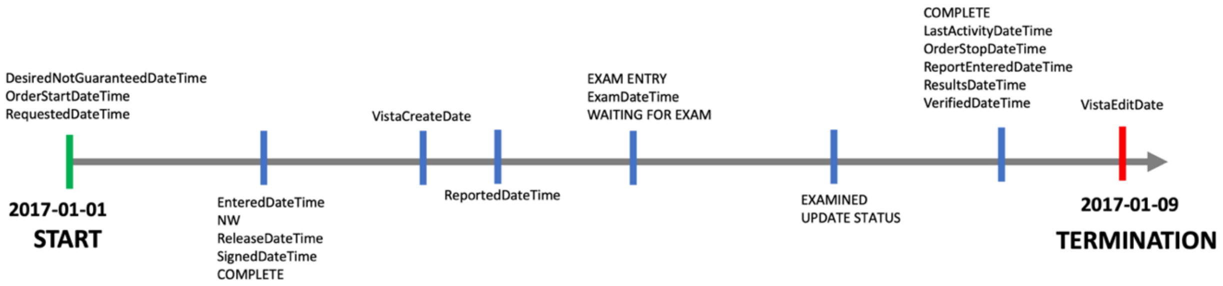


Figure 9. Temporal clustering of the sample order illustrated on a timeline.

When all distinct event types and possible links between them are considered, displaying a state transition diagram would result in a complicated "spaghetti diagram." The interpretability of the data suffers from this complexity, and it is hard to extract actionable insights. Therefore, in discussion with domain experts at the VA, we mapped the raw event sequences to the OASIS WS-Human Task Specification. The finalized rules of the mapping algorithm are summarized in Table 2, where the left column has the OASIS states that are considered relevant in the Radiology domain of the CDW (note that the Obsolete state is ignored) and there right column explains the conditions at which orders transition. These rules are applied sequentially on each cluster of events to extract OASIS state transitions per order.

Table 2. Mapping rules that generate simplified event logs from raw event-sequence data

Transition to	Rule/s
Created	If Entered Date Time is found, transition to Created.
Ready	If Released Date Time and a new (NW) action are taken, transition to Ready. If Released Date Time and a release hold (RL) action are taken, and order is at Suspended, transition to Ready. If Released Date Time and Signed Date Time are found, and order is not at Ready and no Order Stop Date Time is recorded, transition to Ready.
Suspended	If Released Date Time and a hold (HD) action are taken, transition to Suspended.
Reserved	If EXAM ENTRY is found, transition to Reserved.
InProgress	If any cluster of Dates or Actions takes place after transitioning to Reserved, transition to InProgress.
Completed	If there is Results Date Time, transition to Completed.
Failed	If order is at Completed and there is Discontinued Date Time recorded, then transition to Failed.
Error	If order is at Ready and there is Discontinued Date Time recorded, then transition to Error.
Exited	If there is Discontinued Date Time recorded (else: not at Completed or Ready), then transition to Exited.

In addition to Table 2, a couple of additional assumptions are made. First, we consider only event types that occur after the first event, which is Entered Date Time. Another assumption is to ignore the second transition when two events are considered as a trigger of the same transition. For example, EXAM ENTRY and Exam Date Time records follow each other; however, the order of their appearance can be different because of the communication lags in VistA updates (either Exam Date Time or EXAM ENTRY can occur first). In those cases, we allow only one transition to account for the redundancy.

2.4.2.3 Other Domains

In addition to the Radiology domain, we prepared raw event sequences for Laboratory, Consults, and Outpatient Medication (RxOut) domains. Raw event-sequence sample data for a Laboratory order:

```
|EnteredDateTime,2017-05-02 13:55:00|NW,2017-05-02 13:55:00|ReleaseDateTime,2017-05-02
13:56:00|SignedDateTime,2017-05-02 13:56:00|VistaCreateDate,2017-05-02
15:17:17|OrderStartDateTime,2017-05-05 09:36:00|LabChemSpecimenDateTime,2017-05-05
09:36:01|LastActivityDateTime,2017-05-05 11:39:00|OrderStopDateTime,2017-05-05
11:39:00|ResultsDateTime,2017-05-05 11:39:00|LabChemCompleteDateTime,2017-05-05
11:39:17|VistaEditDate,2017-05-05 12:21:25
```

Consult orders have additional activities in the CDW that are the closest records to an event log. These additional activities are relatively informative to represent transactions. Raw event-sequence sample data for a Consult order:

```
ReleaseDateTime,2017-04-05 11:56:00|SignedDateTime,2017-04-05 11:56:00|EnteredDateTime,2017-
04-05 11:56:00|NW,2017-04-05 11:56:00|OrderStartDateTime,2017-04-05 11:56:00|CPRS RELEASED
ORDER,2017-04-05 11:56:33|RequestDateTime,2017-04-05 11:56:33|PRINTED TO,2017-04-05
11:56:34|VistaCreateDate,2017-04-05 17:56:30|RECEIVED,2017-04-10 14:26:44|ADDED
COMMENT,2017-04-10 14:27:09|ADDED COMMENT,2017-06-07
12:00:19|DiscontinuedDateTime,2017-06-07 15:19:00|OrderStopDateTime,2017-06-07
15:19:00|LastActivityDateTime,2017-06-07 15:19:00|DISCONTINUED,2017-06-07 15:19:41|PRINTED
TO,2017-06-07 15:19:42|VistaEditDate,2017-06-07 15:22:05
```

Raw event-sequence sample data for a RxOut order:

```
|DispensedDate,2017-06-19 00:00:00|IssueDate,2017-06-19 00:00:00|LoginDate,2017-06-19
00:00:00|EnteredDateTime,2017-06-19 14:45:00|NW,2017-06-19 14:45:00|ReleaseDateTime,2017-06-19
14:45:00|SignedDateTime,2017-06-19 14:45:00|ChartCopyPrintedDateTime,2017-06-19
14:45:31|VistaCreateDate,2017-06-19 14:45:41|FinishingDate,2017-06-19 15:15:51|HD,2017-06-19
15:16:00|ReleaseDateTime,2017-06-19 15:16:00|FillDateTime,2017-06-20 00:00:00|LoginDate,2017-06-
20 00:00:00|OrderStartDateTime,2017-06-20 00:00:00|DiscontinuedHoldUntilDateTime,2017-06-20
08:46:00|ReleaseDateTime,2017-06-20 08:46:00|RL,2017-06-20 08:46:00|ReleaseDateTime,2017-06-20
08:50:37|ReleaseDateTime,2017-06-22 18:32:42|CompletedDateTime,2017-09-17
00:00:00|OrderStopDateTime,2017-09-17 00:00:00|LastActivityDateTime,2017-09-18
20:00:00|VistaEditDate,2017-09-18 20:02:07
```

Laboratory and Consult domain event-sequence data is also generated with the same method and similar mapping rules (OASIS) as the Radiology domain. RxOut mapping is ongoing and will be accomplished in the next period of performance. See [2] for a comprehensive summary on the raw event sequences and the mapped event sequences.

2.4.3 Count-Based Data for Detection—Statistical Process Control

Statistical process control (SPC) detectors work on daily counts of discontinued orders (rejected or cancelled) for different stations and domains. Basically, the detector calculates mean and standard deviation (std) of the counts going back for a year from the detection day. Subsequently, it removes the anomalies (the counts occurred above mean+3std) from the yearly data and recalculates the new mean and std. The detection is done on this cleaned data (to better capture the normal mean) for the counts that exceed the new mean+3std. If the detection is done, the detection decision logic and how the aggregate order counts break into different nature and reason of discontinuations and staff kind are presented. There are a couple of assumptions that go into the detector. Weekends and Weekdays are treated separately, mean is calculated for one year, mean+3std is assumed as the upper limit, and detection is done only for the frequencies that are greater than 10. Also, confidence is assigned to detections based on the number of stds for which the frequency is higher than the mean (3–5 stds = Low, 6–8 stds = Medium, and 8+ stds = High confidence). The summary of the activity flow for an SPC-based detector is in Figure 10.

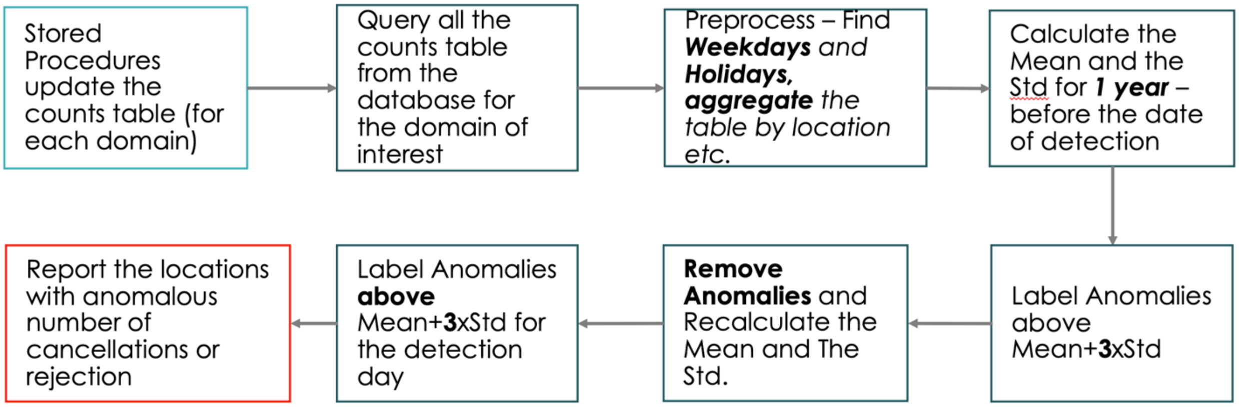

Stored Procedures update the counts table (for each domain)
Query all the counts table from the database for the domain of interest
Preprocess – Find Weekdays and Holidays, aggregate the table by location etc.
Calculate the Mean and the Std for 1 year – before the date of detection
Label Anomalies above Mean+3xStd
Remove Anomalies and Recalculate the Mean and The Std.
Label Anomalies above Mean+3xStd for the detection day
Report the locations with anomalous number of cancellations or rejection

Figure 10. Activity flow of the SPC detector.

Since the initial implementation does not have any labeled data, see [3] for details on how we detect outliers in the data using the upper bound in a SPC technique as a threshold. Data points above the threshold are regarded as outliers. Figure 11 represents a sample detection from the tool. Figure 12 represents the additional information for the detection in Figure 11 that is provided to the analysts for further investigation. Along with this information, we also provide the distribution of human- and machine-generated orders and how many distinct orderable items and patients are impacted from the cancellations or rejections for that particular station and day.

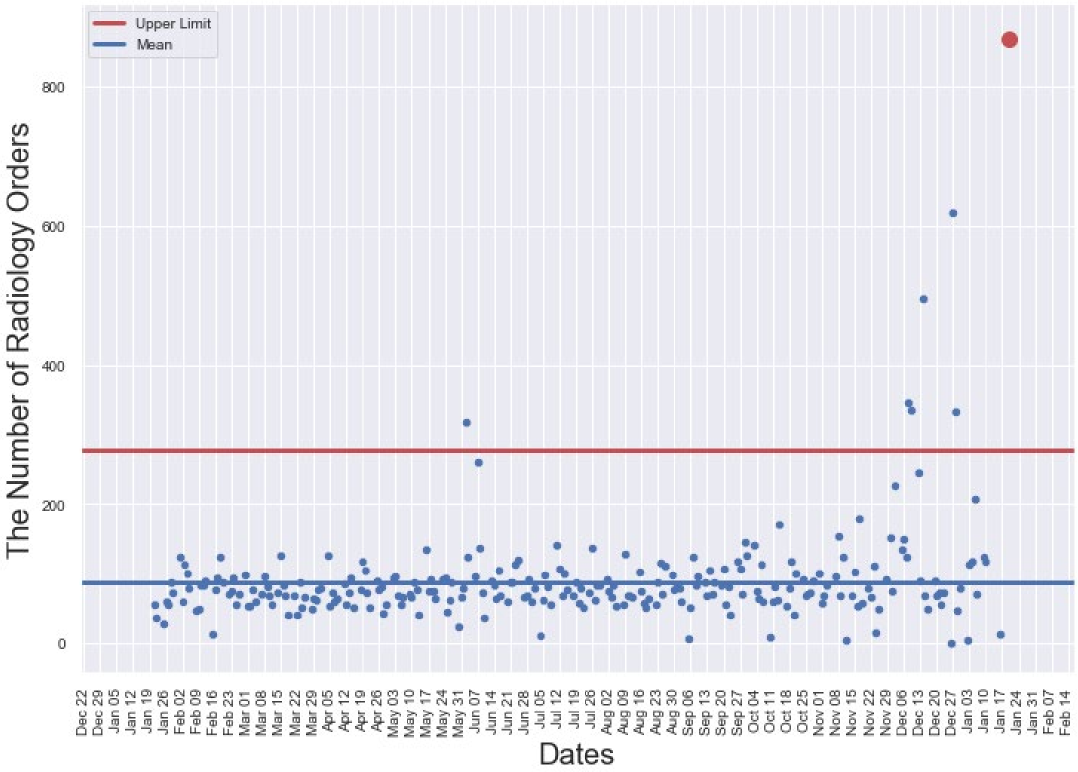


Figure 11. Detection of anomalous number of radiology order cancellations at a clinical location on January 20, 2017. The red line represents the upper bound, and the red dot is the detection of an anomalous number of cancellations.

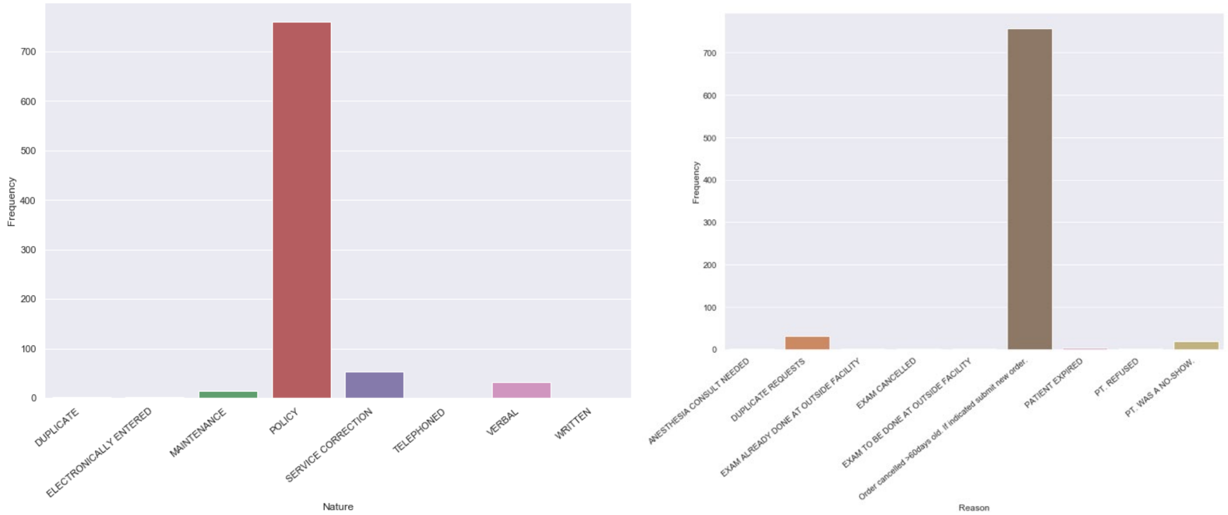


Figure 12. Distribution of Nature of discontinuations (left) and distribution of Reason of discontinuations.

2.4.4 Event-Sequence Data for Detection

Raw event sequences are used for multiple exploratory analysis such as calculating statistics on durations between transitions. In the following sections we describe two exploratory analyses that are shared with SMEs and detailed in different publications.

2.4.4.1 Latent Dirichlet Analysis

We divided Consults raw event-sequence data into three groups by associating their stop date with a termination activity. These groups are: (i) Completed (~1.8 M orders), (ii) Discontinued (~375 K orders), and (iii) Cancelled (~52 K orders). We found out that terminated orders can only be in one of these three states. Completed orders are the ones that transitioned through all necessary states. Discontinued orders may or may not make it all the way to the Consult service while cancelled orders are the ones cancelled by the service. In this analysis, we focused only on datasets for Discontinued and Cancelled orders since they are more likely to incorporate orders with adverse events. In addition to the aforementioned processing steps, we enriched the event space of the dataset by combining the events with time intervals. We associated the time intervals between events with wait times as follows: "0," "<1min," "<1hour," "<1day," "<1week," "<1month," and "1month+," where "0" represents the initial event. The latter strings are named 'time bins' herein. As an example, if event B happened two hours after the first event A, we represent event B as "B_<1day" and event A as "A_0." We applied the time bins to the Discontinued and Cancelled orders and prepared the final datasets. The new data structure is described subsequently:

Order-ID | (A_0, Timestamp) | … | (Z_<1week, Timestamp)

Latent Dirichlet analysis (LDA) is a statistical model that represents each document as a mixture of arbitrary number of latent topics/groups. It leverages the bag-of-words approach to explain the probabilities of falling into each topic for each document. LDA is widely used for document classification in natural language processing tasks and has already been applied to analyze event sequences on healthcare services of patients [4]. We applied LDA as described in the sequel. Instead of following the clinical history of patients and treating patient history as a document, each order is defined as a single document. Each event type (different dates and activities) in our corpus (event sequences) is defined as a word in our dictionary. We specifically used the open source Gensim package and Mallet's LDA implementation [5] to run LDA on the datasets prepared. Additionally, we leveraged another open source tool LDAvis [6] on the LDA results and visualized the event frequencies and clusters to discern patterns in the data. Since LDA is an unsupervised learning algorithm, by exploring the relevant frequencies of events in each topic, the user can get insights on the context of the orders and the ability to define potential hazardous patterns. Please see [4] for the details and results of this analysis.

2.4.4.2 Process Mining

We applied the rules in Table 2 and generated the mapped event logs data (of Radiology). Application of process mining resulted in a simplified state transition graph. Since we have the time stamps of the state transitions, we could generate the state transition graph with the mean number of days that passes in between those transitions. This information is quite useful to identify transitions on which the orders are likely to stall. Domain experts can easily examine these graphs to identify irregular and rare transitions, apply heuristics, and calculate statistics. Figure 13 emphasizes the most common path with thicker transition lines and shows the start and termination transitions with dashed lines. Most orders exhibit a regular transition pattern that follows Created, Ready, Reserved, InProgress, and Completed in order. After discussion with domain experts, we determined some candidate hazards such as transitions from Completed to Failed. However, further analysis is needed to determine more complex hazard conditions for other transitions by combining durations and distributions of occurrences. Such an analysis is done in [5], and other process mining applications on other datasets can be found in [2].

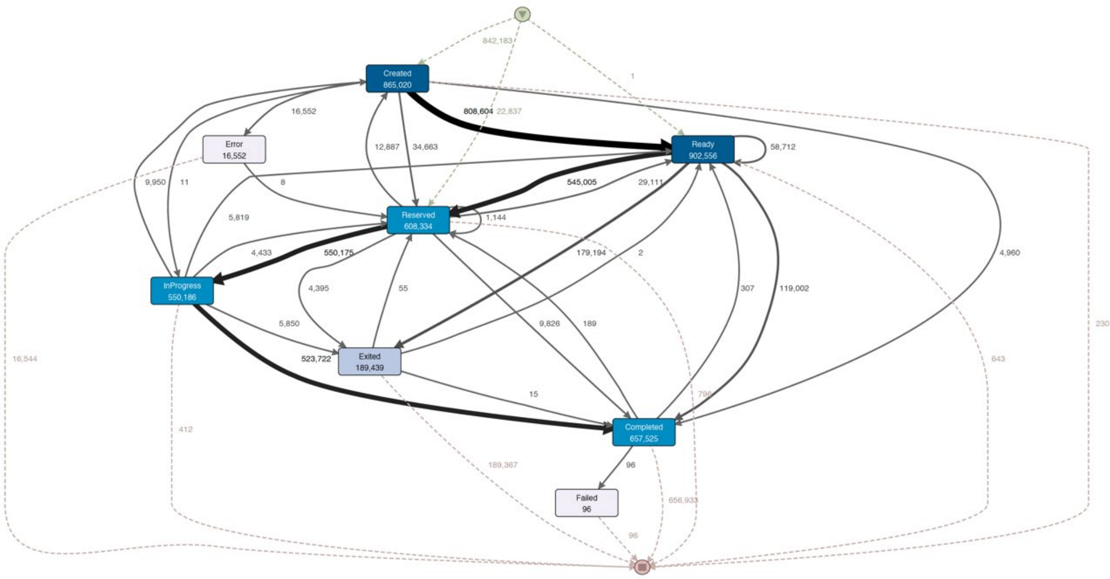


Figure 13. Sample state transition diagram with frequencies of transaction generated by the process mining tool Disco.[4]

2.4.4.3 Other Exploratory Analysis

Along with LDA and process mining results, we explored techniques such as Word2vec [6] and graph pruning to understand the raw event sequences we extracted. We leveraged a parallel version of Word2vec implemented at ORNL to find time-relevant embeddings (word/event vectors) of events and used those embeddings to predict the context of event sequences. Using clustering techniques, we were able to distinguish Cancelled, Discontinued, and Completed orders, but additional analysis was needed (see Figure 14).

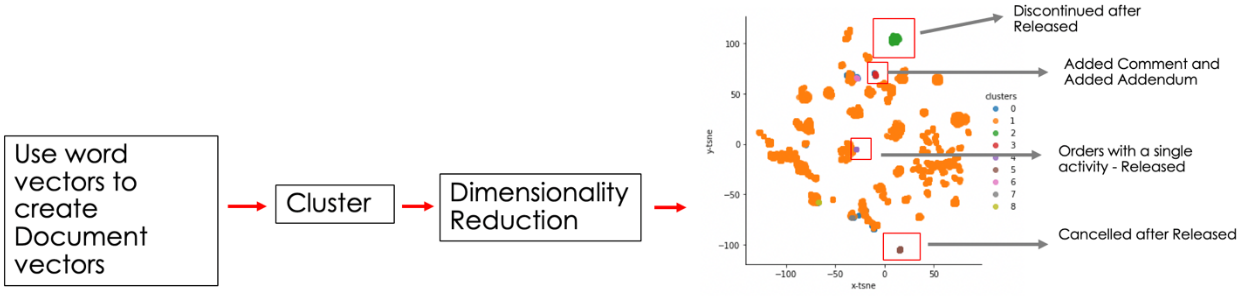


Figure 14. Sample clustering on results by Word2Vec implementation.

Regarding graph pruning, we calculated the state transition matrix for event sequences (state = event + time bin) on the Consult data that we applied LDA on and were able to predict the orders that are likely to be cancelled at earlier stages (see Figure 15). This was possible because of time bins we incorporated into the state space. However, this approach needs further investigation on which of the predicted cancellations could be classified as hazards.

[4] https://fluxicon.com/disco/.

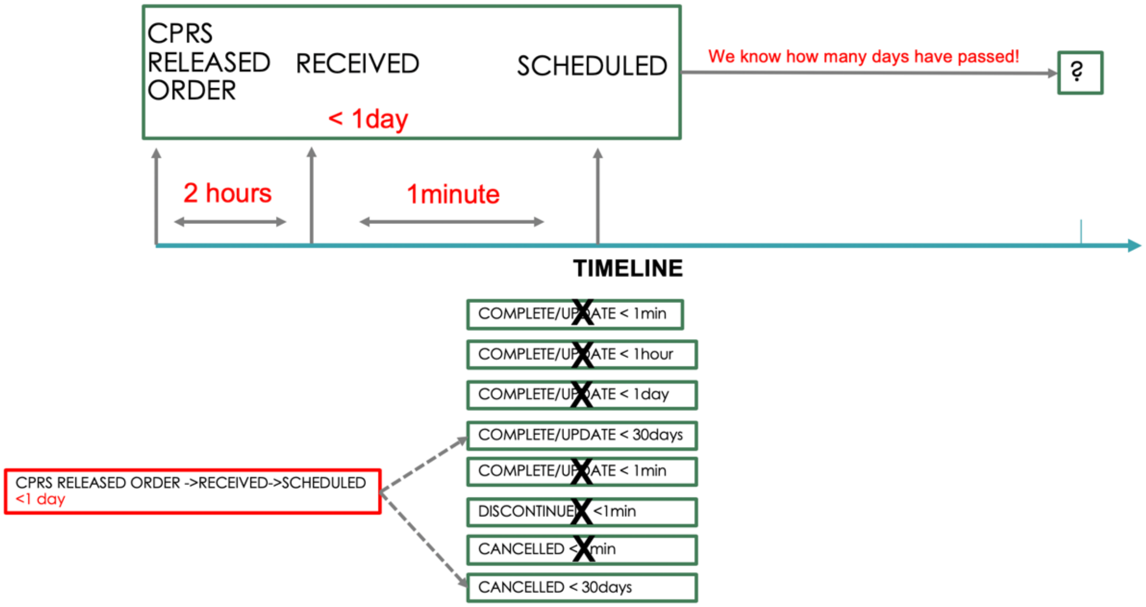


Figure 15. Timeline of an order to illustrate pruning technique. Based on what state the order is and how much time has passed since the last transition, high probability transitions of the next state can reveal cancelled orders at earlier stages.

2.5 WEB CRAWLER ON OSEHRA VISTA

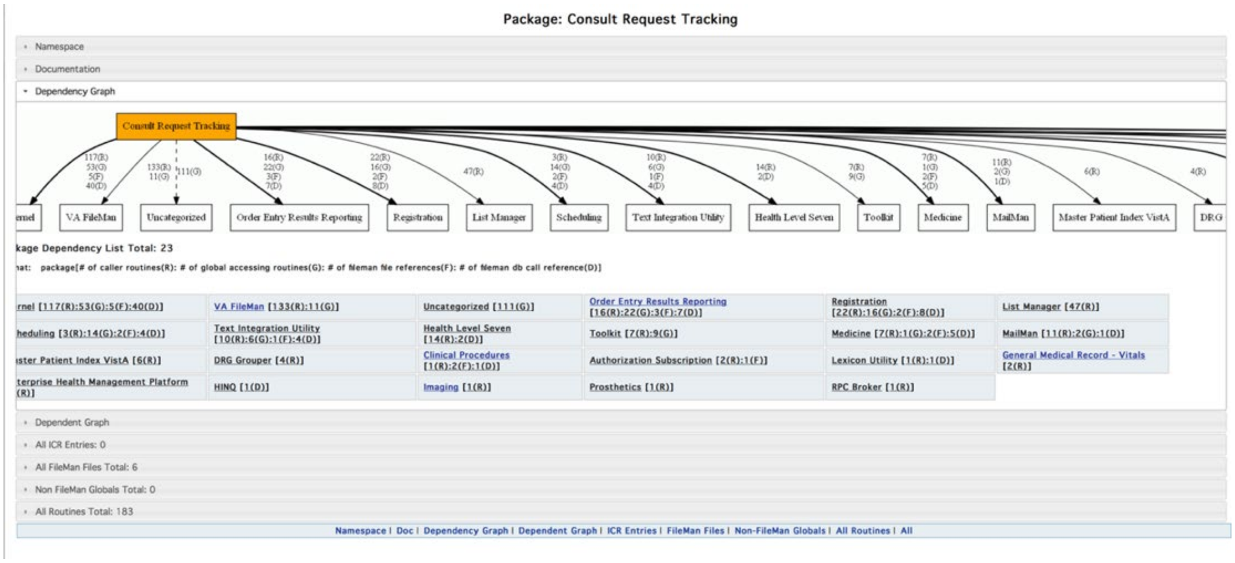
Additionally, an automated web crawler is implemented to extract field information for each *FileMan* file from the code repository documentation of OSEHRA.[5] The extracted information includes the field name, data structure of the field from which we can infer how it is populated, and whether or not it is a required field.

To achieve the third mapping (VistA *FileMan* to CDW), the meta schema/domain in CDW is also leveraged. If the source of a CDW column is VistA, we are able to map the desired *FileMan* files and fields to columns in CDW. This capability enables us to extract related columns in CDW that used identified data fields in the *FileMan* system as a source and calculate statistics for the probabilistic model. The auto-surveillance system will be observing these fields to identify anomalies and hazards.

2.5.1 Additional Data Resources

Additionally, three major resources have been identified that can provide information on the interconnectedness of the VistA system. The first resource is the cross-reference documentation of OSEHRA VistA.[6] This documentation includes *FileMan* data dictionary and graphs denoting dependencies between packages in VistA. The source tree repository[7] provides source data that feeds the cross-reference documentation. The repository was explored, and the source data of the documentation was extracted for the dependency information. This will help us identify the number of the fields queried/populated by each package and the number of routines called. A typical dependency graph is shown in Figure 16, and the accompanying JSON file is shown in Figure 17.

[5] http://code.osehra.org/dox/
[6] http://code.osehra.org/dox/
[7] https://github.com/OSEHRA-Sandbox/VistA-M/tree/FOIA_Dec_2017



Package: Consult Request Tracking

- Namespace
- Documentation
- Dependency Graph

kage Dependency List Total: 23

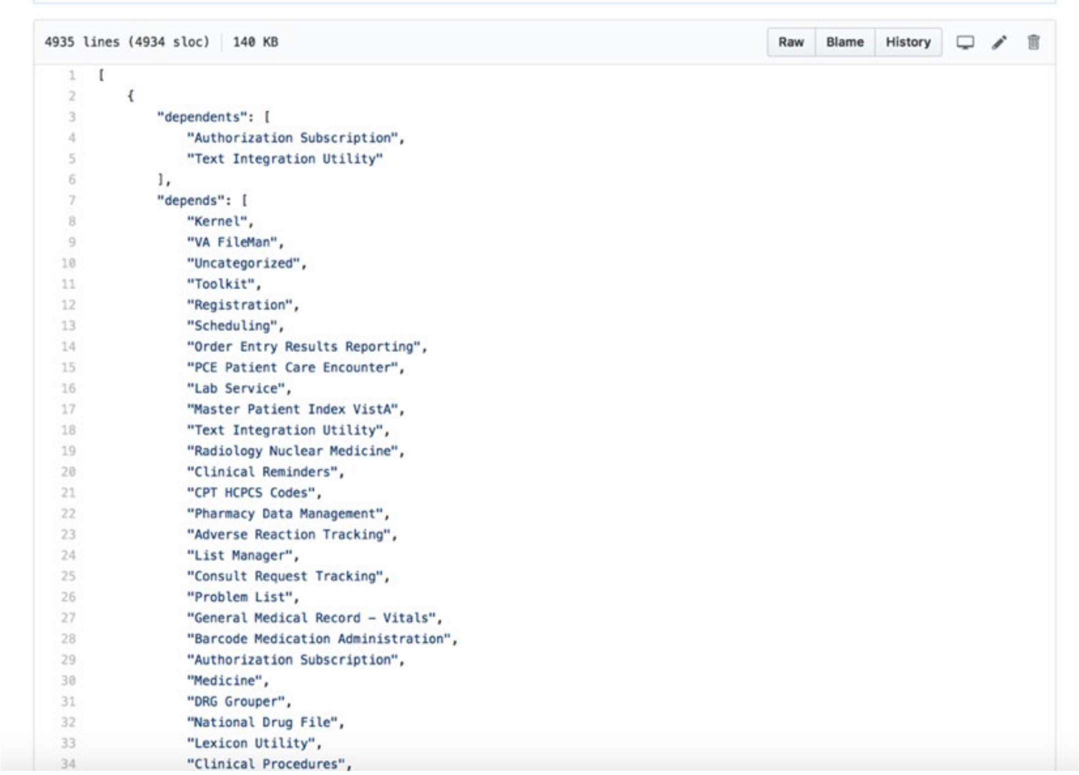
iat: package[# of caller routines(R): # of global accessing routines(G): # of fileman file references(F): # of fileman db call reference(D)]

rnel [117(R):53(G):5(F):40(D)]	VA FileMan [133(R):11(G)]	Uncategorized [111(G)]	Order Entry Results Reporting [16(R):22(G):3(F):7(D)]	Registration [22(R):16(G):2(F):8(D)]	List Manager [47(R)]
heduling [3(R):14(G):2(F):4(D)]	Text Integration Utility [10(R):6(G):1(F):4(D)]	Health Level Seven [14(R):2(D)]	Toolkit [7(R):9(G)]	Medicine [7(R):1(G):2(F):5(D)]	MailMan [11(R):2(G):1(D)]
ister Patient Index VistA [6(R)]	DRG Grouper [4(R)]	Clinical Procedures [1(R):2(F):1(D)]	Authorization Subscription [2(R):1(F)]	Lexicon Utility [1(R):1(D)]	General Medical Record - Vitals [2(R)]
terprise Health Management Platform (R)]	HINQ [1(D)]	Imaging [1(R)]	Prosthetics [1(R)]	RPC Broker [1(R)]	

- Dependent Graph
- All ICR Entries: 0
- All FileMan Files Total: 6
- Non FileMan Globals Total: 0
- All Routines Total: 183

Namespace | Doc | Dependency Graph | Dependent Graph | ICR Entries | FileMan Files | Non-FileMan Globals | All Routines | All

Figure 16. Example (Consult Request Tracking) dependency information from VistA cross-reference documentation.



4935 lines (4934 sloc) | 140 KB

```
[
    {
        "dependents": [
            "Authorization Subscription",
            "Text Integration Utility"
        ],
        "depends": [
            "Kernel",
            "VA FileMan",
            "Uncategorized",
            "Toolkit",
            "Registration",
            "Scheduling",
            "Order Entry Results Reporting",
            "PCE Patient Care Encounter",
            "Lab Service",
            "Master Patient Index VistA",
            "Text Integration Utility",
            "Radiology Nuclear Medicine",
            "Clinical Reminders",
            "CPT HCPCS Codes",
            "Pharmacy Data Management",
            "Adverse Reaction Tracking",
            "List Manager",
            "Consult Request Tracking",
            "Problem List",
            "General Medical Record - Vitals",
            "Barcode Medication Administration",
            "Authorization Subscription",
            "Medicine",
            "DRG Grouper",
            "National Drug File",
            "Lexicon Utility",
            "Clinical Procedures",
```

Figure 17. JSON file of the dependency graphs in Figure 15.

A second resource is ViViaN,[8] which visualizes VistA and its namespace. This tool uses the relationship files (in JSON format) that can be compiled from its repository.[9] By parsing these JSON files, it is

[8] https://code.osehra.org/vivian/

[9] https://github.com/OSEHRA/ViViaN

possible to find relationships between packages in the VistA system with which clinical users interact. A third resource is the set of VHA Business information models[10] that give insights into the relational database (CDW) fields. The team explored all these resources and parsed package relationships to gain insights and consider them for future use.

2.6 ADDITIONAL VISTA DATA

Required fields in VistA are extracted using the web crawler described in the previous sections as this information does not exist in CDW. Required fields are the fields in VistA that must be filled. By incorporating required fields into CDW environment, we were able to analyze the fill rate of those fields. The process is shown in Figure 18:

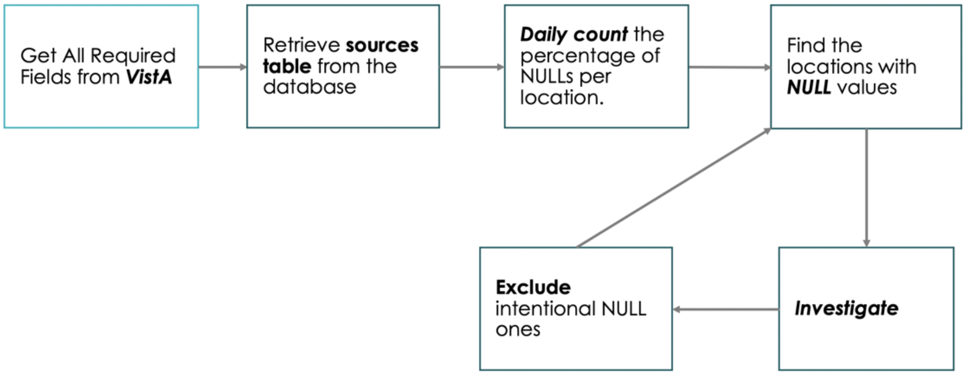

Get All Required Fields from VistA
Retrieve sources table from the database
Daily count the percentage of NULLs per location.
Find the locations with NULL values
Investigate
Exclude intentional NULL ones

Figure 18. Activity flow of required fields fill rate analysis.

Table 3 presents some sample required fields and their fill rates when we queried 1 month of records in early 2018. By definition, we expected to see 1.0 at fill rate for required fields in CDW; however, that was not the case. We concluded that some of the fields them are intentionally ignored to be transferred (such as RequestVistaErrorDate), or some could be overriding the requirement by leaving those fields empty (such as MaxRefills). A candidate detector could be designed around this finding since dropping fill rate could be indicative of VistA system level down times in different locations as well as misuse of HIT by some users. This analysis will be revisited in the future to see if we can explicitly design detectors to monitor certain required fields.

Table 3. Some required fields and their fill rates

Locations	Field/Column Name	Fill rate
129	CPRSStatus	1.0
129	RequestVistaErrorDate	0.000
3	ToRequestServiceName	~0.999
128	MaxRefills	~0.70–0.75, Max 0.89
4	FileEntryDateTimeTransformSID	~0.00002

[10] https://bim.osehra.org/content/_Z_.p-8.mL.oE.eG.uP.oQ.a3VW.pGA_root.html

2.7 FUTURE DIRECTION IN DATA MODELING

We have explored multiple directions in automation of the data extraction and preprocessing operations. For the initial analysis, preparing aggregate tables (counts data model) in a relational database was enough to design detectors. In the long run, we are aiming to design a data model that has more granularity with the option to aggregate if needed. We have researched various ideas, and among them we will start with exploration of time as a dimension because we think it can further aid in recognition and computation of anomalies. Currently, data is stored in a relational database with timestamps attached. However, these are neither used nor tracked for anomaly detection. Time series databases can track these changes, analyze changes in the past, monitor the present, and predict changes in the future [7, 8]. The shift in using time series databases for data analysis can be evidenced by the popularity in adoption of InfluxDB and Apache Druid databases. In fact, it is the fastest growing database category for the past two years [8, 9].

In this project, we will start with keeping the counts data in a big time series database that is indexed by timestamps and additional order features such as urgency, along with the existing ones in Table 1. We can keep multiple frequencies (i.e., discontinuations, creations, suspensions) in such a series. As the new data will be indexed by time stamps and the features, it will allow fast slice and dice analytics and grouping the counts by different features and minute level timescale.

Moreover, a time series database provides an opportunity to merge two data models we have generated (event sequence and counts) and use one single model. Event sequences can also be stored in time series format before the OASIS mapping is performed.

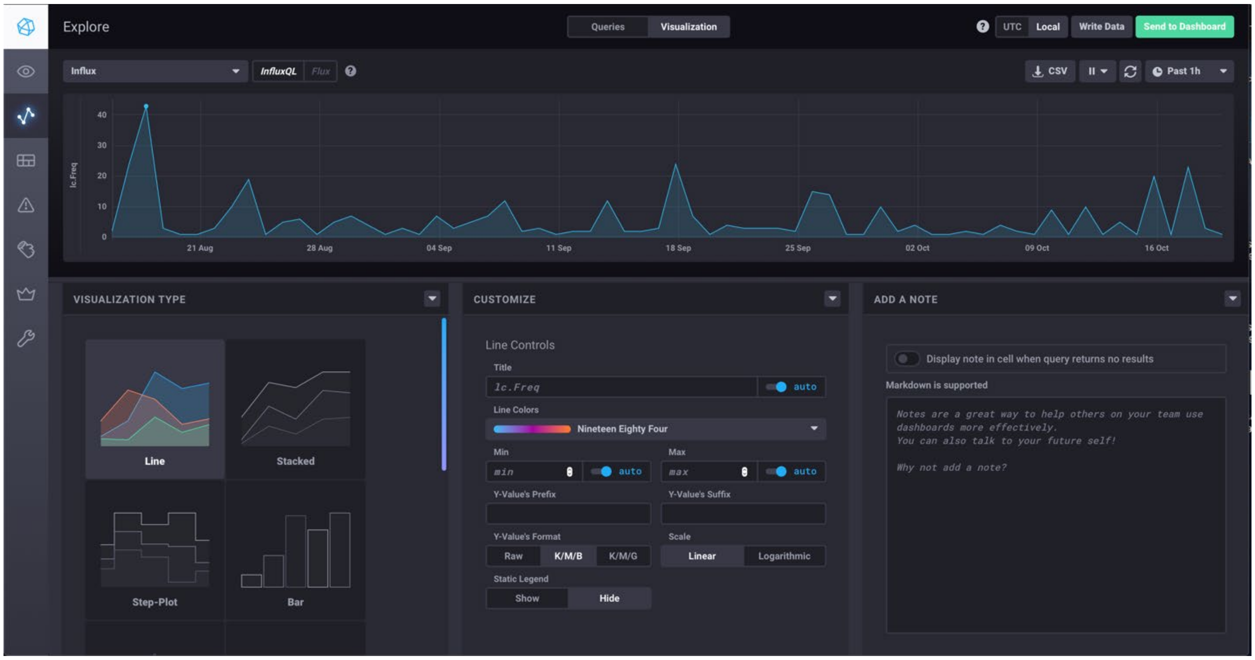
We recently investigated various time series databases—InfluxDB, Prometheus, Graphite, Apache Druid, and TimescaleDB. Based on comparisons between the features, disadvantages, applying machine learning algorithms, and integration of the database within the current architecture, InfluxDB[11] provides notable benefits. Built with the notion of high-performance to support time-stamped data specifically, InfluxDB optimizes the data in a format that leverages the best of the both worlds—structured schema found in relational databases and key-value pairs at the heart of NoSQL solutions. This provides support for fast querying on "Big Data" at the granular or aggregate level [10, 11]. InfluxDB provides rich features like SQL-like queries, GroupBy, and TopN functionalities including the ability to identify data gaps and in-built aggregation such as percentile and std. Multiple dimensions coupled with time can be used in detecting anomalies. It supports one million+ writes per second, which many other competitive solutions are unable to match [10].

Two notable technologies that come along InfluxDB are Chronograf and Kapacitor. The former supports ad hoc exploration and querying (including SQL-like query language) and interactive data visualization capabilities on time series data, as illustrated in Figure 19. The data used in Figure 19 are the laboratory counts data from CDW. Chronograf can be useful for querying in a web interface and for instant visualization purposes after the detection is done.

The latter, Kapacitor is a data processing engine that supports integration with external anomaly detection and machine learning libraries via open application programming interfaces (APIs) or plugins. These technologies would allow us to use our in-house anomaly detection libraries to be applied on the timeseries-based count and/or event-sequence data, while leveraging various features including fast slice and dice analysis on multidimensions, compare patterns, ad hoc exploration, and interactive data

[11] https://influxdb-python.readthedocs.io/en/latest/include-readme.html

visualization. We plan to apply these database technologies for use in the previous data models and evaluate their performance in terms of aiding anomaly detection and scalability in the next term.



Line Controls

Title: lc.Freq ● auto

Line Colors: Nineteen Eighty Four

Min: min ● auto

Max: max ● auto

Y-Value's Prefix

Y-Value's Suffix

Y-Value's Format: Raw [✓ K/M/B] K/M/G

Scale: [✓ Linear] Logarithmic

Static Legend: Show [✓ Hide]

○ Display note in cell when query returns no results

Markdown is supported

Notes are a great way to help others on your team use dashboards more effectively.
You can also talk to your future self!

Why not add a note?

Figure 19. Chronograf interface visualizing the number of cancelled orders for a location.

3. HAZARD ANALYSIS AND CHARACTERIZATION

The hazard analysis and characterization task consists of three subtasks: probabilistic model development, event-based hazards detection, and system-based reliability performance evaluation.

3.1 PROBABILISTIC MODEL DEVELOPMENT

In this section, we describe a software conceptual model, using the graphical model, that captures the transaction process model (TPM) for the Consult domain to perform statistical analysis. A key aspect of this study is the determination of the probability that an error or class of errors associated with specific data items and process steps will result in process failure such as an incorrect medical diagnosis and/or treatment. Hence, we propose conducting a probabilistic study based on Monte Carlo simulation to an information flow graph to identify paths and path segments that contribute to process error. Error probabilities or uncertainty levels are induced at each decision step of the graphical model based on discussions with VA experts and clinical partners. The model was calibrated using a finite set of data fields (from notional data) and initialized. This conceptual model serves as a basis to generate the statistical features of the errors, sequence of errors (e.g., unmatched event time in the EHR systems), frequency of errors, and correlation between a variety of steps in relationship with errors. Key statistical metrics include quantification of frequency of errors and correlation of error probabilities across the workflow. The objective of the model is to enable the design of error detection algorithms and quantify the efficacy of error detection algorithms. A blueprint for model development and a tool for probabilistic evaluation of decision-making will, thus, be developed to enable hazard detection methods.

3.1.1 Model Description

In a given node-edge graphical model, for each of the data fields, there are four possible outcomes (O = 4): (1) data is entered correctly in the data field, (2) data is entered incorrectly in the data field, (3) no data is entered correctly, and (4) no data is entered incorrectly in the data field. Assume that each outcome has a known probability of occurring first. All the four outcomes can occur, but the order is unknown. There is, thus, one starting point and O! ending points (! represents the factorial operation). To know the sequence of events and probability associated with each endpoint, we conducted 1,000,000 Monte Carlo simulation runs to estimate the histogram, the probability density function, and the cumulative density function for the outcomes.

For the initial implementation, we converted the high-level Consult TPM shown in Figure 20 into an information flow graph shown in Figure 21, which can be simplified as shown in Figure 22.

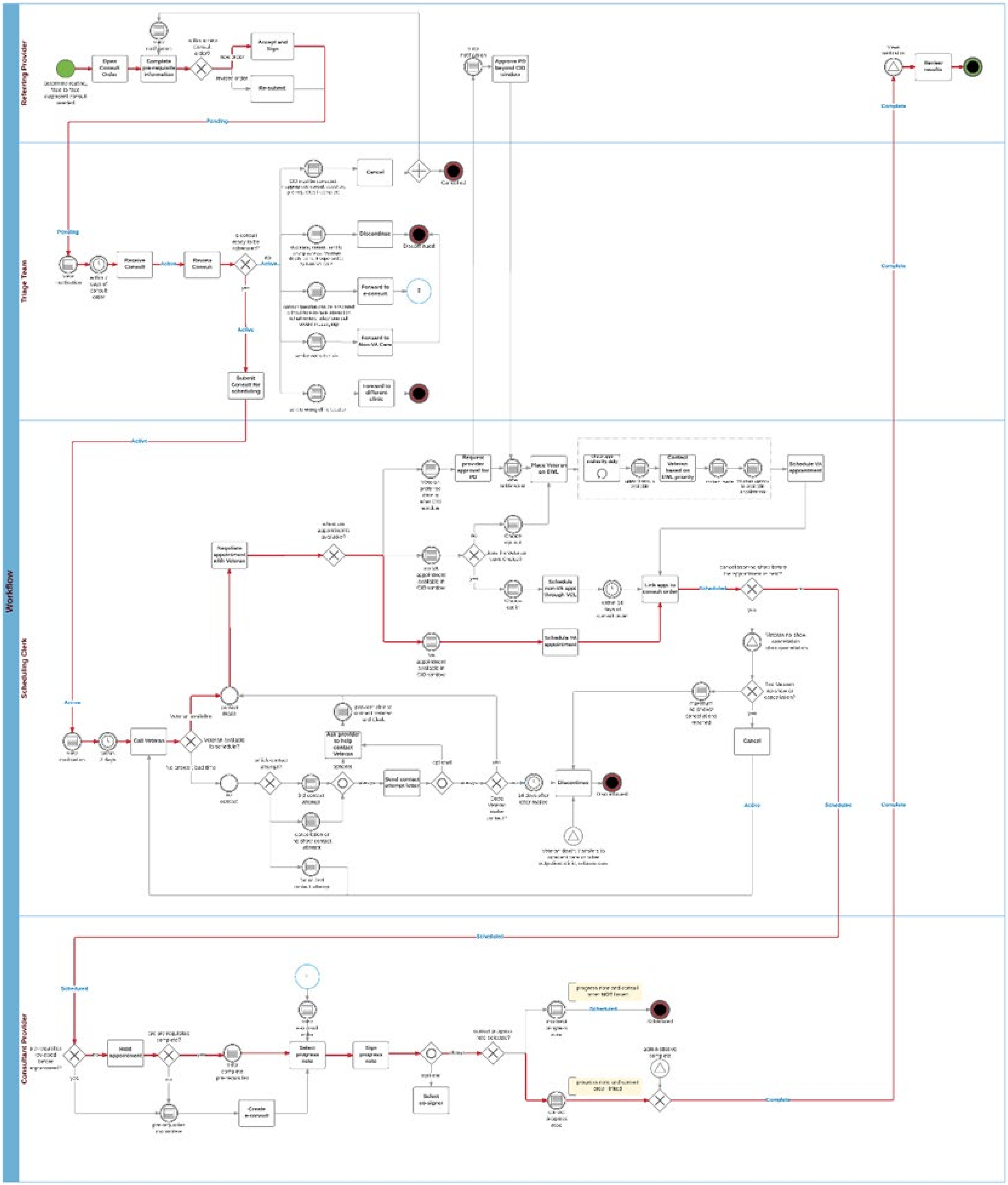


Figure 20. A high-level Consult TPM.

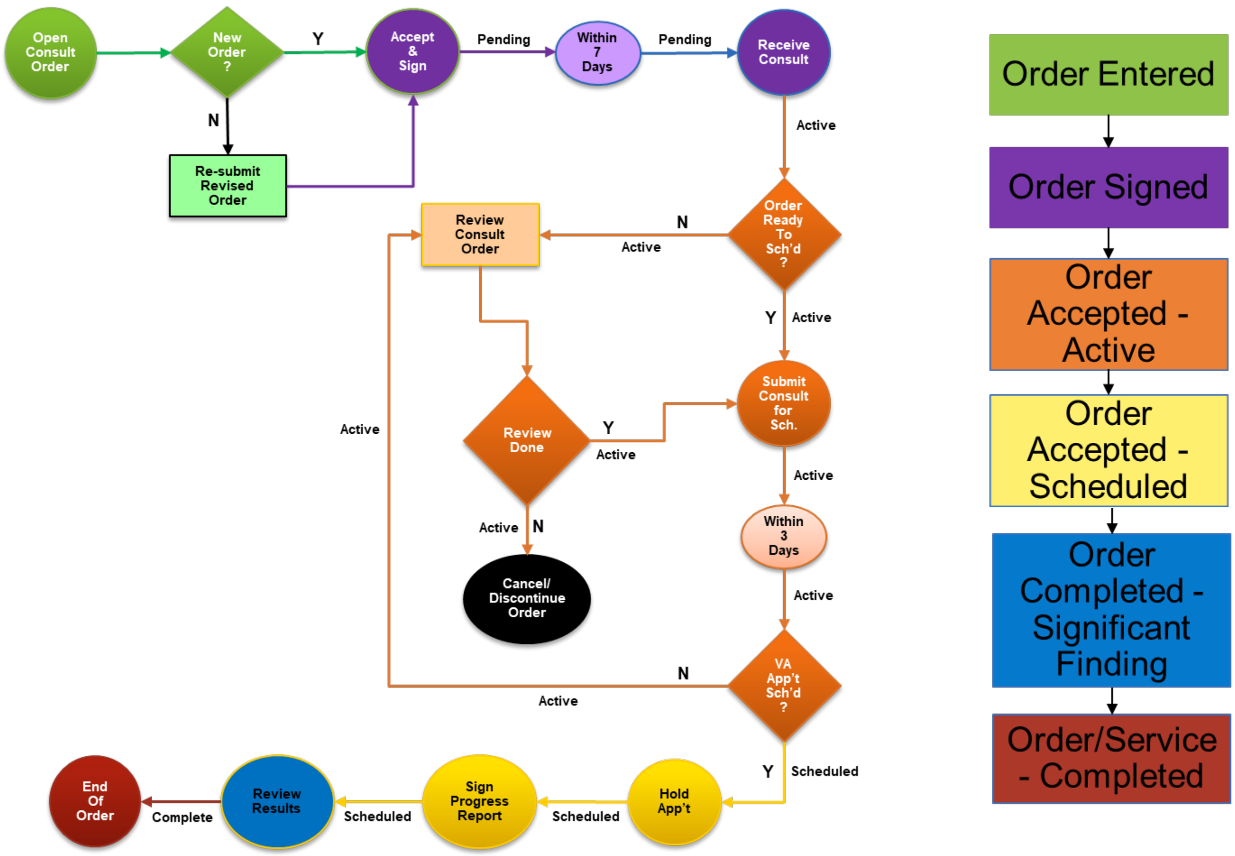


Figure 21. An equivalent information flow graph for the high-level Consult TPM shown in Figure 20.

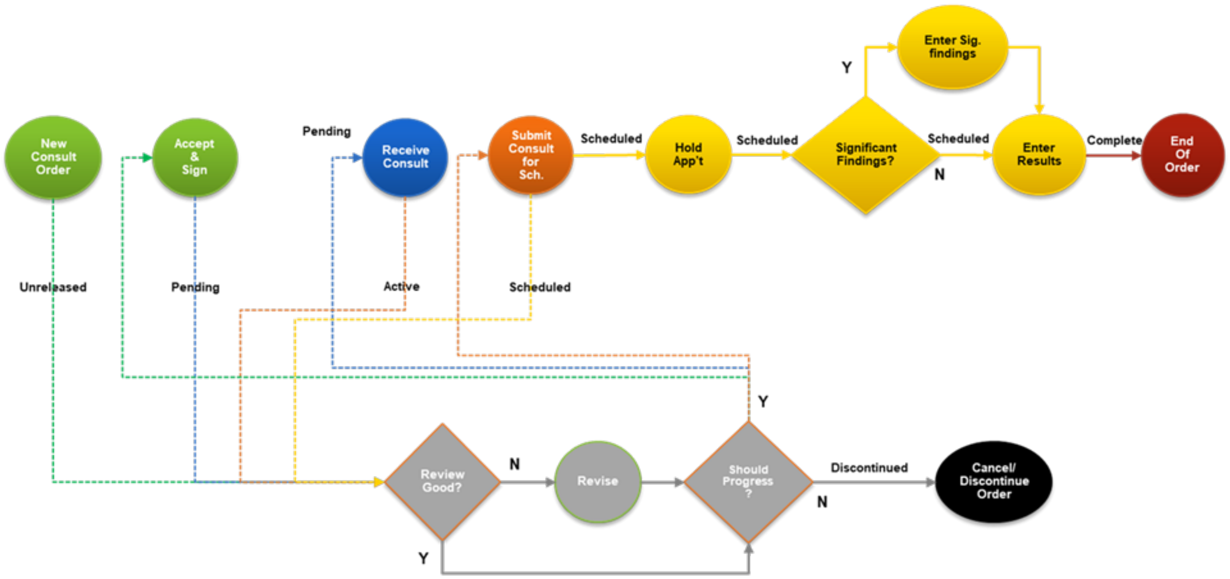


Figure 22. A simplified version of the information flow shown in Figure 21.

In Figure 21 and Figure 22, an information flow is represented as nodes and links. Each node represents a state in TPM, and each link represents the relationship between nodes. On the one hand, nodes are independent if the outcomes of one node has no effect on the outcomes of the downstream nodes. On the other hand, nodes are dependent if the outcomes of an upstream node have an effect on the outcomes of a downstream node.

3.1.2 Scenario Simulation

To simulate a hypothetical scenario, we can model the TPM in Figure 21 as shown in Figure 23.

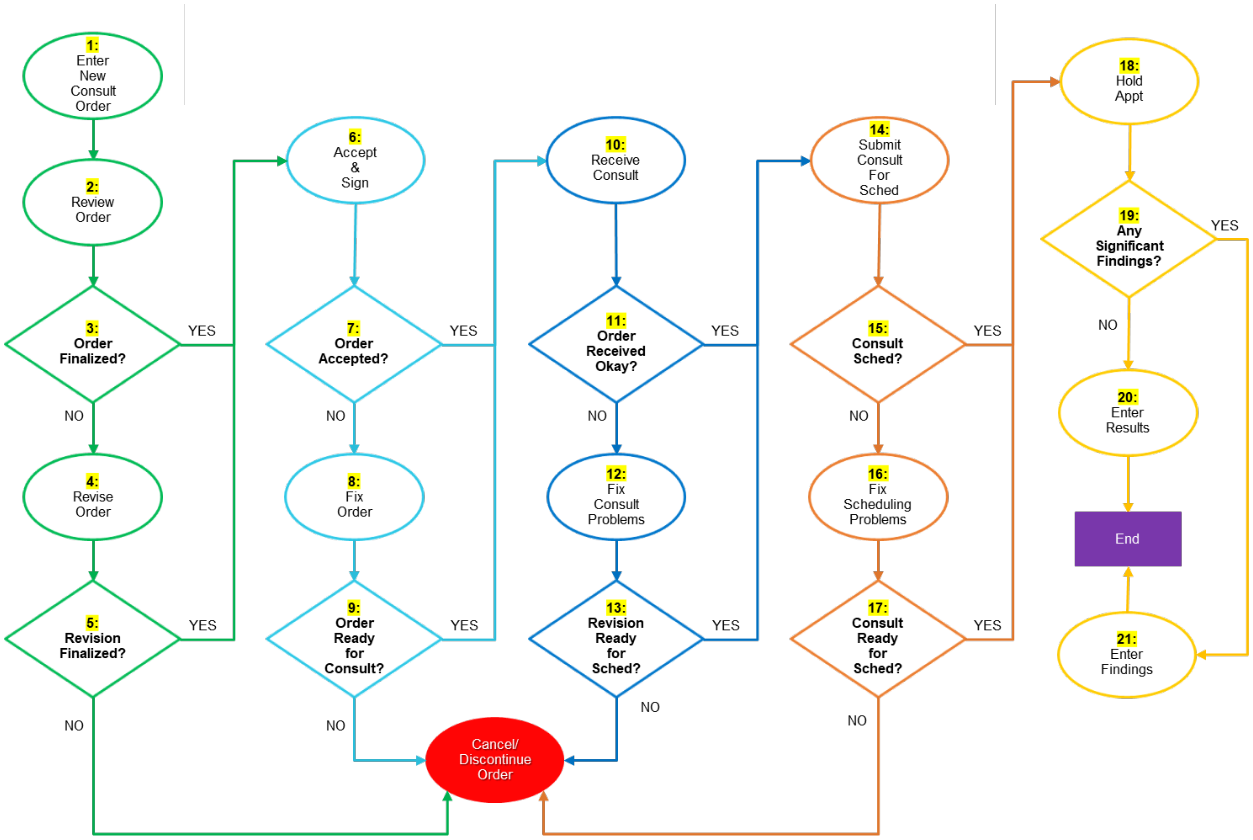


Figure 23. The information flow used to model the hypothetical Consult activity flow.

In this scenario, we assume that the probabilities of data entry errors at each node is 10% (5% error when there is data and 5% error when there is no data). Note that the data entry error rate in EHRs range from 2% to 6% [12–16]. We also assume equal split probabilities whenever there is a decision node with multiple branches (50:50 split ratio for a decision node with two branches going forward). Then, 1,000,000 Monte Carlo simulations are conducted on the information flow graph to estimate the probability of failure at each node. In this analysis, we assume all nodes are independent (i.e., an error in a node will not affect errors in other nodes). Figure 24 illustrates the estimated probability density functions (PDFs) for all nodes in the “New Order Process” subprocess of the high-level Consult process. We observe that the probability of error (data entered incorrectly) decreases as data flows down the tree.

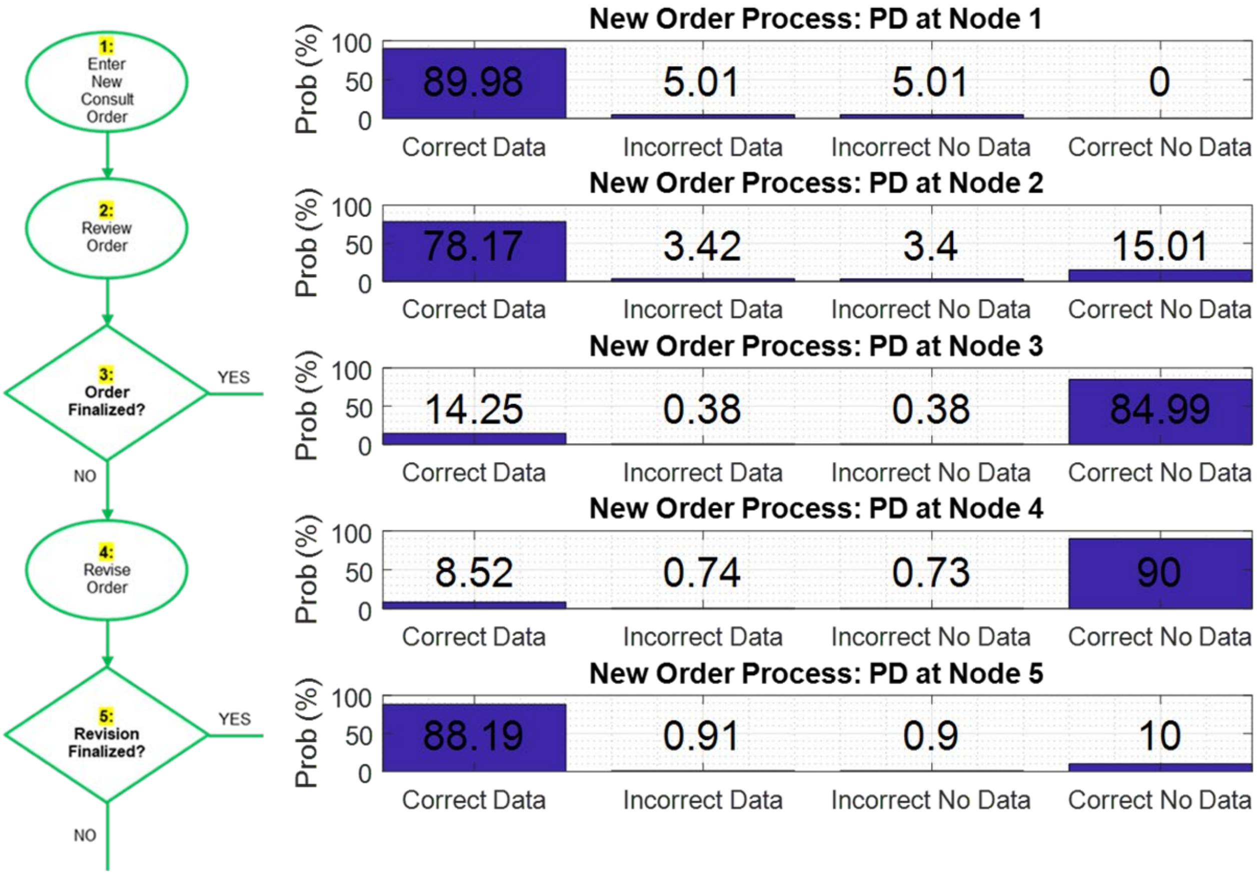


Figure 24. Estimated PDFs for the *New Order Subprocess*.

Based on this data entry error assumption, the PDFs for the remaining part of the information graph are shown in Figure 25, Figure 26, Figure 27, and Figure 28; these are equivalent to the remaining four processes in the high-level Consult process.

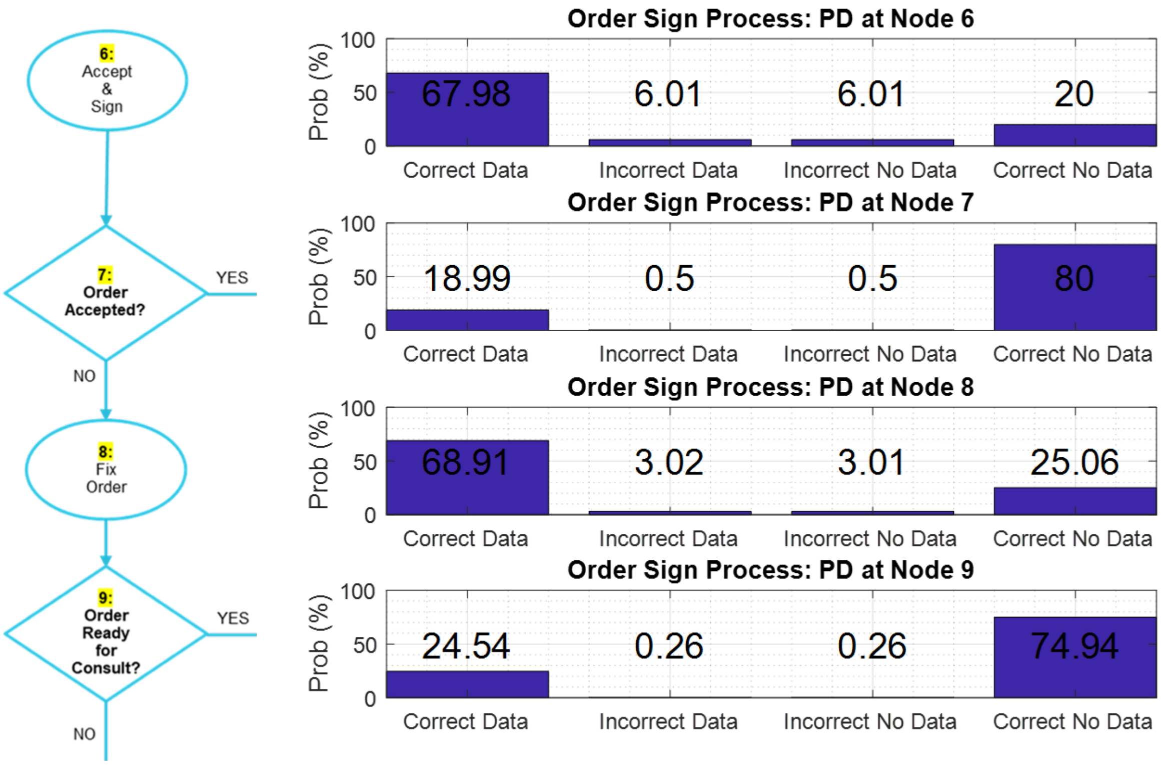


Figure 25. Estimated PDFs for the *Order Sign Subprocess*.

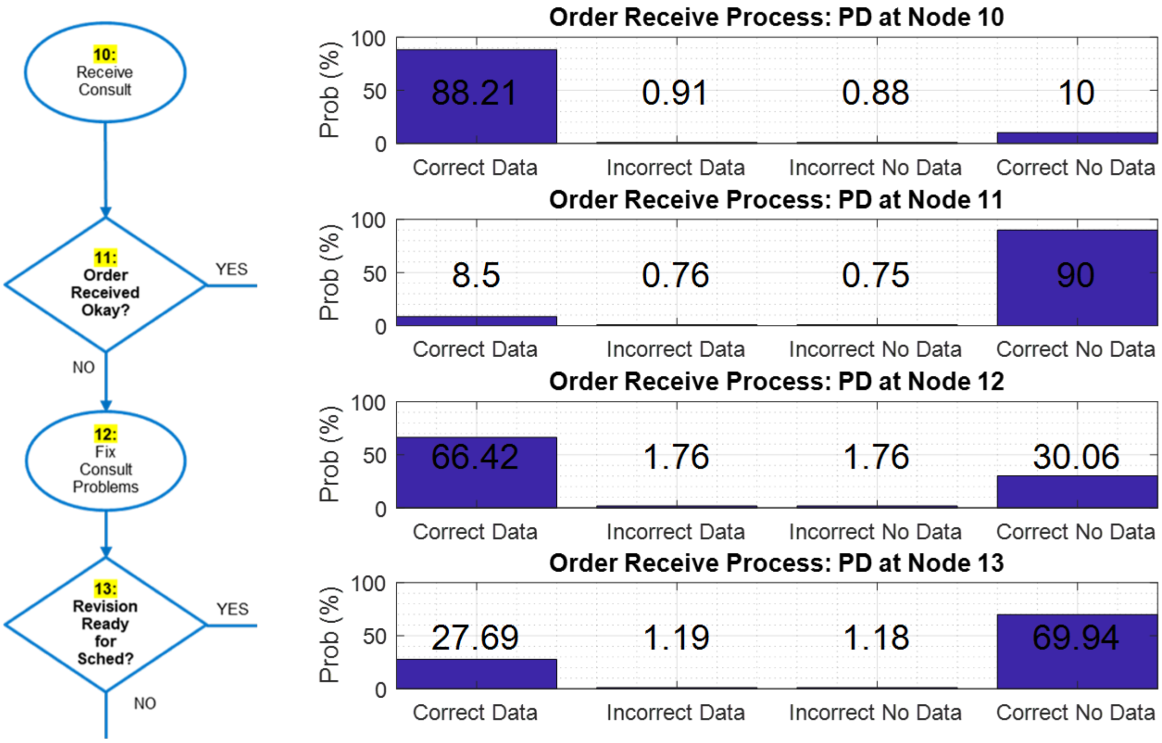


Figure 26. Estimated PDFs for the *Order Receive Subprocess*.

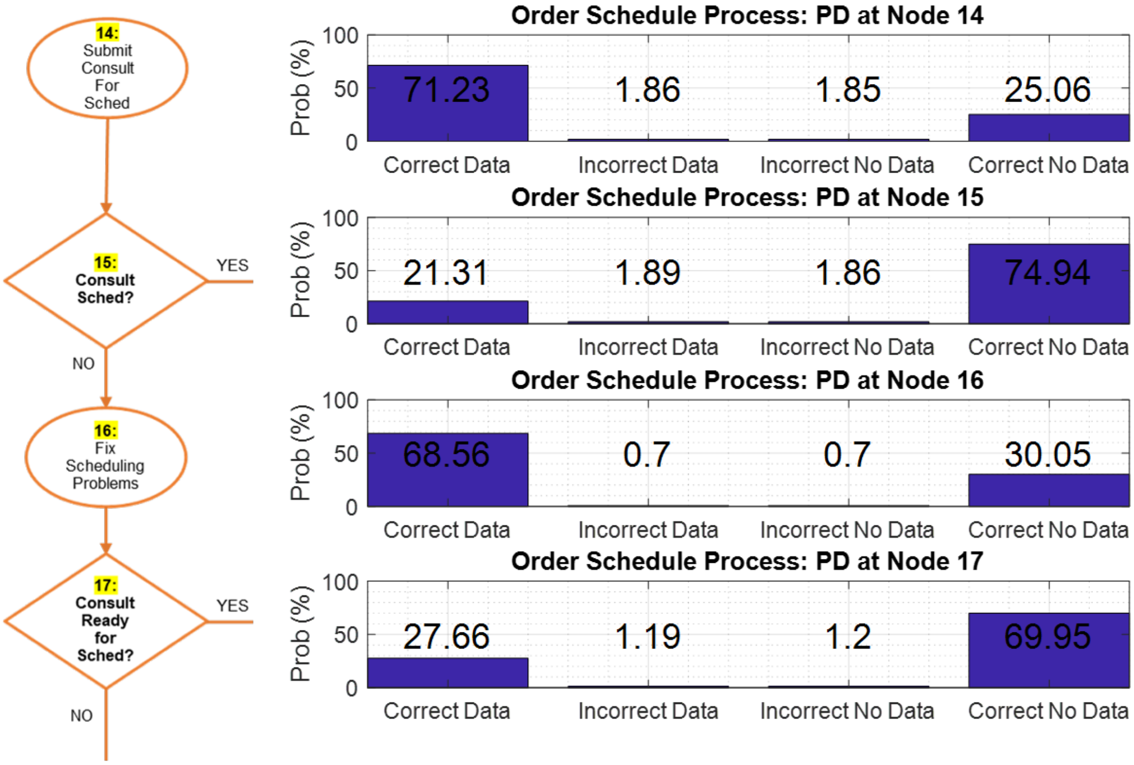


Figure 27. Estimated PDFs for the *Order Schedule Subprocess*.

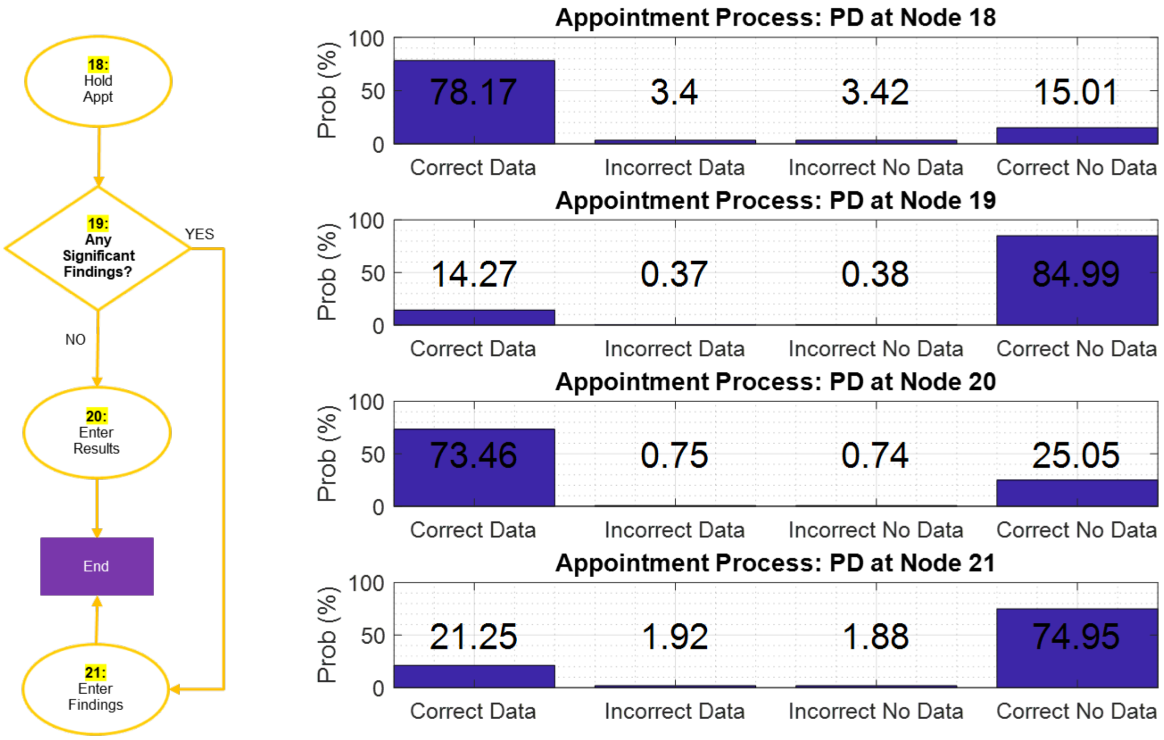


Figure 28. Estimated PDFs for the *Appointment Subprocess*.

Based on the assumptions in this scenario, we can conclude that the probability that an error will occur at the end of the Consult process is about 3.8% (1.92% when there is data and 1.88% where there is no data). As expected, the probability of error decreases over time; however, it is still a high probability for large-scale EHR systems.

3.2 EVENT-BASED HAZARD DETECTION APPROACHES

This subtask uses data-driven approaches to identify potential hazards for further investigation. The task assumes that hazards are not known a priori. This assumption, thus, not only provides an opportunity to discover previously unknown or undetected hazards but also allows for analysts to remain in the loop. The analysts can review the detected hazards and determine whether they are true hazards or just noise in the data. The analysts' reviews will serve as feedback to the hazard detection methods so that future occurrences are characterized correctly. This is a dynamic approach in which the detectors are continuously trained and improved using the analysts' feedback as additional data points. This approach is analogous to the concept of incremental learning in machine learning literature. Using this approach, we developed a series of hazard detectors (HDs) as presented in the following subsections. In all cases presented in this report, we define *anomaly as a point in time where the behavior of the observed system is significantly different from previously observed behaviors*.

3.2.1 Online Change Point Detection Approach

One of the limitations of the SPC approach discussed in Section 2 is that it is not amenable to online detection when new data points become available. The online change point detection (CPD) approach discussed in this section was developed to eliminate this challenge.

3.2.1.1 Introduction

CPD is finding changes in the underlying distribution of a time series. Basically, CPD methods measure the discrepancy in the distribution of data points in the immediate past (called the embedding) window and the immediate present (horizon) window. The two windows may or may not overlap each other, as shown in Figure 29.

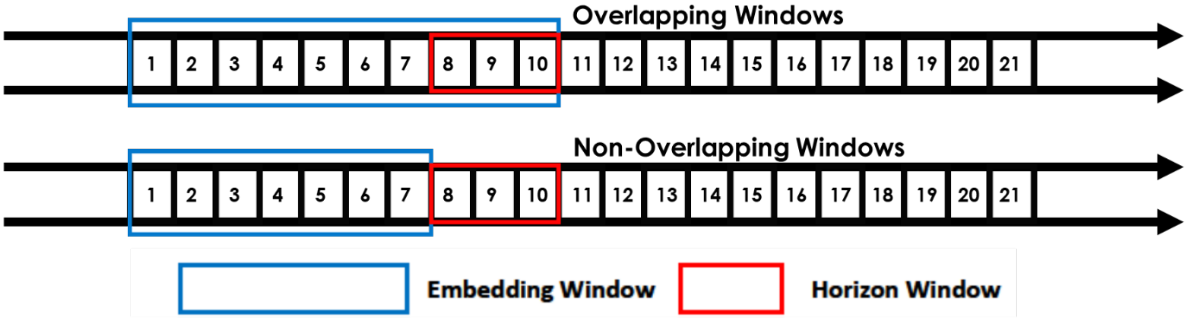


Figure 29. Relationship between embedding and horizon windows in the implementation of CPD methods.

According to [17], CPD methods are divided into two main groups: *online* methods, which aim to detect changes as soon as they occur in a real-time setting, and *offline* methods, which retrospectively detect changes when all samples are collected. Online methods are often referred to as event or anomaly detection, while offline methods are called signal segmentation. A detailed survey of offline algorithms for the detection of multiple change points in time series is presented in [17]. Most of the presented offline algorithms require a known number of change points to solve the discrete optimization problem. This is a challenge for applications with no known number of change points.

3.2.1.2 Bayesian Change Point Detection Algorithm for Breakpoints

To eliminate this challenge, a Bayesian CPD (BCPD) algorithm is presented in [18]. In their study, change point is defined as an identification of abrupt changes in the generative parameters of sequential data. Specifically, the algorithm assumes that a sequence of observations $x_1, x_2, \ldots, x_T$ may be divided into nonoverlapping partitions. The delineations between these partitions are called the change points. The algorithm further assumes that the data within each partition are independent and identically distributed from some probability distribution. The goal of this algorithm is to estimate the posterior distribution since the last change point given the data so far observed. See [18–20] for detailed information about this algorithm.

The BCPD algorithm can be implemented for a batch dataset in an offline approach as well as for a real-time dataset in an online approach. The results of some implementations of the algorithm using offline and online approaches on synthetic datasets are shown in Figure 30 and Figure 31. In both cases, the online implementation approach gives a higher probability of detection compared with its offline counterpart. In Figure 30, the six partitions were correctly detected with a probability of more than 0.8. For the offline implementation, only one of the partitions was detected with a probability of more than

0.5. Similar conclusions can be reached for the results shown in Figure 31, even though the partitions in the datasets have different characteristics.

Another conclusion from this experiment is that this algorithm is not appropriate for detecting between breakpoints; that is, this algorithm detects the start and the end of a breakpoint, but it does not detect points within a breakpoint. In domains such as health care, it is not sufficient to detect only the start and the end of a breakpoint and not the values between breakpoints that exhibit abrupt changes. Consequently, we proposed a modification to the BCPD algorithm to handle this problem.

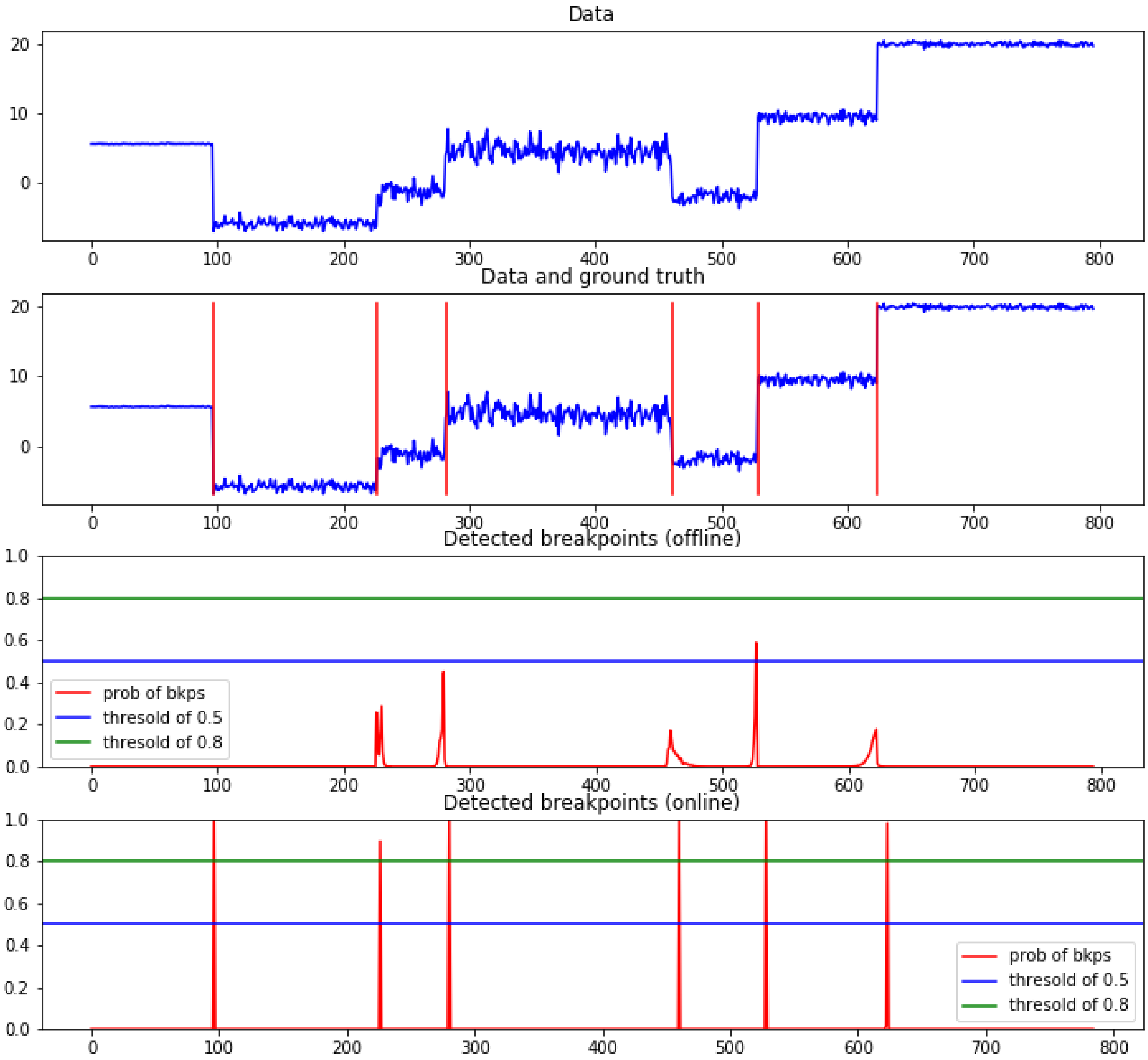


Figure 30. Implementation of BCPD algorithm on synthetic dataset 1.

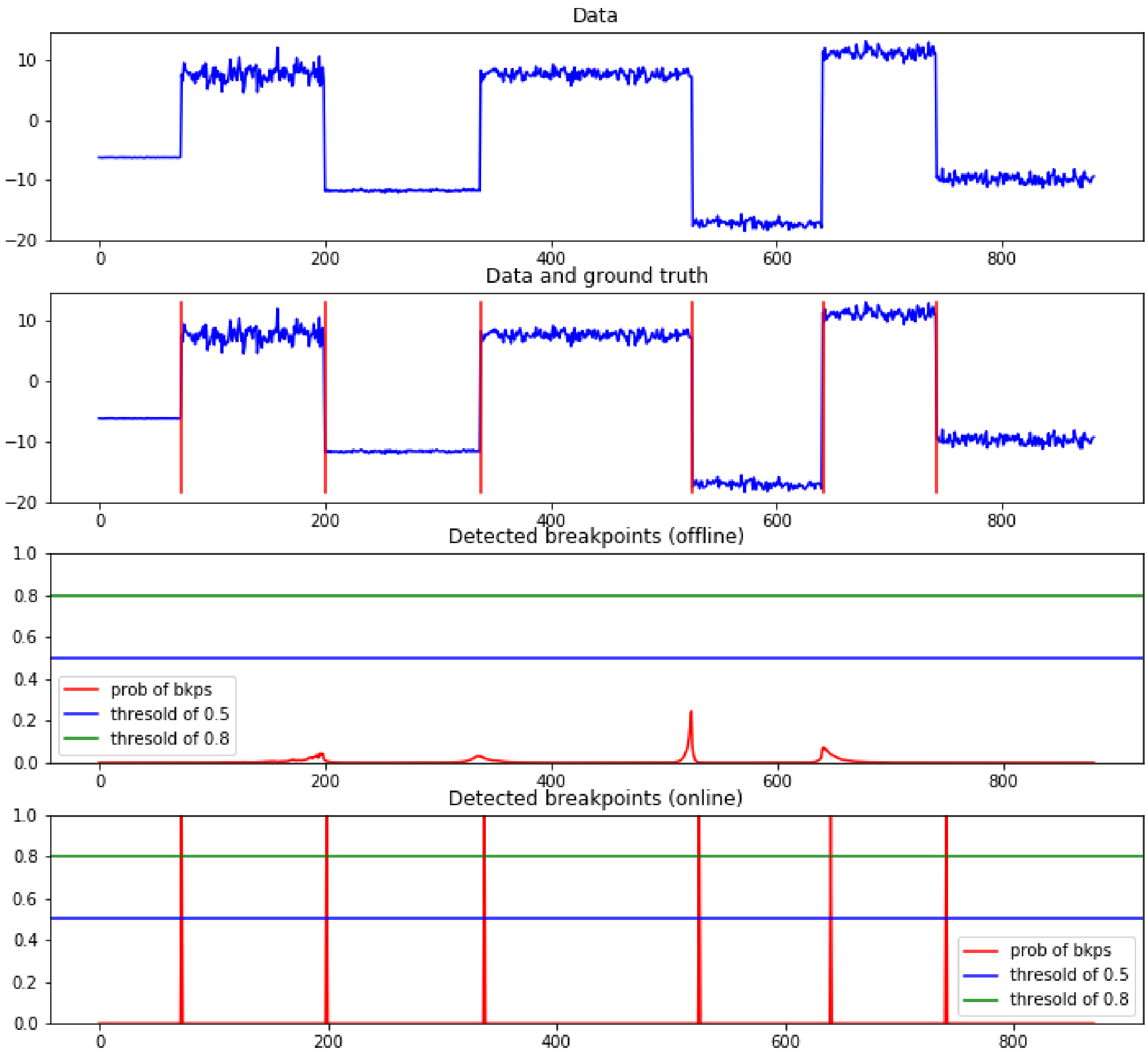


Figure 31. Implementation of BCPD algorithm on synthetic dataset 2.

3.2.1.3 Bayesian Online Change Point Detection Algorithm for Change Points within Breakpoints

We consider the case shown in Figure 32 in which there are two major breakpoints identified by red lines. These two breakpoints are easily detected by the BCPD algorithm. In addition to these two points, however, we also would like to also detect all the points within these two boundary points. Hence, we propose a Bayesian Online Change Point Detection (BOCPD) algorithm. For our approach, we focus on using overlapping windows as depicted in Figure 33.

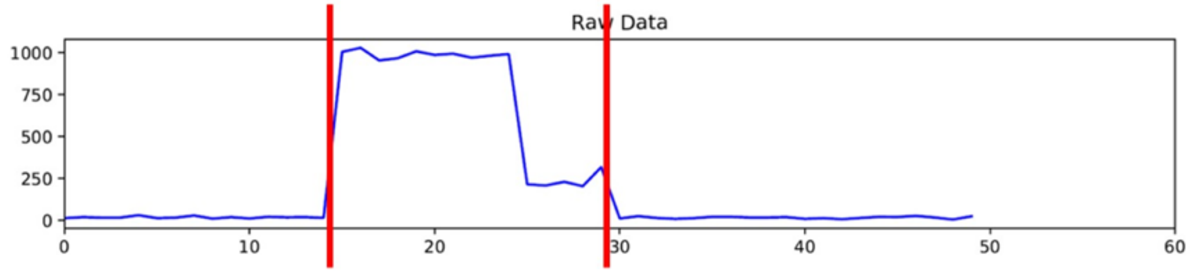


Figure 32. Illustration of detecting change points within breakpoints.

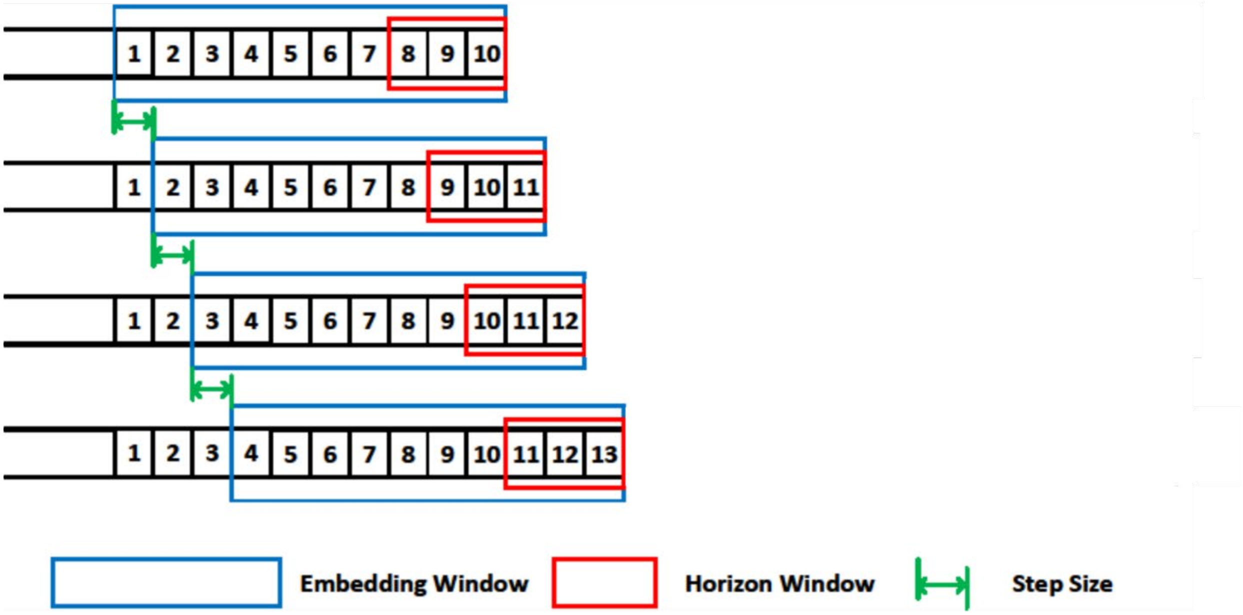


Figure 33. Procedure for implementing the overlapping windows for the BOCPD algorithm.

An illustration of this procedure is shown in Figure 34. In this case, the raw data (top chart) is presented to the algorithm as one data point at a time (middle chart), and the output of the CPD is shown as a probability of detection (bottom chart).

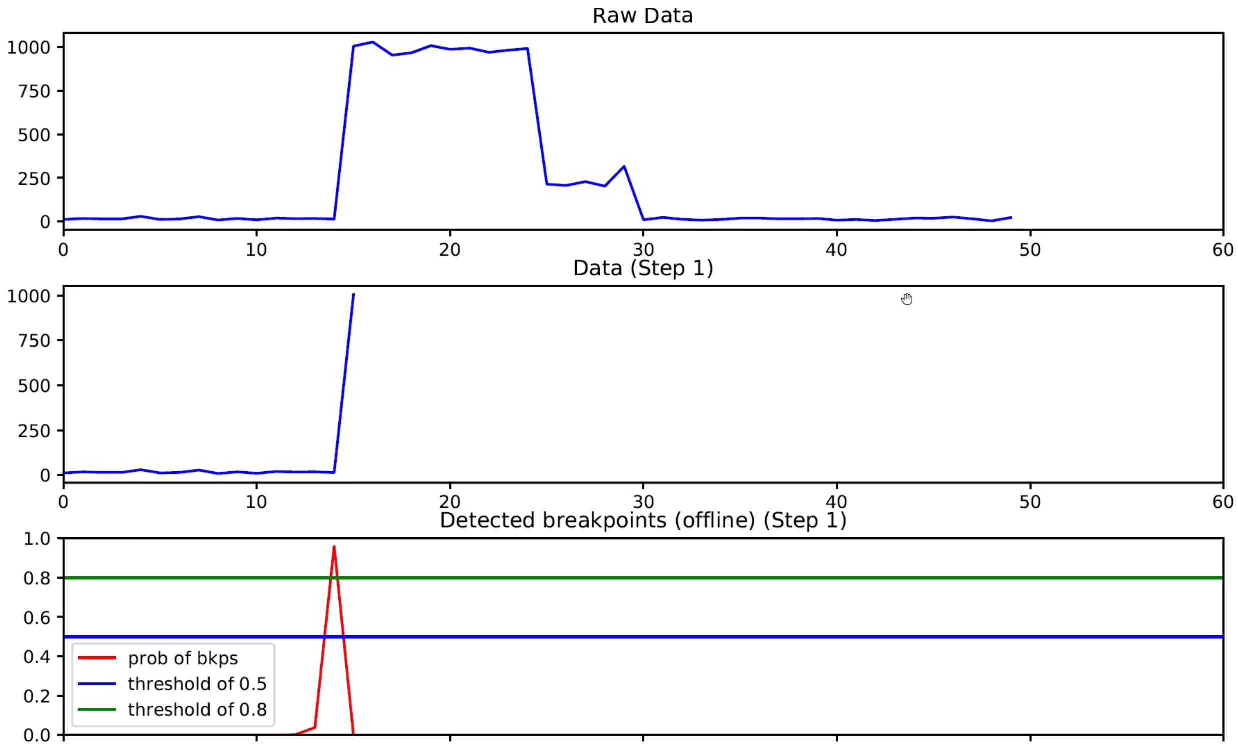


Figure 34. Illustration of the results of the overlapping window for the BOCPD algorithm.

Using this procedure, all the breakpoints are easily detected. However, after a data point is detected as an anomaly, a replacement strategy is also implemented in order not to bias the subsequent detection process. The replacement approach is, thus, that a detected data point is replaced with the average of the previous l points, where l is the length of the embedding window. By replacing the anomalous value with a value characteristic of the prior nonanomalous data points, we maintain a reasonable level of accuracy for future detections.

The same procedure was applied to Order Rejection count data, and the results are shown in Figure 35. The detected breakpoints (or anomalous count data points) are identified with a red circle. Overall, all the counts over about 150 are detected as breakpoints.

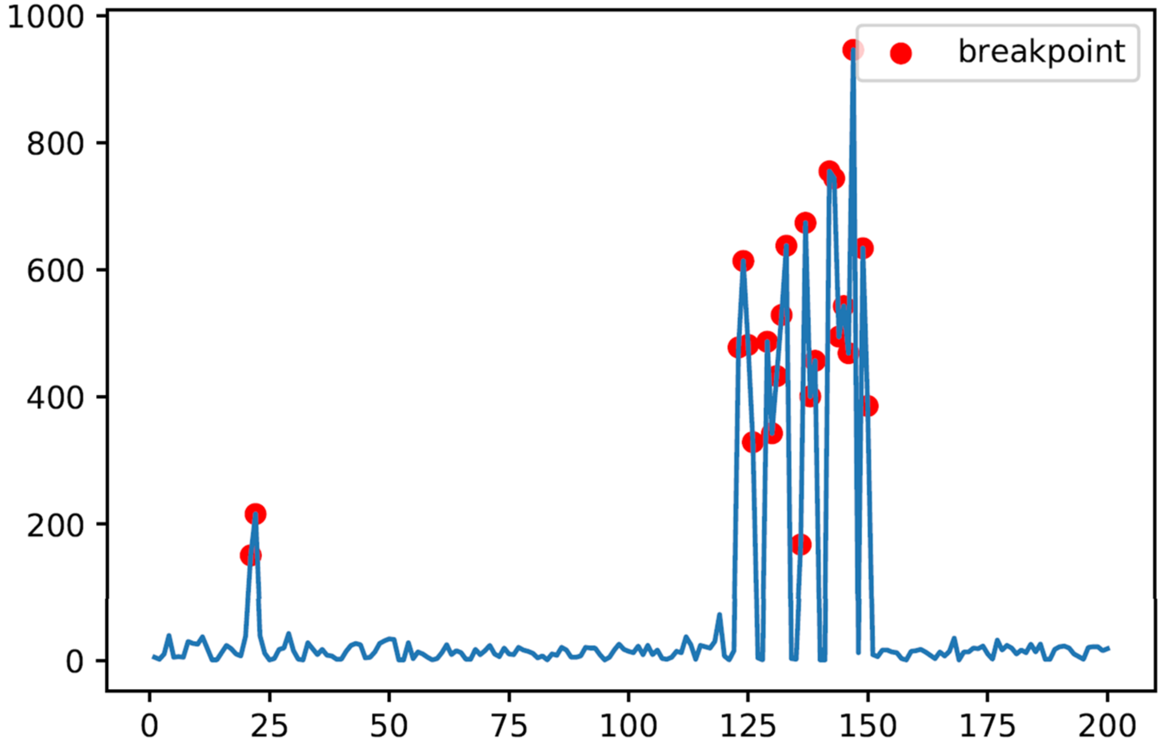


Figure 35. Results of the BOCPD algorithm for order rejection count data.

3.2.2 Real-Time Anomaly Detection Using Forecasting Approach

The two previous approaches, SPC and OCPD, do not allow for forecasting the future value of the monitored system. In this section, we present a third approach that will add that capability to the HD tool suite.

3.2.2.1 Introduction

The proposed real-time anomaly detection using the forecasting approach starts with building a forecasting model that can be used to predict future values in the series. Because of variability in the modeled system, in most cases, a given predicted value will differ from the corresponding observed value. If this difference for a given data point is above or below a prespecified threshold, the data point is classified as an anomaly. Thus, the challenge in anomaly detection for time series data lies not only in developing an accurate model to predict future values but also in establishing robust thresholds for determining whether points are anomalous [21]. Although many methods exist for building time series forecasts, we use a machine learning technique called online support vector regression [22]. The need for using advanced machine learning techniques on EHR data has been suggested [23]. Similarly, various approaches have been employed for producing error thresholds (i.e., confidence intervals) for predicted values. However, we use the concept of robust confidence intervals as outlined in [21] for our approach.

Support vector machineis an unsupervised learning technique used in classification problems. When support vector machine techniques are applied to regression problems, the algorithm is called support vector regression (SVR) [24]. In conventional batch implementations of SVR, every time the training set is modified, the model must be retrained from scratch [22]. This is unsuitable and computationally inefficient for EHR time series data in which additional points are regularly added to the series in real time. To address this inefficiency, [22] introduced an "Accurate Online Support Vector Regression" (AOSVR) algorithm. With AOSVR, every time a data point is added to the training set, the trained SVR function is efficiently updated rather than being retrained from scratch [22]. Thus, AOSVR in combination with an anomaly detection approach provides an avenue for HIT safety managers to make real-time decisions.

3.2.2.2 Mathematical Overview of SVR

In nonparametric regression, the response variable *Y* is given by

$$Y = \mu_Y\left(\mathbf{x}\right) + \varepsilon\left(\mathbf{x}\right),$$

where $\mu_Y\left(\mathbf{x}\right)$ is the expected value of *Y* and $\varepsilon\left(\mathbf{x}\right)$ is a random error term [25]. The purpose of SVR is to estimate $\mu_Y\left(\mathbf{x}\right)$ using a known dataset *T* called a training set. Let $I = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m\}$ be the set of training inputs and $O = \{y_1, y_2, ..., y_m\}$ be the corresponding set of outputs. The training set is then given by $T = \left\{\left(\mathbf{x}_1, y_1\right), \ . \ . \ . \ , \left(\mathbf{x}_m, y_m\right)\right\}$. From the training set, construct a linear regression function of the form

$$f\left(\mathbf{x}\right) = \mathbf{w}^T \varphi\left(\mathbf{x}\right) + b$$

on a feature space **F**. Here, **w** is a vector in **F** called the weight vector, *b* is the bias term, and $\varphi\left(\mathbf{x}\right)$ maps **x** to a vector in the higher dimensional feature space F. This mapping accounts for nonlinearities between the input vector and the response variable. The weight vector **w** and the bias *b* are obtained by solving the following primal optimization problem:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\mathbf{w}^{\mathbf{T}}\mathbf{w} + C\sum_{i=1}^{m}\left(\xi_i^+ + \xi_i^-\right) \\ \text{subject to:} & \begin{cases} y_i - \mathbf{w}^{\mathbf{T}}\varphi(\mathbf{x}_i) - b \le \varepsilon + \xi_i^+ \\ \mathbf{w}^{\mathbf{T}}\varphi(\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^- \\ \xi_i^+, \xi_i^- \qquad\qquad \ge 0, \end{cases} \end{array} \tag{1.1}$$

where *C* is a positive regularization constant that "penalizes" y-values that differ from *f*(*x*) by more than *ε*. The slack variables, ξ_i^+ and ξ_i^-, correspond to the size of this difference for upper and lower deviations, respectively. Data points falling outside of the *ε*-tube (Figure 36) are defined by Vapnik's *ε*-insensitive loss function to have no contribution to the regression model (their coefficients equal zero) [26]. Points lying outside or on the *ε*-tube (called support vectors) have nonzero coefficients and thus contribute to the regression function.

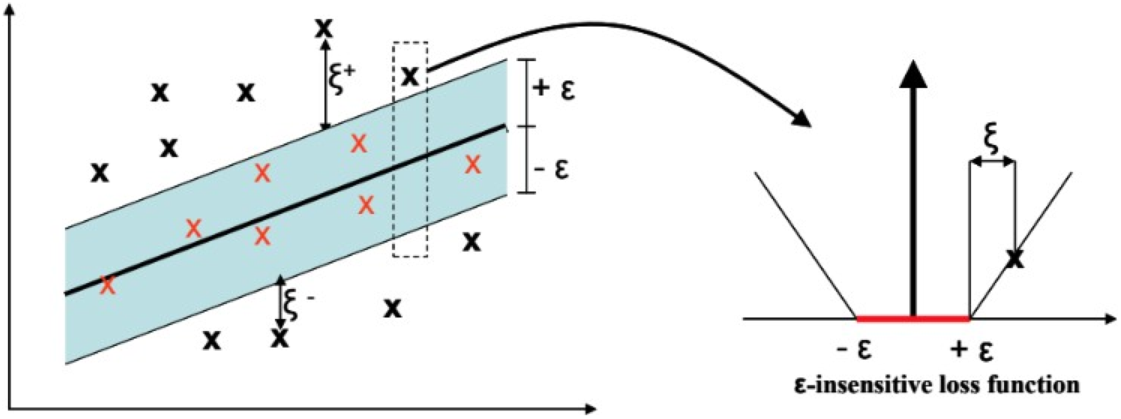


Figure 36. Illustration of ε-tube and ε-insensitive loss function (adopted from [14]).

Formally, the ε-intensive loss function $|\xi|_\in$ is described by

$$|\xi|_\varepsilon = \begin{cases} 0, & \text{if } |\xi| \le \varepsilon \\ |\xi| - \varepsilon, & \text{otherwise.} \end{cases} \tag{1.2}$$

The dual formulation of the primal optimization problem is crucial for extending SVR to nonlinear functions [24]. Once the related Lagrangian function is obtained and the Karush-Kuhn-Tucker conditions are applied, we obtain the dual formulation

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\sum_{i,j=1}^{m} K(\mathbf{x}_i, \mathbf{x}_j)(\alpha_i^+ - \alpha_i^-)(\alpha_j^+ - \alpha_j^-) \\ & + \varepsilon\sum_{i=1}^{m}(\alpha_i^+ + \alpha_i^-) - \sum_{i=1}^{m} y_i(\alpha_i^+ - \alpha_i^-) \\ \text{subject to} \quad & \begin{cases} \sum_{i=1}^{m}(\alpha_i^+ - \alpha_i^-) & = 0 \\ \alpha_i^+, \alpha_i^- & \in [0, C], \end{cases} \end{aligned} \tag{1.3}$$

where α_i^+ and α_i^- are Lagrange multipliers. See [22], [24], and [26] for more detailed mathematics related to the Lagrangian function and dual formulation. Finally, resolving the dual problem gives the adjusted regression function

$$f(x) = \sum_{i=1}^{m}(\alpha_i^+ - \alpha_i^-) K(\mathbf{x}_i, \mathbf{x}_j) + b. \tag{1.4}$$

Here, $K(\mathbf{x}_i, \mathbf{x}_j)$ is known as the kernel function. The kernel function allows nonlinear function approximations to be made with SVR but maintains the computational efficiency present when making linear approximations [22, 24] There are many common kernel functions including linear, polynomial, and radial basis kernels. Our current analysis uses radial basis kernels.

3.2.2.3 Data Normalization

For the purpose of testing our approach on existing data, the time series data is split into training, validation, and test sets. Note that although the data used is preexisting, the SVR model receives inputs one point at a time as though the operation was occurring online (that is, in real time). Before the data is put into the AOSVR algorithm, it is normalized between 0 and 1 using the following equation:

$$X_{norm} = \frac{X - X_{\min Train}}{X_{\max Train} - X_{\min Train}}, \tag{1.5}$$

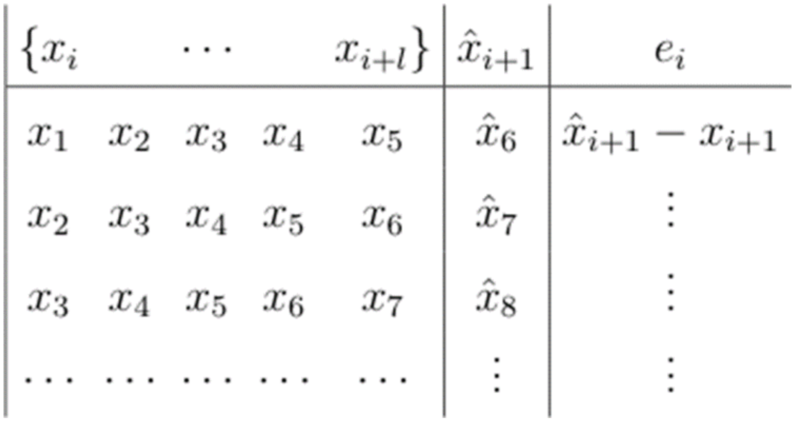
where $X_{\max Train}$ and $X_{\min Train}$ are the maximum and minimum values, respectively, from the training portion of the data [21].

3.2.2.4 Robust Confidence Intervals

Many of the existing methods for computing confidence intervals for model predictions rely on the assumption that the model errors follow a probability distribution such as a Gaussian distribution [21, 27]. However, in practice, this assumption may not hold. The method for computing robust confidence intervals outlined in [28] is advantageous in that it makes no assumptions regarding potential distribution of the errors. It does, however, rely on the idea that errors of a given size generally occur with similar frequency as they have in the past. That is, provided that the model does not overfit the training data, errors of a given size that occur in the training and validation sets will occur with similar frequencies in the test set [21].

The procedure for computing robust confidence intervals is as follows. After obtaining the trained model, errors are computed on both the training and validation sets. Suppose the combined training and validation set (maintaining order) consist of values $\{x_1, \ldots, x_n\}$ with an embedding length of l. Then, starting with x_1, the first l values are fed into the trained model to predict $\hat{x}_{(l+1)}$. Taking the difference between the predicted value and the observed value, we obtain $e_1 = \hat{x}_{(l+1)} - x_{(l+1)}$. This process is repeated such that $e_i = \hat{x}_{(l+i)} - x_{(l+i)}$ until the end of the dataset is reached (Table 4).

Table 4. Procedure for obtaining the error set



$\{x_i$		$\cdots$		$x_{i+l}\}$	$\hat{x}_{i+1}$	e_i
x_1	x_2	x_3	x_4	x_5	$\hat{x}_6$	$\hat{x}_{i+1} - x_{i+1}$
x_2	x_3	x_4	x_5	x_6	$\hat{x}_7$	$\vdots$
x_3	x_4	x_5	x_6	x_7	$\hat{x}_8$	$\vdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\vdots$	$\vdots$

After the set of errors $e = \{e_1, \ldots, e_n\}$ is obtained, the errors are sorted in ascending order by value. The set of errors, minus a number of sample errors from each extreme, is used to build the robust confidence interval. For small datasets, the number of sample errors to be removed from each extreme of the error set is (n × p − 1), where n is the number of errors in the set and p is the corresponding significance level of the confidence interval. If (n × p − 1) is not a whole number, the floor function should be applied to it. For example, if the error set consists of n = 100 values and we seek a 95% confidence interval (p = .025), then (n × p − 1) = 1.5. Thus, one value should be removed from each extreme of the error set. We denote this new error set by e_{CI}. The confidence interval for each new predicted value, $\hat{x}_p$, is then formed as $\left\{\hat{x}_p + \min\left(e_{CI}\right), \hat{x}_p - \max\left(e_{CI}\right)\right\}$.

3.2.2.5 Replacement Strategy

Using AOSVR, we have the ability to detect anomalies in real time. After a new point is added, AOSVR updates the regression function to minimize the difference between the predicted value and what was actually observed. However, if the observed value is anomalous, using the anomalous value to update the regression function will decrease the performance of future predictions of nonanomalous data. By replacing the anomalous value with a value characteristic of the nonanomalous data before updating the regression function, we maintain a reasonable level of accuracy for future predictions. Our current strategy for obtaining this characteristic value is to replace an anomalous point with the average of the previous l points, where l is the embedding length.

3.2.2.6 Workflow for the Anomaly Detection Approach

Figure 37 depicts the workflow for the anomaly detection approach. First, the time series data is preprocessed, which includes normalizing the data between 0 and 1. Next, the SVR model is trained using data from the training set. Once the trained model is obtained, it is used to compute the training and validation set errors as outlined in the previous section. Next, the trained model is used to make predictions on the test set data. For each new test point, the model prediction and previously obtained error set is used to compute the confidence interval. If the corresponding observed value falls outside of the confidence interval (it is an anomaly), we replace the observed value with the average of the previous l values. That average value is then used in updating the regression function. If the observed value falls within the confidence interval (it is not an anomaly), the observed value is not replaced and is used to update the regression function. The updated model is then used to predict the next point, from which the process continually repeats until the end of the test set.

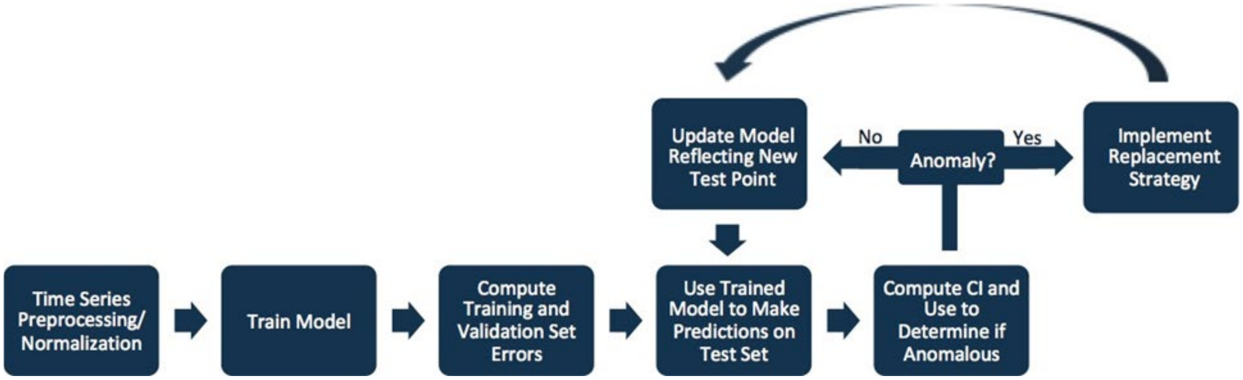

Update Model Reflecting New Test Point
No
Anomaly?
Yes
Implement Replacement Strategy
Time Series Preprocessing/ Normalization
Train Model
Compute Training and Validation Set Errors
Use Trained Model to Make Predictions on Test Set
Compute CI and Use to Determine if Anomalous

Figure 37. Anomaly detection workflow diagram.

3.2.2.7 Some Numerical Results

The anomaly detection approach summarized in the previous section was tested using EHR data from the VA CDW database to detect anomalous numbers of cancellations at a single hospital. Figure 38 shows the number of cancellations per day and detected anomalies using a 95% robust confidence interval. Although there appears to be one false positive, the approach is successful in detecting all true positives.

Figure 39 illustrates the inaccuracy that can occur for future predictions if a replacement strategy is not implemented before updating the regression function. We see not only an increase in false positives but also the addition of a false negative and a significant change in the overall prediction model.

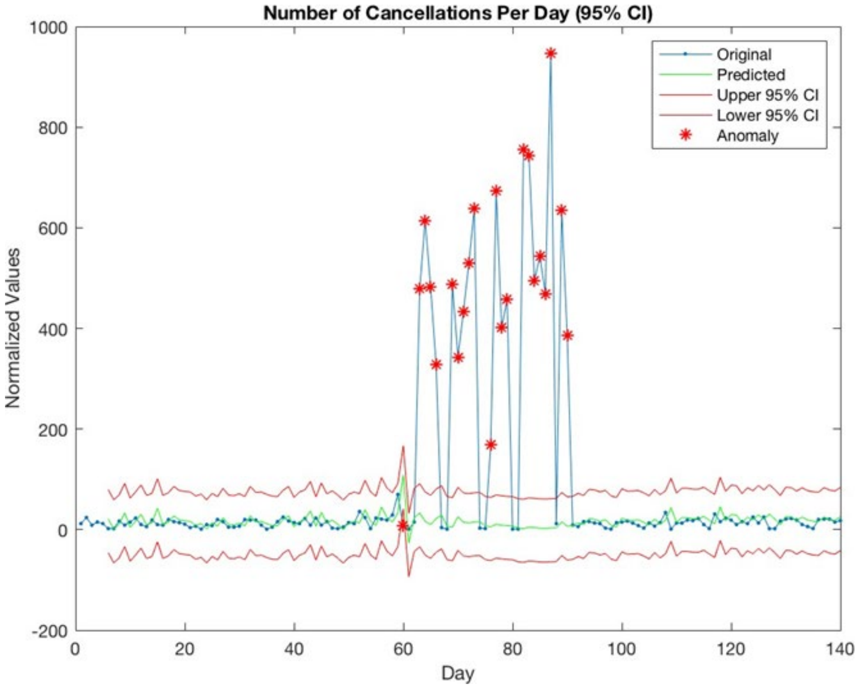


Figure 38. Anomaly detection with replacement.

Since, in the context of number of cancellations, HIT safety managers are only concerned with unusually high numbers of cancellations, we make a slight modification to our approach. That is, instead of considering points to be anomalous if they lie either above or below the upper and lower confidence bounds, respectively, we consider a point to be anomalous only if it is above the upper 95% confidence bound. With this modification, the replacement strategy is implemented only for points falling above the upper 95% confidence bound. Figure 40 depicts the number of cancellations per day where anomalies are detected using this modified strategy. Notice that now there are no false positives and that all anomalies are successfully detected.

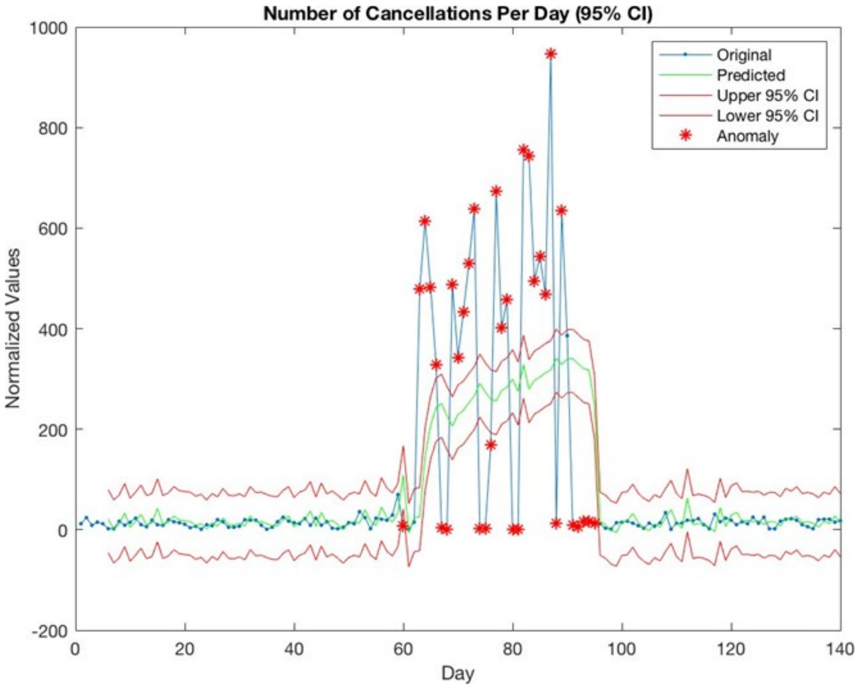


Figure 39. Anomaly detection without replacement.

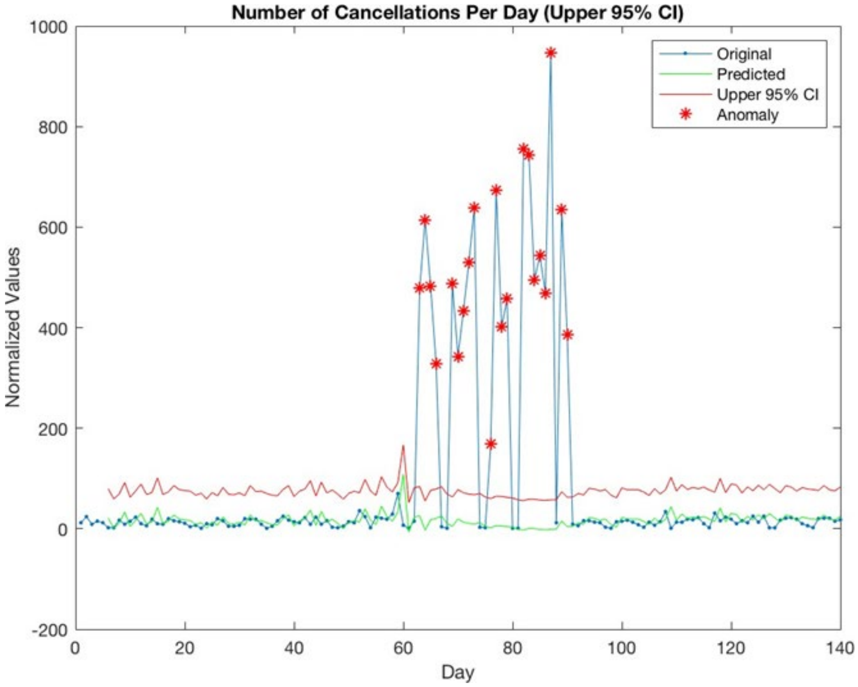


Figure 40. Anomaly detection with replacement (upper 95% confidence interval only).

3.3 SYSTEM-BASED RELIABILITY PERFORMANCE EVALUATION

This section focuses on the overall health of the HIT system. The assumption for this approach is that it is possible for the available detectors discussed in the previous sections to miss some hazards; therefore, there is a need for an additional capability that will survey the overall HIT system for emerging degradation in system reliability as a result of hidden or emerging hazards. Such a surveillance layer will provide insights that might lead to further investigation or the development of new detectors. In this section, we present a Markov chain (MC) model to understand system behaviors that might impact HIT reliability.

3.3.1 Perturbation Analysis Using Discrete Time Markov Chain

Specifically, we propose a discrete-time Markov chain (DTMC) to investigate the dynamic behavior of HIT using transition count data from EHRs. This is achieved by first developing a TPM for the clinical Consults workflow. The TPM represents a set of states in the workflow. The TPM is then converted into a DTMC as a basis for transition of orders between the states in the TPM. Then, a system transition matrix (STM) is created using the transition count data from EHRs to obtain the transition probabilities between states. The TPM is an ideal workflow description and has several degrees of freedom on use. Using a perturbation analysis algorithm, we systematically altered a combination of the transition probabilities. A simulation approach based on DTMC called Markov simulation was used to investigate the alternative workflow configurations and capture the dynamics of the submitted orders over longer time periods. These different workflow configurations will help analysts identify scenarios that could affect system reliability. The assumption for the Markov simulation is that the number of states in the DTMC model is readily reducible using the stochastic characteristics of MCs.

3.3.1.1 Discrete-Time Markov Chain

The MC model is a widely used analytical tool for studying the dynamic system behavior in several applications including health care (e.g., [29]). However, it has not been used to evaluate the reliability of HIT systems. In this section, we explore that opportunity. The MC is a discrete-time stochastic process where the conditional probability of any future event depends only on the present state and is independent of past states. The Markovian property can be expressed as follows for all states $i_0, i_1, i_2, \ldots, i_{t-1}, i_t$ and time $t \geq 0$:

$$P\left(X_{t+1} = i_{t+1} \mid X_t = i_t, X_{t-1} = i_{t-1}, \ldots, X_1 = t_1\right). \tag{1.6}$$

$$P\left(X_{t+1} = i_{t+1} \mid X_t = i_t\right). \tag{1.7}$$

Since the MC model assumes that the conditional probability does not change over time, for all states *i* and *j* and all *t*, $P\left(X_{t+1} = j \mid X_t = i\right)$ is independent of *t*, as expressed in Eq. (1.8):

$$P\left(X_{t+1} = j \mid X_t = i\right) = p_{ij}, \tag{1.8}$$

where p_{ij} is the transition probability that given the system is in state *i* at time *t* it will be in state *j* at time $(t+1)$.

3.3.1.2 The Perturbation Algorithm

The perturbation algorithm (PA) has been extensively studied in theoretical and computational literature. In the HIT context, the PA will focus on the highly probable workflow as opposed to brute force possible perturbations so that discovering high-impact disturbances can be automated. The PA [30] requires that a user first select the row of interest (the primary row) in the MC; the secondary columns are then the nonzero probability columns corresponding to the primary row. In addition, the user selects a perturbation limit, *L*— the maximum probability value for the primary column. Using the primary row, the nonzero element of interest (primary cell) is selected and the transition probability is increased/decreased by an increment amount, *v*, up to the user-defined *L* value. At the same time, the other nonzero elements in the primary row are decreased/increased, respectively, by the proportion of *v* using user-defined weight factors. The algorithm can also be used for simultaneous perturbation of two rows to capture where inter-row dependencies exist.

3.3.1.3 The Model Description

Our approach is motivated by an application of DTMC to high-performance computing systems [30]. The proposed approach first defines the Consult clinical workflow as a TPM. A simplified TPM representation is shown in Figure 41. This representation has one starting node (i.e., “New Consult Order”) and two terminating nodes (i.e., “End ofOrder” and “Cancel/Discontinue Order”). The different colors in the figure depict different states in the TPM. There are seven distinct colors that represent the seven (high-level) states for Consult orders. These states are *input*, *accept*, *schedule*, *revision*, *release*, *discontinue/cancel*, and *complete*. Some of these states can be decomposed into substates. However, for simplicity, we used only these high-level states for the analysis in this paper.

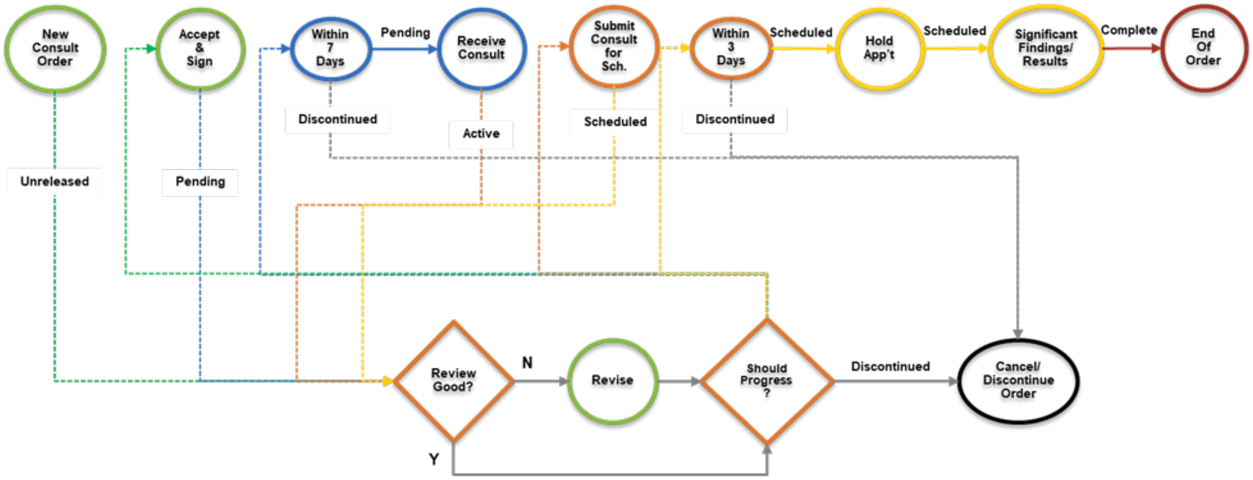


Figure 41. A simplified TPM for clinical Consult workflow.

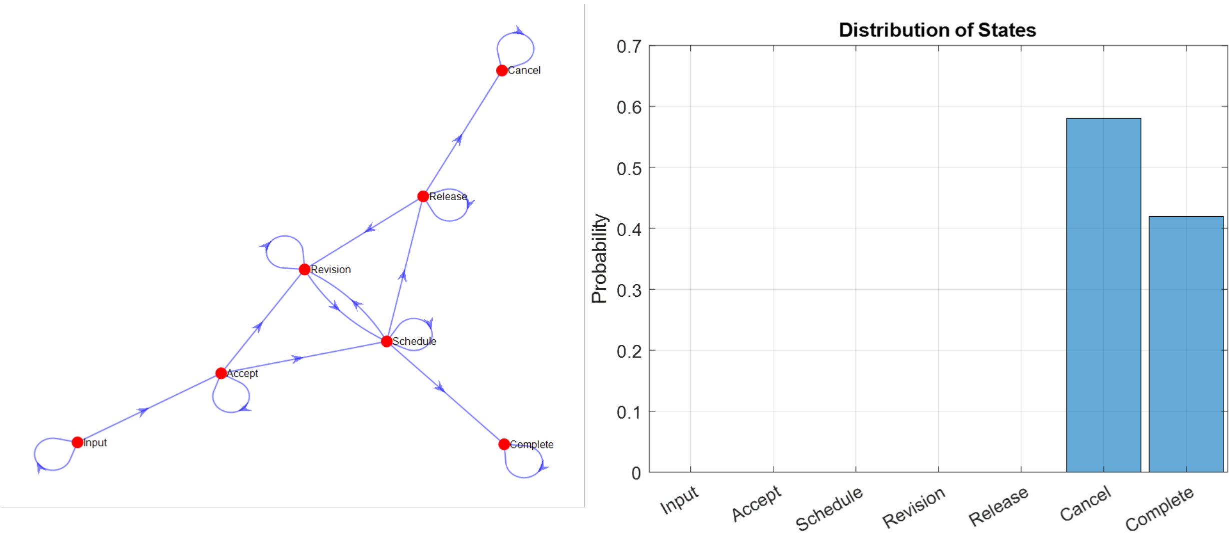
The TPM is converted into a MC representation. We then used transition counts (frequency of transition between states or nodes in MC) data to calculate the individual transition probabilities in the STM. Each cell in a STM is, thus, the probability of transition from state *i* at time *t* to state *j* at time *t* + 1, written as $s_i \to s_j$. The probability of transitioning between any two states, s_i, s_j, written as p_{ij} is given by Equation (1.9):

$$p_{ij} = \frac{f_{ij}}{\sum_{k=1}^{n} f_{ik}}, \tag{1.9}$$

where f_{ij} is the frequency of $s_i \to s_j$, $\sum_{k=1}^{n} f_{ik}$ is the sum of frequencies of transitions from s_i to all states s_k to which s_i could transition, *k* ranges from $1,\ldots,n$, and *n* is the number of states. The STM summarizes the dynamics of the underlying TPM. Thus, the STM provides an opportunity to change these probabilities and simulate the evolution of the system over time. The systematic perturbation of the STM will reveal a wide range of behaviors. In this section, we present some of the simulation results of the probability distribution in each state of the TPM. The results presented are based on the STM shown in Table 5. The corresponding MC for this STM is shown in Figure 42a.

Table 5. STM for Consult orders over a period of time

States	1 Input	2 Accept	3 Schedule	4 Revision	5 Release	6 Cancel	7 Complete
1	0.03	0.97	0	0	0	0	0
2	0	0.05	0.84	0.11	0	0	0
3	0	0	0.03	0.03	0.74	0	0.20
4	0	0	0.80	0.20	0	0	0
5	0	0	0	0.50	0.20	0.30	0
6	0	0	0	0	0	1.0	0
7	0	0	0	0	0	0	1.0



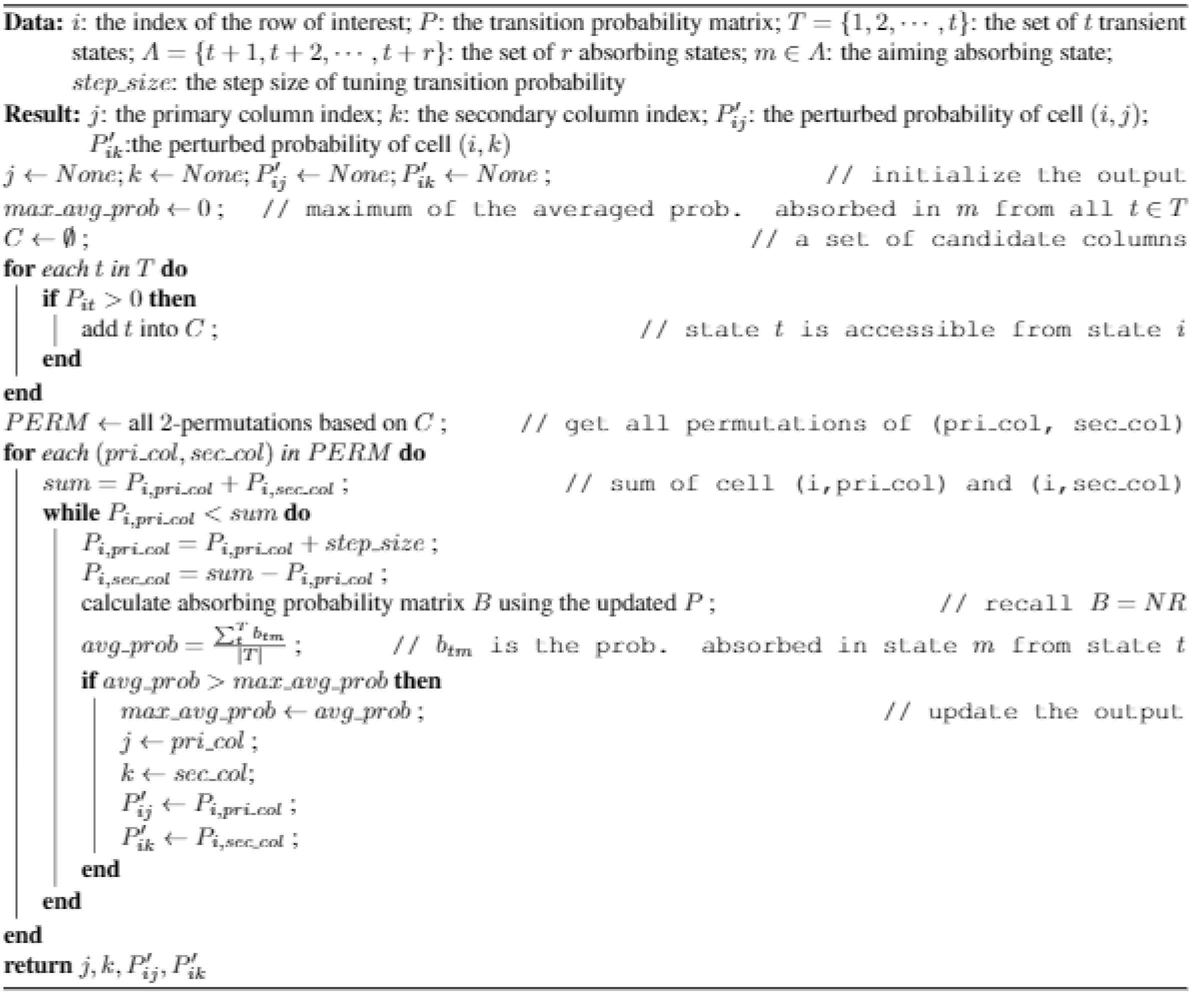
(a): The MC for the STM shown in Table 1

(b): PD for each state after 1000 random walks through the chain

Figure 42. The MC for the STM shown in Table 5.

The PA can be summarized as shown in Table 6. The PA requires that a user first select the row of interest (the primary row) in the MC; the primary columns, then, are the nonzero probability columns corresponding to the selected primary row. In addition, the user selects a perturbation limit (the maximum probability value for the primary column). Using the primary row, the nonzero element of interest (primary cell) is selected, and the transition probability is increased/decreased by an increment amount up to the user-defined limit. At the same time, the other nonzero elements (secondary columns) in the primary row are decreased/increased, respectively, by the same proportion. The algorithm can also be used for simultaneous perturbation of two rows to capture where inter-row dependencies exist. Intuitively, the purpose of perturbing a matrix is to increase or decrease the probabilities of being absorbed in certain states. For example, when scheduling tasks using MCs, we expect to increase the probability of successful completion and decrease the probability of errors.

Table 6. Row-wise perturbation algorithm



```
Data: $i$: the index of the row of interest; $P$: the transition probability matrix; $T = \{1, 2, \cdots, t\}$: the set of $t$ transient
    states; $\Lambda = \{t+1, t+2, \cdots, t+r\}$: the set of $r$ absorbing states; $m \in \Lambda$: the aiming absorbing state;
    $step\_size$: the step size of tuning transition probability
Result: $j$: the primary column index; $k$: the secondary column index; $P'_{ij}$: the perturbed probability of cell $(i, j)$;
    $P'_{ik}$:the perturbed probability of cell $(i, k)$
$j \leftarrow None; k \leftarrow None; P'_{ij} \leftarrow None; P'_{ik} \leftarrow None$ ;            // initialize the output
$max\_avg\_prob \leftarrow 0$ ;   // maximum of the averaged prob.  absorbed in $m$ from all $t \in T$
$C \leftarrow \emptyset$ ;                                                    // a set of candidate columns
for each $t$ in $T$ do
    if $P_{it} > 0$ then
        add $t$ into $C$ ;                                    // state $t$ is accessible from state $i$
    end
end
$PERM \leftarrow$ all 2-permutations based on $C$ ;      // get all permutations of (pri_col, sec_col)
for each $(pri\_col, sec\_col)$ in $PERM$ do
    $sum = P_{i,pri\_col} + P_{i,sec\_col}$ ;                   // sum of cell (i,pri_col) and (i,sec_col)
    while $P_{i,pri\_col} < sum$ do
        $P_{i,pri\_col} = P_{i,pri\_col} + step\_size$ ;
        $P_{i,sec\_col} = sum - P_{i,pri\_col}$ ;
        calculate absorbing probability matrix $B$ using the updated $P$ ;        // recall $B = NR$
        $avg\_prob = \frac{\sum_t^T b_{tm}}{|T|}$ ;       // $b_{tm}$ is the prob.  absorbed in state $m$ from state $t$
        if $avg\_prob > max\_avg\_prob$ then
            $max\_avg\_prob \leftarrow avg\_prob$ ;                              // update the output
            $j \leftarrow pri\_col$ ;
            $k \leftarrow sec\_col$;
            $P'_{ij} \leftarrow P_{i,pri\_col}$ ;
            $P'_{ik} \leftarrow P_{i,sec\_col}$ ;
        end
    end
end
return $j, k, P'_{ij}, P'_{ik}$
```

3.3.1.4 Some Numerical Results

The following numerical results demonstrate that the perturbation of a set of state transitions coupled with Markov simulation can be used to evaluate the reliability of EHR systems.

3.3.1.5 Evaluating the Proportion of Complete Orders

To evaluate the proportion of orders in the Complete state, we simulated 1,000 random walks through the MC shown in Figure 42a and compared the probability distribution (Figure 42b) for the seven states. This figure illustrates that the probability distribution (PD) of orders in *complete* state is less than the probability of orders in *cancel* state. Based on these results, analysts may be interested in increasing the PD of orders in *complete* state. To achieve this goal, we altered the transition probabilities in Table 1 as discussed in the following subsections.

3.3.1.6 Evaluating the Effects of Increasing the Proportion of Release Orders on the Overall System Performance

According to the 3rd row in Table 5, the transition probability (TP) of submitted orders from the *schedule* state to the *release* state is 0.74. As a process improvement strategy, we implemented a PA on the 3rd row such that $L = 0.90$ for the primary cell (column 5) in Table 5 at a rate $v = 0.02$. In addition, we decrease the TP for all the secondary columns in Table 5, except for column 7, to zero and the TP for column 7 to 0.10 at the same rate of v. An implementation of this algorithm resulted in a revision to the 3rd row as shown in Table 7. Figure 43 shows the results of the simulation, that is, the PD for orders in *complete* and *cancel* states, respectively. Figure 43a shows that an increase in the TP between the schedule and release states without a corresponding increase in the other TP values will result in a decrease in the PD for the *complete* state and an increase in the PD for the *cancel* state; this is probably due to the dependency between the *release* state and the *cancel* state in the 5th row in Table 5

Table 7. First modification to row 3 in Table 5

States	1 Input	2 Accept	3 Schedule	4 Revision	5 Release	6 Cancel	7 Complete
3	0	0	0	0	0.90	0	0.10

In addition, Figure 43b suggests that to avoid a situation where the PD of orders in *cancel* state is more than the PD of orders in *complete* state, the TP for *schedule* to *release* should not be more than about 0.785 (the break-even point for the two states). This figure also suggests that, even with the existing dependency in the 5th row, it is possible to achieve an increase in the PD for orders in the *complete* state by decreasing the TP for *schedule* to *release*. The results of a decrease in TP for column 5 are presented in following subsection.

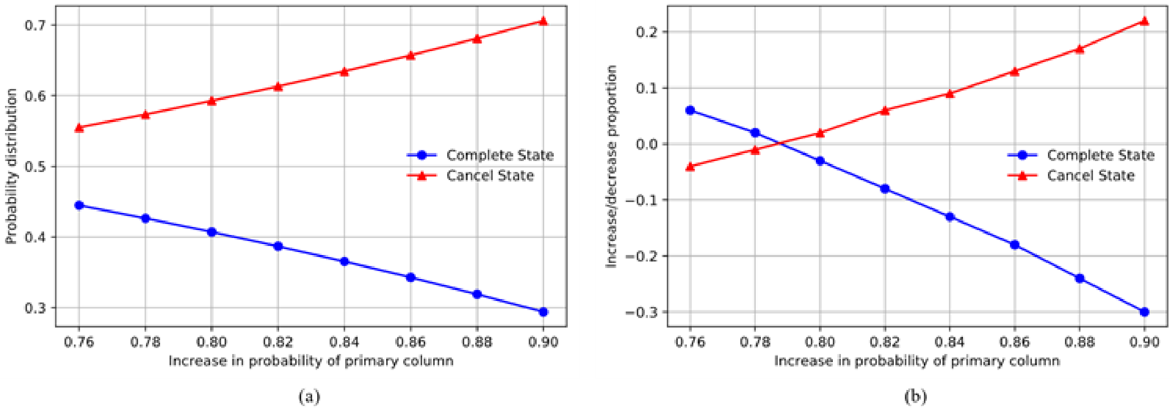


Figure 43. PD for the PA condition shown in Table 7.

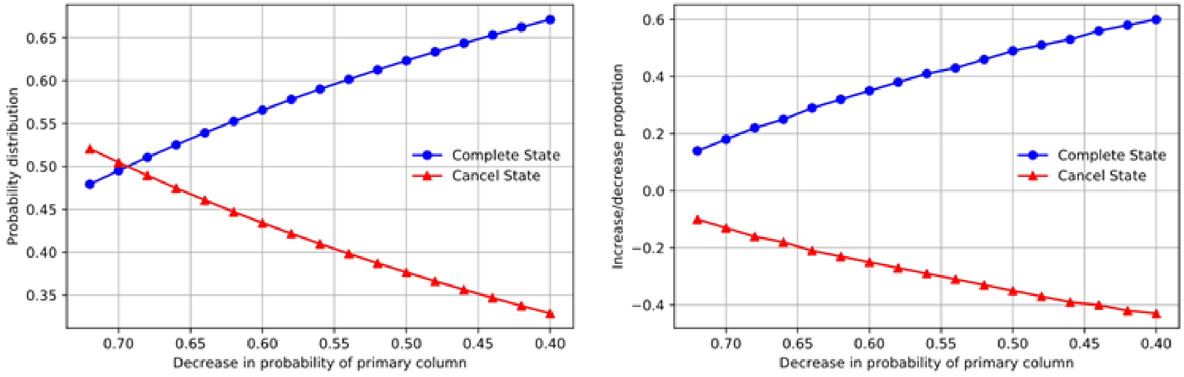
3.3.1.7 Evaluating the Effects of Decreasing the Proportion of Cancel Orders on the Overall System Performance

For this implementation, we modified the 3rd row in Table 5 as shown in Table 8. For this scenario and for the secondary columns, we also simulated combinations of the sum of TP that will achieve the highest PD for the *complete* states. These TPs are designated as a, b, c, and d in Table 8. Their sum should be equal to 0.60 and $v = 0.02$.

Table 8. Second modification to row 3 in Table 5

States	1 Input	2 Accept	3 Schedule	4 Revision	5 Release	6 Cancel	7 Complete
3	0	0	a	b	0.40	c	d

The PD for the *complete* and *cancel* states and the proportion of probability change are shown in Figure 44. Comparing these results to the PD in Figure 42, we see an increase in PD for the *complete* state. This perturbation simulation scenarios also shows a PD value less than 0.35 for the *cancel* state.



(a)

(b)

Figure 44. PD for the PA condition shown in Table 8.

4. AUTOMATED SURVEILLANCE SYSTEM TO DETECT HAZARDS IN HIT SYSTEMS

4.1 OBJECTIVES

The VA wishes to modernize its IT capabilities to manage the growth in scope and scale of its patient population to improve the health care services they provide to veterans and to manage the ever-increasing cost of health care delivery. The functionality described in this section represents a portion of the VA's efforts to use novel technologies to meet these objectives.

The HIT HD project objective is to design, prototype, and demonstrate automated surveillance for HIT HD. Initially, this capability will reside in ORNL's Private Health Information (PHI) enclave at Oak Ridge, Tennessee. The PHI enclave will host the automated surveillance tool that operates on the VA's CDW data. This section describes the automated surveillance system in terms of requirements, architecture, the prototype developed, and its verification and validation.

4.2 REQUIREMENTS

The functional requirements presented in this document are derived from discussions, documentation, and other input provided by VA SMEs. The majority of these requirements were derived during a full day, onsite workshop between the VA and ORNL team members held on February 26, 2019. Requirements analysis focused on the development of user personas and associated functional requirements.

4.2.1 User Personas

Five user personas have been identified to be supported by the automated surveillance tool:

1. Analyst
2. Hazard detector manager
3. HIT safety manager
4. Facility/regional safety manager
5. Point-of-care users

For the prototype system, the HD manager has been the focus of requirements gathering and analysis. The following high-level functional requirements have been identified for this persona.

1. View a summary page of results of all hazard detection by date. (Summary View)
2. View facility-level result details from a detector, including facility number, name, city, and state, in a map layout. (Explore View)
3. Implement VA VISN shapefile as base map. (Explore View)
4. View aggregate patient-level and order-level result details from an HD, including facility number and name, in a grid layout. (Details View)
5. Be able to identify an HD by the number/identifier used by ORNL modelers.
6. Know the last date/time an HD was run.
7. View hazard detection model details such as SQL/Python code, CDW tables/fields used, and any transformations/calculations.
8. Provide a summary of the hazard detection model details in narrative format.
9. Provide feedback to ORNL modelers regarding validity of HD or fields used in HD that can be used to improve HD performance.
10. Monitor HD performance over time to detect evidence of model degradation.

Future work will focus on the creation of additional requirements for the other identified personas.

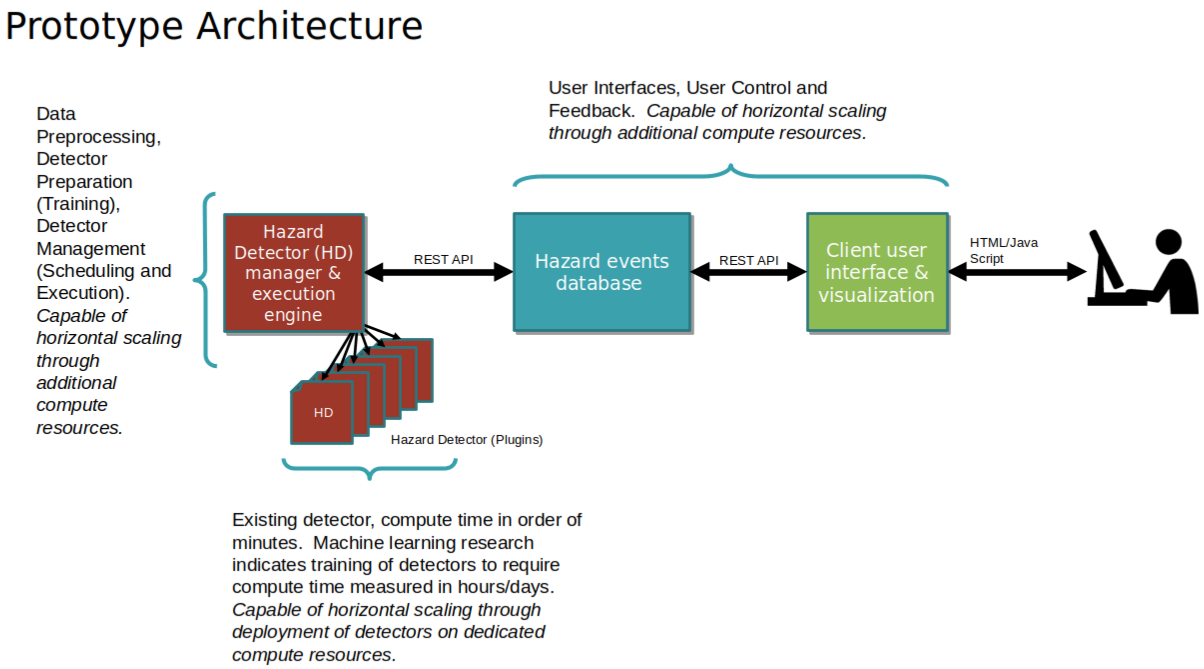
4.3 ARCHITECTURE

Regarding the architecture of the prototype system, it was noted early in the design process that scalability is a key concern for the project. Based on this significant consideration, a Representational State Transfer (REST)-style and microservices-based architecture was selected for the prototype.

REST-style architecture supports horizontal scaling (i.e., the addition of computing resources as needed throughout the application) in all components aside from the persistent datastore where vertical scaling (additional resources in a single logical server) is employed. The user interface, backend server communication hub, and detectors support the horizontal scaling capability. Microservices enable the loosely coupled and independently deployable approach of the REST services to meet the resiliency, elasticity, and responsiveness necessary to provide a next-generation solution to meet the VA's health care data needs. Using emerging technologies and high-performance mechanisms would enable us to seamlessly and rapidly further on-demand scalability [11].

Another significant scaling concern is scaling of the detectors, both in compute time and in development time. The prototype implements a hierarchical approach to detector development to address this concern. A hierarchical detector architecture speeds development time through reuse of data preprocessing approaches, event output formats, and common code reuse. The architecture also lends itself to deployment of the groups of detectors on dedicated computer hardware to handle compute scalability concerns as needed.

Figures 45 and 46 detail the overall architecture of the system and the hierarchical detector architecture employed in the system.



Prototype Architecture

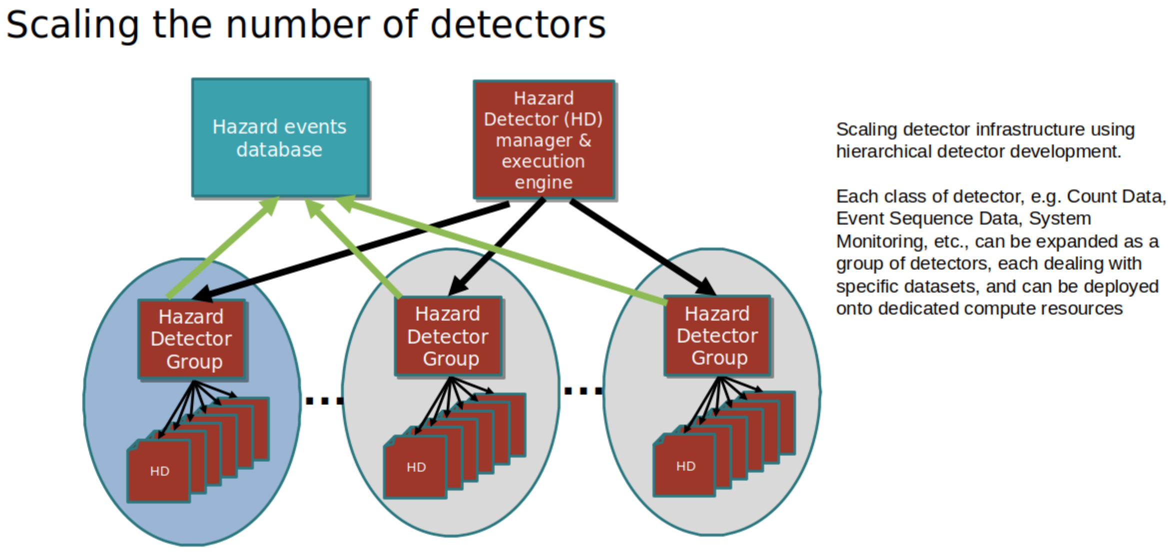
Data Preprocessing, Detector Preparation (Training), Detector Management (Scheduling and Execution). *Capable of horizontal scaling through additional compute resources.*

User Interfaces, User Control and Feedback. *Capable of horizontal scaling through additional compute resources.*

Existing detector, compute time in order of minutes. Machine learning research indicates training of detectors to require compute time measured in hours/days. *Capable of horizontal scaling through deployment of detectors on dedicated compute resources.*

Hazard Detector (HD) manager & execution engine
REST API
Hazard events database
REST API
Client user interface & visualization
HTML/Java Script
HD
Hazard Detector (Plugins)

Figure 45. Prototype architecture.



Scaling the number of detectors

Scaling detector infrastructure using hierarchical detector development.

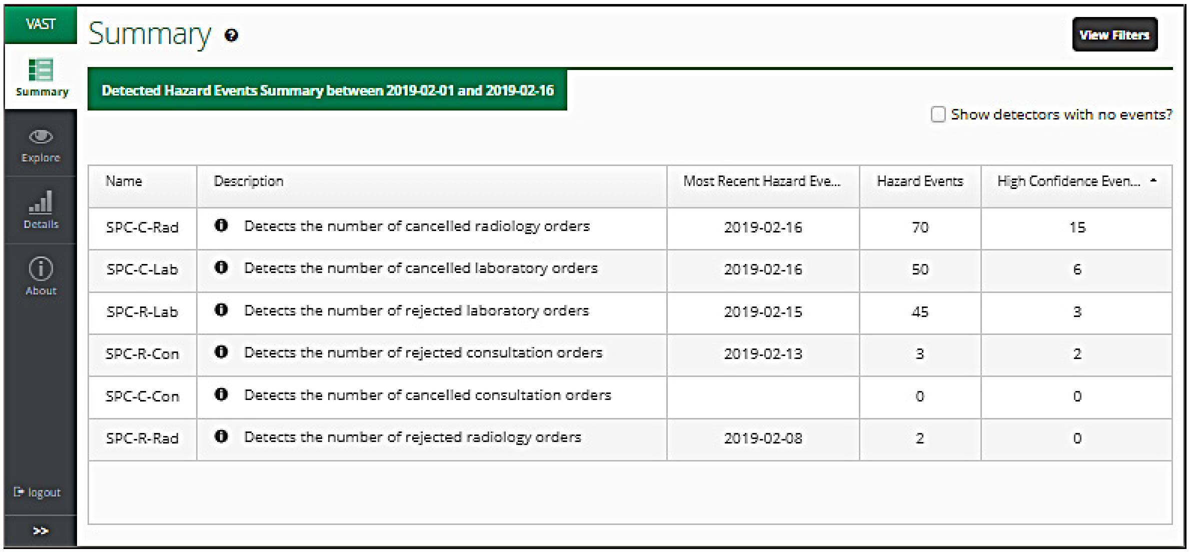
Each class of detector, e.g. Count Data, Event Sequence Data, System Monitoring, etc., can be expanded as a group of detectors, each dealing with specific datasets, and can be deployed onto dedicated compute resources

Figure 46. Scaling number of detectors.

4.4 PROTOTYPE

The prototype consists of three main views, shown in Figures 47–49.

Summary View—provides high-level summary of registered detectors. The view supports filtering by the date the potential detection occurred, allowing the end user to focus on particular time periods.



Summary

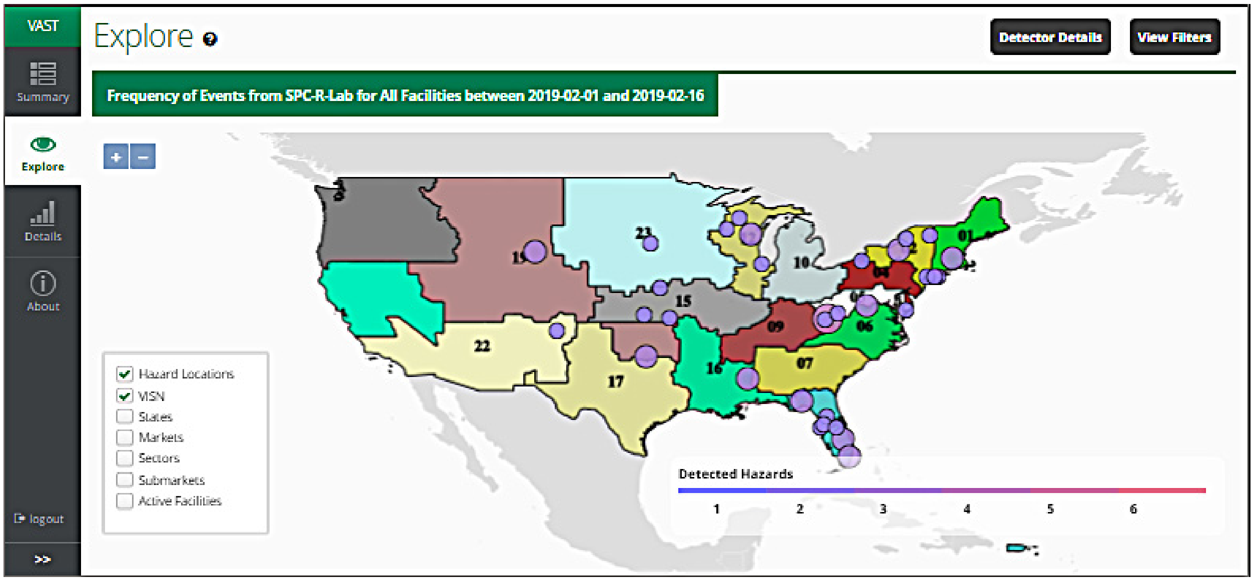
Detected Hazard Events Summary between 2019-02-01 and 2019-02-16

□ Show detectors with no events?

Name	Description	Most Recent Hazard Eve...	Hazard Events	High Confidence Even...
SPC-C-Rad	Detects the number of cancelled radiology orders	2019-02-16	70	15
SPC-C-Lab	Detects the number of cancelled laboratory orders	2019-02-16	50	6
SPC-R-Lab	Detects the number of rejected laboratory orders	2019-02-15	45	3
SPC-R-Con	Detects the number of rejected consultation orders	2019-02-13	3	2
SPC-C-Con	Detects the number of cancelled consultation orders		0	0
SPC-R-Rad	Detects the number of rejected radiology orders	2019-02-08	2	0

Figure 47. Summary View (Requirement 1).

Explore View—provides a map-based view for further exploration of information on potential detections. This view supports filtering by date as in the Summary View and adds support for filtering by a specific detector. This additional functionality allows the end user to modify the detector being visualized on the map without returning to the Summary View.



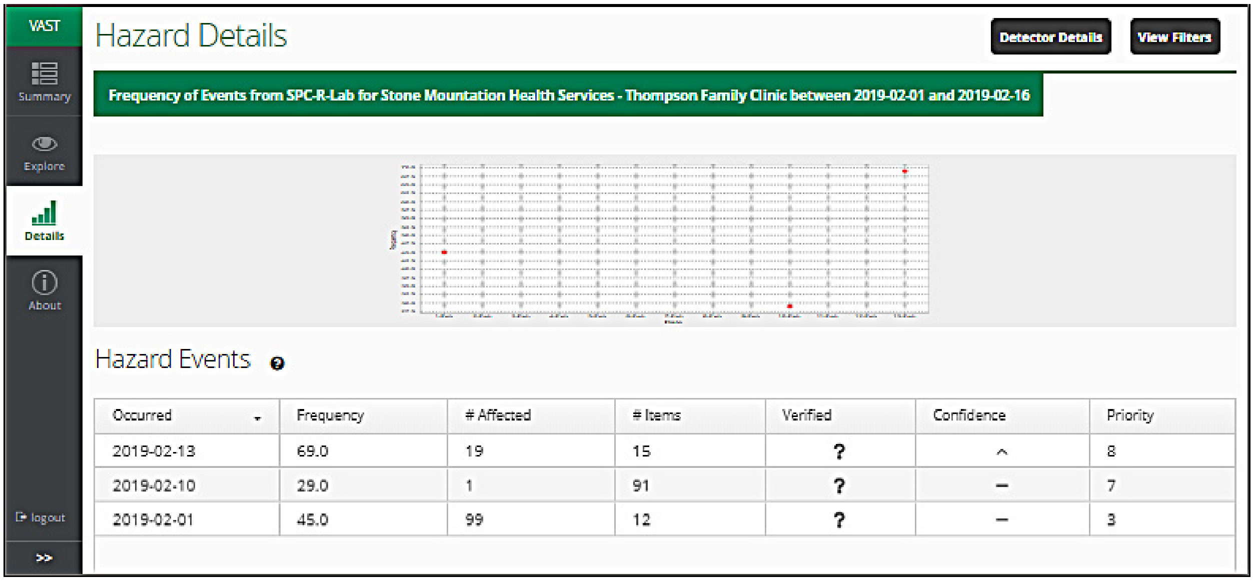
Explore

Frequency of Events from SPC-R-Lab for All Facilities between 2019-02-01 and 2019-02-16

☑ Hazard Locations
☑ VISN
□ States
□ Markets
□ Sectors
□ Submarkets
□ Active Facilities

Figure 48. Explore View (Requirements 2 and 3).

Details View—provides detailed information on potential detections. The view provides date and detector filtering as in the Summary and Explore views. Additionally, the view supports filtering by facility. This functionality allows the end user to modify the selected facility without returning to the Explore View.



Hazard Details

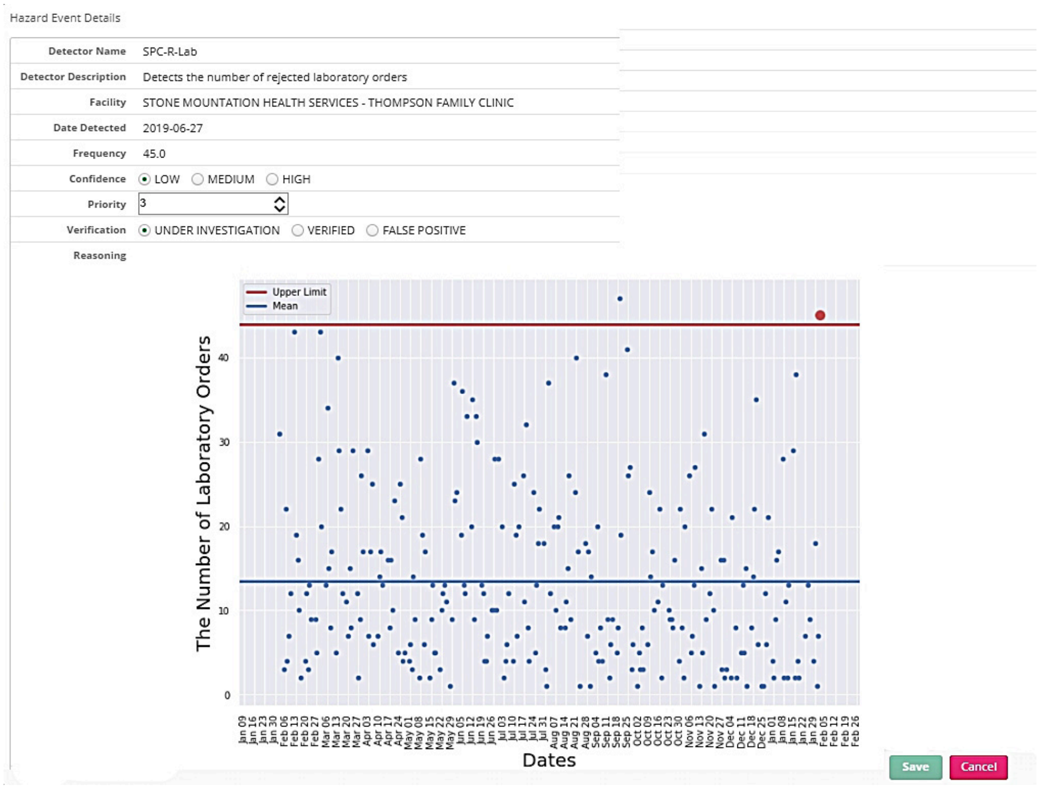
Frequency of Events from SPC-R-Lab for Stone Mountation Health Services - Thompson Family Clinic between 2019-02-01 and 2019-02-16

Hazard Events

Occurred	Frequency	# Affected	# Items	Verified	Confidence	Priority
2019-02-13	69.0	19	15	?	^	8
2019-02-10	29.0	1	91	?	–	7
2019-02-01	45.0	99	12	?	–	3

Figure 49. Details View (Requirement 4, 5).

Hazard Event Details Dialog (Figure 50)—provides more detailed information for a specific potential detection, as well as functionality allowing the end user to provide feedback on the potential detection.



Hazard Event Details

Detector Name	SPC-R-Lab
Detector Description	Detects the number of rejected laboratory orders
Facility	STONE MOUNTATION HEALTH SERVICES - THOMPSON FAMILY CLINIC
Date Detected	2019-06-27
Frequency	45.0
Confidence	● LOW ○ MEDIUM ○ HIGH
Priority	3
Verification	● UNDER INVESTIGATION ○ VERIFIED ○ FALSE POSITIVE
Reasoning	

Figure 50. Hazard Event Details Dialog (Requirements 5, 6, 8, 9).

4.5 VERIFICATION AND VALIDATION

Verification and validation (V&V) of the prototype is described herein in two threads, namely (a) the overall prototype (Section 4.5.1) and (b) additional V&V for individual HDs (Section 4.5.2).

4.5.1 Overall Prototype Verification and Validation

Regarding V&V activities for the prototype, a rigorous test-driven development paradigm has been employed throughout the project. These tests act as an automated gateway in the continuous integration and continuous deployment pipeline employed in the project. As new code is introduced in the main development branch of the project, automated scripts ensure tests are executed before automated deployment to the project's integration server.

A total of 42 tests have been developed that are executed in an automated fashion as part of every deployment cycle of the project. These tests cover the following functionality in the prototype:

- Detector Registration
- Duplicate Detector Registration – Expected Failure Case
- Detector De-Registration
- Hazard Event Reporting
    - Detector Not Registered – Expected Failure Case
    - Detector Not Running – Expected Failure Case
    - Facility Does Not Exist – Expected Failure Case
    - Metric Value Out of Range – Expected Failure Case
    - Missing Metric Value – Expected Failure Case

  - Missing Date Occurred – Expected Failure Case
  - Missing Date Detected – Expected Failure Case
  - Valid Event
  - Validated Query Counts from Reported Events
  - Duplicate Event Report – Expected Failure Case
- Hazard Event Feedback
  - End-User Feedback
- Detector Status
  - Running
  - Completed
  - Running – Detector Not Registered – Expected Failure Case
  - Running – Detector Not Running – Expected Failure Case
  - Running – Detector In Error State – Expected Failure Case
  - Completed – Detector Not Registered – Expected Failure Case
  - Completed – Detector Not Running – Expected Failure Case
  - Completed – Detector In Error State – Expected Failure Case
- Facility Related
  - Events by Detector at Facility
  - Facility API Integrity

In addition to these automated tests, on each deployment cycle to the integration server, a high-level functional test script is executed by human resources. This script covers the following functionality:

1. Login
2. Sorting Summary View
3. Filtering Summary View
4. Information Popups Summary View
5. Navigation from Summary View to Explore View
6. Layer Control Explore View
7. Map Controls Explore View
8. Detected Events Control Explore View
9. Right-Click Facility Explore View
10. Detector Information Control Explore View
11. Filter Explore View
12. Navigation from Explore View to Details View
13. Information Popups on Details View
14. Sorting Details View
15. Filtering Details View
16. Detector Information Control Details View
17. Provide Feedback Details View
18. Expand/Collapse Menu Bar
19. Logout

Any errors that are detected are immediately noted as issues and addressed before moving to the next phase of the deployment life cycle. On a defined schedule and after all tests pass, code is migrated from the project's integration server to ORNL's Knowledge Discovery Infrastructure environment.

4.5.2 Additional Verification and Validation for Hazard Detectors

The range of V&V techniques for the HDs is as varied as the approaches used in the detectors themselves. For the currently implemented HDs, specific V&V tests were developed. SPC-based HDs are verified using metamorphic testing and are validated using two techniques (see [31]). Performance analysis for the HD was conducted and is presented in [32]. The SPC-based Rejected Orders HD passed all V&V tests. The validation was simple on this detector, which is fully defined by the heuristic used in the detector. However, the SPC-based Cancelled Orders HD is not fully defined by its heuristic and requires SMEs as humans-in-the-loop to identify those events or event sets that constitute true hazards of this type. Once that is done, a new detector can be created that combines the original heuristic plus the behavior identified by the SMEs. The new detector will then be fully defined by its heuristic and learning and can then be tested using traditional classifier evaluation methods. Given the lack of SME participation in the active learning loop, the SPC-based Cancelled Orders HD has not yet been fully validated.

5. PROJECT LESSONS LEARNED

Any data science effort (or data-driven study) starts with the fundamental assumption that ***data exist***. The second assumption is that ***patterns exist in the data for which we are seeking***. The fundamental difficulty in this project was determining what part of the data (in the massive CDW) we should focus on and how we could represent the data that exhibits patterns indicative of hazards. Success requires bottom-up exploratory querying as well as top-down revision of the research questions we ask. Domain expertise was critical in guiding this iterative process. We came up with two data models that allowed us to explore the potential of hazard detection in some cases and demonstrate hazard detection algorithms in others. However, this ongoing process will lead to further identification of data columns in CDW and will fine-tune and improve the detection algorithms, as well as develop new methods. The following list is a mix of know-how we derived during our exploration:

- CDW has filtered VistA data, which means that not all data fields in VistA are transferred to CDW.
- CDW is a relational database consisting of views that are organized for different purposes (mostly not research related).
- Many columns in CDW can be updated; therefore, for those columns we observe only the latest update. We do not have access to logs of VistA or the full history of those data fields.
- If we would like to get an almost full history of a patient or an order from CDW, we need to design our own data listener that will record every update that ETL processes operate on the columns of interest. This is possible through VA's Special Purpose Views (SPV) database within CDW (technically CDW data but with ETL operations and times recorded). Even then, we can capture only daily history. For example, if three updates are done on a Results Date Time column on the same day, we will capture only the last update.
- CDW ETL processes run every day after midnight transferring the data of yesterday. Although, this process is expected to be done by 8:00 am, it is sometimes possible that the ETL processes continue all day (sometimes transferring data not from yesterday but for the day the ETL process is running).
- We found that everything starts with an order (a file 100 record in VistA), and the transactions on orders (their history) are indicative of adverse events.
- We also found that anything is possible in VistA! It is a highly configurable and flexible system that allows some users to bypass known workflows. Because workflows are not strictly enforced and constrained, orders can transition from any state to any other state.
- We usually asked this question: If anything is possible in VistA, what should not be possible? We first investigated normal patterns but then realized there can be many normal patterns! So, we found investigating rare/irregular patterns to be one of the directions we should take.
- Another direction to take is to follow an important design principle; that is, to focus on the most common operations (events), with a focus on analyzing the most visited path to optimize the

common case. Those processes should be made as fast and as effective as possible, which could bring important improvements and resource savings to the organization.

- Since we do not have labeled hazard data, we had to focus on unsupervised learning methods. Basically, we find patterns, discuss those patterns with SMEs, try to extract explicit definitions of hazards from those discussions, and, if agreed, design detectors. This was successful for some of the SPC-based detectors such as finding anomalous numbers of rejected order counts, cancellations of machine-generated orders, or cancellations with machine-generated reasons of discontinuation.
- We wanted to keep our methods as generalizable as possible, which is why we did not go into orderable item name-level analysis. We treated each order type (Radiology, Consults, etc.) uniform within to extract generalizable patterns. However, the need to break the orders into further groups based on ordering facility or subdomain was found to be necessary for the next round of analysis. SMEs require additional information before deciding on explicit hazard definitions, and our iterative process will continue to extract more relevant fields to address this need.
- Additionally, we keep an order features table in the HITHD database that we have not made much use of so far. That table will need further exploration to extract any feature per order that helps us to distinguish its characteristics. Orderable item name is one of the features, urgency is another, signature required could be another, etc.
- Applying process mining has proven useful in identifying and analyzing process models and in identifying outliers. Process mining helps us to gain insight on possible anomalies in HIT.
- The use of the OASIS Human Task state transitions diagram was helpful to define termination states and cases where clinical orders did not get completed successfully. However, we still need to look at the original event sequence to identify what went wrong.

6. SUMMARY

Most of the work done on this project in FY 2019 was on CDW exploration for better understanding of the data, as well as the development of methods for detecting anomalies in count data. Consequently, we gained significant insights on how the data is collected while analyzing some of the datasets. Clinical orders are found to be the most important atomic units of VistA, and they are of particular interest for safety officers in VHA. Moreover, transactions that occur during the life cycle of an order can be indicative of adverse events and interruptions in care delivery. Hence, we focused our efforts on revealing significant patterns in order-related data. Furthermore, raw event sequences can be for multiple exploratory analysis such as calculating statistics on durations between transitions. Both directions were explored in FY 2019. The CDW data presented unique features that led to the development of new methods for analyzing breakpoints in time series data. Therefore, we explored both statistical and machine learning techniques in an online approach. Both methods led to some promising results. The design and development of the surveillance tool helped address some of the challenges about ways to present model outputs to safety managers.

7. REFERENCES

[1] OASIS, *Web Services Human Task (WS-Human Task) Specification Version 1.1, Committee Specification Draft 12/Public Review Draft 05*. 2012. http://docs.oasis-open.org/bpel4people/ws-humantask-1.1.html.

[2] H. B. Klasky et al. 2019. *Process Mining In Healthcare – A Case Study for the Corporate Data Warehouse of the Veterans Affairs Office*. ORNLTM-2019/1302. Oak Ridge: Oak Ridge National Laboratory.

[3] O. A. Omitaomu, O. Ozmen, M. M. Olama, L. L. Pullum, T. Kuruganti, J. Nutaro, H. B. Klasky, H. Zandi, A. Advani, A. L. Laurio, M. Ward, J. Scott, J. R. Nebeker. 2019. "Real-Time Automated Hazard Detection Framework for Health Information Technology Systems." *Health Systems*. Doi: 10.1080/20476965.2019.1599701.

[4] O. Ozmen, H. B. Klasky, O. A. Omitaomu, M. M. Olama, T. Kuruganti, L. L. Pullum, M. Ward, J. Scott, A. L. Laurio, and J. R. Nebeker. 2019. "Topic Modeling to Discern Irregular Order Patterns in Unlabeled Electronic Health Records," *IEEE-Embs International Conference on Biomedical and Health Informatics* (Bhi 2019), Chicago.

[5] O. Ozmen, H. B. Klasky, O. A. Omitaomu, M. M. Olama, T. Kuruganti, L. L. Pullum, M. Ward, J. Scott, A. L. Laurio, and J. R. Nebeker. 2019. "Reverse-Engineering Event Sequence Data from Clinical Order Transactions for Machine Learning Techniques," Informs, Seattle.

[6] T. Mikolov, Q. V. Le, and I. Sutskever. 2013. "Exploiting Similarities Among Languages for Machine Translation." Arxiv Preprint Arxiv:1309.4168.

[7] J. Han, G. Dong, and Y. Yin. 1999. "Efficient Mining of Partial Periodic Patterns in Time Series Database." *In Proceedings 15th International Conference on Data Engineering,* Cat. No. 99cb36337. IEEE. pp. 106–115.

[8] M. Asay. 2019. "Why Time Series Databases Are Exploding in Popularity." *TechRepublic*. https://www.techrepublic.com/article/why-time-series-databases-are-exploding-in-popularity/.

[9] Influxdb. 2019. https://www.influxdata.com/time-series-database/.

[10] P. Dix. "Why Time Series Matters for Metrics, Real-Time Analytics and Sensor Data." 2019. Influxdb.

[11] R. Karthik, A. Advani, J. Arnold, E. Begoli, M. Bowie, and H. Klasky. 2018. *Design of Microservices Platform for Scalable Clinical Analytics*. ORNL/TM-2018/974. Oak Ridge: Oak Ridge National Laboratory.

[12] K. A. Barchard, and L. A. Pace. 2011. "Preventing Human Error: The Impact of Data Entry Methods on Data Accuracy and Statistical Results." *Computers In Human Behavior* 27(5): 1834–1839.

[13] S. Bowman. 2013. "Impact of Electronic Health Record Systems on Information Integrity: Quality and Safety Implications." *Perspectives In Health Information Management* 10.

[14] S. I. Goldberg, A. Niemierko, and A. Turchin. 2008. "Analysis of Data Errors in Clinical Research Databases." *In AMIA Annual Symposium Proceedings Archive*. 242–246. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2656002/.

[15] G. R. Kim, A. R. Chen, R. J. Arceci, S. H. Mitchell, K. M. Kokoszka, D. Daniel, and C. U. Lehmann. 2006. "Error Reduction in Pediatric Chemotherapy: Computerized Order Entry and Failure Modes and Effects Analysis." *Archives of Pediatrics & Adolescent Medicine* 160(5): 495–498.

[16] J. T. Mitchel, Y. J. Kim, J. Choi, G. Park, S. Cappi, D. Horn, M. Kist, and R. B. D'agostino Jr. 2011. "Evaluation of Data Entry Errors and Data Changes to an Electronic Data Capture Clinical Trial Database." *Drug Information Journal* 45(4): 421–430.

[17] C. Truong, L. Oudre, and N. Vayatis, N. 2018. "A Review of Change Point Detection Methods." Arxiv Preprint Arxiv:1801.00718.

[18] R. P. Adams, and D. J. Mackay. 2007. "Bayesian Online Changepoint Detection." https://arxiv.org/abs/0710.3742.

[19] P. Fearnhead. 2006. "Exact and Efficient Bayesian Inference for Multiple Changepoint Problems," *Statistics and Computing* 16(2): 203–213. https://link.springer.com/article/10.1007%2Fs11222-006-8450-8

[20] X. Xiang, and K. Murphy. 2007. "Modeling Changing Dependency Structure in Multivariate Time Series" ICML, 1055–1062. DOI: 10.1145/1273496.1273629.

[21] A. L. I. De Oliveira and S. R. De Lemos Meira. 2006. "Detecting Novelties in Time Series Through Neural Networks Forecasting with Robust Confidence Intervals." *Neurocomputing* 70: 79–92.

[22] J. Ma, J. Theiler, and S. Perkins. 2003. "Accurate On-Line Support Vector Regression." *Neural Computation* 15: 2683–2703.

[23] B. Norgeot, B. Glicksberg, and A. Butte. 2019. “A Call for Deep-Learning Healthcare.” *Nature Medicine* 25: 14–15.

[24] O. A. Omitaomu. 2013. *Intelligent Process Monitoring and Control Using Sensor Data*. Germany: Lambert Academic Publishing.

[25] I. D. Lins, E. L. Droguett, M. Das Chagas Moura, E. Zio, and C. M. C. Jacinto. 2015. “Computing Confidence and Prediction Intervals of Industrial Equipment Degradation by Bootstrapped Support Vector Regression.” *Rel. Eng. Sys. Safety* 137: 120–128.

[26] V. Vapnik. 1998. *Statistical Learning Theory*, Wiley.

[27] D. C. Montgomery, G. C. Runger, and N. F. Hubele. 2001. *Engineering Statistics*, Second Edition, Wiley.

[28] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. Depristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean. 2019. “A Guide To Deep Learning In Healthcare.” *Nature Medicine* 25: 24–29.

[29] A. Wright, T. T. T. Hickman, D. McEvoy, S. Aaron, A. Ai, J. M. Andersen, S. Hussain, R. Ramoni, J. Fiskio, D. F. Sittig, and D. W. Bates. 2016. “Analysis of Clinical Decision Support System Malfunctions: A Case Series and Survey.” *Journal of the American Medical Informatics Association* 23(6): 1068–1076.

[30] C. Dabrowski and F. Hunt. 2009. “Using Markov Chain Analysis to Study Dynamic Behavior in Large-Scale Grid Systems.” In *Proc. of the 7th Australasian Symposium Pm Grid Computing And E-Research* (Ausgrid 2009), Wellington, New Zealand.

[31] L. Pullum. 2019. “Hazard Detector Verification and Validation.” ORNL Technical Note, September 30.

[32] L. Pullum. 2019. “Hazard Detector Performance.” ORNL Technical Note, August 30.