Understanding Operating Room Scheduling - Uncovering the Hidden Patterns in the Data

Donald M. Voltz, MD[1], Matthew Joy, MD[2], Alfred Pinchak, MD, PhD[2]

1. Department of Anesthesiology. Aultman Hospital, Canton, Ohio
2. Department of Anesthesiology. MetroHealth Medical Center, Cleveland, Ohio

Introduction:

The operating room is a complex, capital and operational intensive area of any health system. It sits at the unique position where various services and specialties coexist under a constant pressure of fluctuating demands generated from both anticipated and unanticipated patient needs and the need for operational efficiency and resource coordination. The operating room sits at a crossroad, dependent on ambulatory clinics, emergency department, ICU, PACU and patient care areas all impacting flow. Any bottleneck that develops within interdependent locations, equipment and personal impacts the operational efficiency expenses, productivity, and ultimately patient satisfaction.

Anesthesiologists are being asked to address the flow and inefficiencies of the OR. Many are being asked to address process and quality issues because of their unique position in this complex ecosystem. Anesthesiologist often rely upon their emperical knowledge of surgeons, nursing, cases and equipment to make decisions and help maintain effective flow through the pre-operative process. They are increasing being asked for input to guide process and policies of the OR. The use of data-driven validation of these assumptions is becoming more important for anesthesiologist to understand and apply to scheduling and performance measures of any procedural location.

Operational efficiencies, resource, capacity and changing reimbursement patterns have brought more attention to the need for data-driven management, and the development and testing of models to optimize and respond to changing demands. Models of operating room performance, resource scheduling and dashboards have been and continue to be developed in an attempt to better predict resource allocation, thus optimizing operational expenses. These models often require data contained in multiple databases included hospital electronic health records (EHR's), operating room management systems, anesthesiology information management systems (AIMS), and supply chain and financial systems. The extraction, cleansing, and integration of data from various sources can be a time consuming and challenging process fraught with errors. Reports and dashboards have been developed to aggregate and present the metrics needed, however, building additional data products, testing new hypothesis or modeling different predictive models often requires new data extraction, formatting and reporting. In addition, these often internally developed solutions for data analysis are difficult to share and build upon.

Platforms and frameworks allow for enhanced productivity and collaborative development of robust and testable code. This technology paradigm speeds the development of data products by standardizing common data processing tasks including the ability to clean and aggregate large data sets extracted from various data repositories, allowing for use across systems where proprietary data formats do not easily support interoperability. Borrowing from the concept of open source technology used in other business sectors including finance[1], we have begun the development of an interactive operating room analytics platform for use in healthcare data analysis, operational research, development of static and interactive visualizations, along with the ability to share and expand the underlying capabilities as new algorithms and models are developed.

Understanding the various players, flow and variability in a system such as an operating room requires standardization along with customization. Architecting a technology platform that is able to handle common data type conversions along with the ability to add and modify algorithms in a stable, testable and scalable manner is the purpose of this project. This poster will demonstrate some of the functionality of the framework, but the value comes from the ability to define experiments and test hypothesis about operating room performance, operations, flow, risks, and financial performance.

Methods:

Development of an analytics framework starts with selecting an underlying programming language. For this framework, we choose python[2]. Python is an open source computer language supporting the programing paradigms of object orientation as well as functional programming. It has a robust user base in financial, scientific and data computing with the availability of numerous frameworks that have been rigorously tested and are actively being enhanced. Python is a dynamic, general purpose programming language that is easy to learn, runs on all operating systems, is the fastest growing programming language with high programmer productivity. Given these benefits, it is currently being used for many mission critical applications in large corporate, scientific and data intensive areas. Python supports an interactive programming environment, iPython[3], which encapsulates code, data, rich statistical, graphing and interactivity making it an agile utility for exploring ideas, interacting with data and sharing self- contained notebooks to support reproducible results.

The principle foundations of agile development include the delivery of valuable software, rapidly addressing changes, providing an environment to support the need's and perspective of individuals, along with effective and efficient release of applications. These principles were the driving force behind the development of a framework for OR management. With an increasing amount of data (volume), coming from different sources and formats (variety), at ever increasing rates (velocity), and being used to draw differing conclusions (veracity), we found a need to build and test common components in an effort to save time, reduce errors, and limit the creation of unmanageable database quarries specific for each database.

Management of the OR starts with people, cases, and equipment. Using these physical objects, we modeled the attributes and interactions for the foundation of this framework. With the mindset of abstracting consistent attributes along with common mathematical and statistical operations, the framework began to evolve into a useful tool for looking at different aspects of the OR. At a high level, we looked for ways to represent the various pieces of data being collected by OR management, scheduling, financial, supply, AIMS and EHR databases. Using this abstracted view of components, we developed classes to represent real-world concepts, verified and validated code to which data elements can be added for use in more complex analysis or applications. The framework architectural overview (Figure 1) consists of various components that can be combined to generate any number of views into the data for analysis, simulation or visual representation. In a manner similar to software development, the framework components can be combined and extended to develop custom solutions to understand, experiment with and predict OR performance.

In order to understand the design decisions and programming choices, it is best to look at the overview how raw data is turned into a data products; the use of multiple data elements in clever ways, iterating through and combining these elements to solve data-intensive problems that are otherwise difficult, error prone, or not possible with other tools such as database queries, spreadsheets or limited data reports. Figure 2 shows the path from data to data products using this framework. To understand the framework, it is useful to look at some example uses from OR data collected over a 5 year period beginning in 2010. Some of the functionality of the framework capabilities are highlighted using interactive iPython notebooks to process, analyze and visually display the results.

Figure 1: Architectural overview of the perioperative management framework - OR Metrics. The framework consists of a set of core classes to serve as a blueprint for the generation of objects - independent, yet interactive pieces of computer code consisting of attributes and behaviors (methods) that act upon data in the generation of indices and metrics. When these individual pieces are combined and/or processed, new insights can be gained and presented as raw data output or graphical representation. This framework has been architected to provide a robust baseline structure for perioperative data analysis as well as serve as a standardized platform to extend and adopt for any given procedural location.

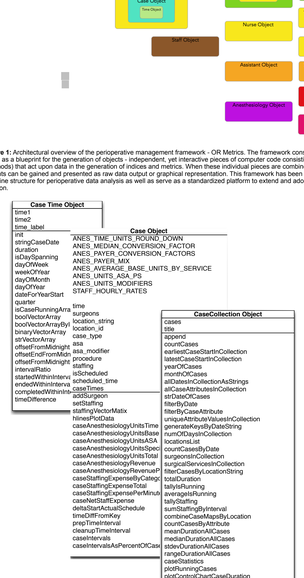

Figure 3: Class representation of some of the core functionality of the framework including CaseTime, Case and CaseCollection Class as UML Format.

Figure 2: Flow diagram demonstrating a high-level flow of how the OR Metrics framework is applied to the evaluation of data obtained from an operating room environment.

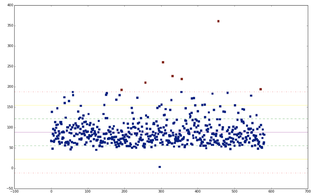

Figure 4: Control chart representation of case duration for two surgeons performing the same procedure (laparoscopic cholecystectomy). Mean value as well as deviation (in standard deviations) represented across all cases performed for 2014 for these two surgeons.

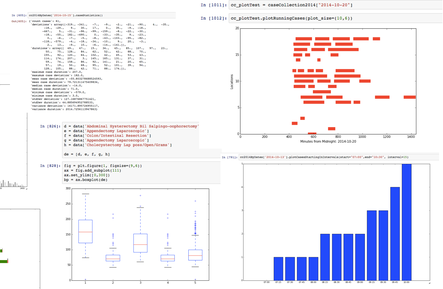

Figure 6: Various data analysis & plots to highlight some of the insight to be gathered from OR data.

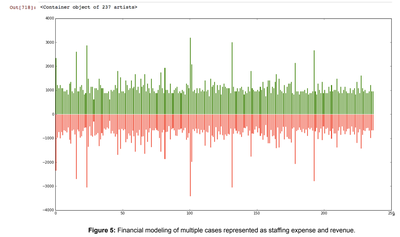

Figure 5: Financial modeling of multiple cases represented as staffing expense and revenue.

Results:

Complete case data was extracted over the course of many years (2010 - 2015) into a comma-delimitated flat file. This file was brought into the active iPython notebook using a high-performance data structure framework called Pandas[4]. Pandas readily coverts flat file data into a two-dimensional data structure called a dataframe.

This dataframe class is able to hold large amounts of data and is easily accessed for the next step, the creation of Case classes for each case represented in the original data set, complete with a variety of built-in and customizable attributes. Iterating through the entire set of data and the creation of a time class along with a case class for each OR case, irrespective of the format for the individual data elements, becomes automated using a principle of functional programming. The created Pandas dataframe supports the concept of iteration where an object (the dataframe) represents a stream of data. This iterator returns one element at a time. Coupling this iteration with a concept of list comprehension, the entire set of data can be efficiently converted into customized classes where data elements are standardized and validated for errors. Standardization of data elements is an important component of any framework since other objects, algorithms and methods depend on this verified data. Given the various ways data can be stored in databases and flat files, each element must be converted into a format that is known and acceptable for the rest of the code to utilize. Standard programming loops are not as processor efficient and can result in delays and crashes when working with large datasets. The decision to incorporate the Pandas dataframe was to prevent these issues as well as to efficiently convert raw data into objects that can be acted upon using the rest of the framework.

The OR management framework being presented here consists of a group of objects, each serving a unique and encapsulated set of both attributes (data representing a given state of some real world entity) and behaviors (methods to process and produce new data) given the current state of of the object. Using principles of object-oriented programming, the authors have abstracted the real-world pieces of the operating room to create interdependent and interoperable objects to represent surgical procedures (cases), people, and equipment (schedule, location availability and constraints). Designing a model for individual object composition as well as how the objects can be combined to create a realistic and reproducible understanding of how the OR performs from an operational, efficiency, and financial perspective has allowed us to uncover and present patterns that heretofore required high-level, custom data extraction and programming skills to realize. This presentation is to highlight the construction of the OR management framework as well as to present some example use cases where we have used the framework to gain insight into the performance of different OR's over a period of 5 years.

The main functional entity of an operating room is the Case class and this was used as the base unit in the development of the framework. Using standard attributes including start and stop times, surgeon(s), procedure, anesthesiology specific case characteristics, staffing and resource requirements, and other descriptive characteristics, we created an abstract computer representation that would allow for any possible type of case that might be performed. This abstract representation consists of various attributes (Figure 3) and the resulting behaviors and indices that might be needed to perform analysis on a given case of a group of many cases. One of the most focused upon attributes of any Case is the Time. Given the many ways time may need to be formatted and represented, we quickly realized it would be more efficient to create a separate CaseTime class (figure 4) that could encapsulate all the various formatting and date-time mathematics required for analysis of an OR. The Case class, thus became a composite class with its own behaviors and attributes, one of which being the custom CaseTime class where all data-time math and representation would take place.

The CaseCollection class does not have many attributes, when compared to the Case class, other than a label and a data structure to hold any number of Case class instances. The power of this class comes from the ability to iterate and summate any combination of Case attributes contained in the data structure. The functional programming principle of recursion, allows for the creation of any number of additional CaseCollections, each containing a group of Cases organized by any of the attributes of either the Case or CaseTime objects or list comprehension to complete complex mathematical manipulation of each Case contained by a given case collection. The simplest recursive function is to take a year's worth of cases contained in a CaseCollection and break them into separate collections by surgeon or day of the week, etc. List comprehensions can be as simple as counting the number of cases contained in a given case collection or as complex as a control chart looking at the variability between two or more surgeons for a given procedure (Figure 4). Financial modeling, including revenue generation, staff and other expenses can also be performed using the concept of a case collection along with list comprehension (Figure 5).

The power of this Class becomes apparent as different questions are posed about the surgical schedule over any given time period. Using data extracted from an OR management system for an entire year, it is converted into an array of Case objects, each of which consists of many attributes (Figure 3), one of which being a CaseTime object. Within the CaseCollection object is code allowing for the generation of additional CaseCollections, each containing all of the cases with the same value for any given attribute such as surgical procedure, surgeon, or any number of complex variants you would like to explore (such as all cases beginning between the hours of 6:52 am and 11:47 pm). With this functionality, data can be easily segregated into bins where both simple and complex data summaries and statistics can be generated. For example, all laparoscopic cholecystectomies, for each surgeon over the course of 1 month, 1 year or 5 years can be easily and rapidly created in the framework and then each of these collections can generate descriptive statistical analysis or the plotting of control charts without the need for complex database queries, additional calculations and data manipulation in a spreadsheet followed by the creation of plots in another program. As new data visualizations and algorithms become available, they can be added to the framework allowing for an expanding and more functional way to look at OR data.

The OR management code framework has allowed us to look at distributions of cases presenting to the PACU from a number of different perspectives including day of the week, time of the day and given total case volumes and duration for any given day. In addition, with the ability to look at the data from different perspectives and by changing how we group and organize the cases, we can create historical data distributions that feed into predictive and machine learning simulations to be applied to future scheduling and financial models. Being able to apply a standard code base to data collected in various databases and between institutions brings reproducible results and quick developmental time lines.

The OR management framework presented in this poster has been in development for the past 6 years. We are continually working on expanding the code to include the ability to assess block utilization, financial performance and efficiency, staff and other resource scheduling and allocation and working to develop simulations and predictive algorithms that can be used in real-time decision making to improve the performance of an expensive resource while optimizing the surgeon and patient experience. Our goal is to apply this framework to validate hypothesis about OR performance and uncover areas where improvements can be made so as to realize gains in efficiency and financial indices.

Conclusions:

The OR management framework presented in this poster has been in development over the past 6 years and remains in continual development. We are working on expanding the code base to include the ability to assess block utilization, financial performance, efficiency, staff and other resource scheduling and allocation and working to develop simulations and predictive algorithms that can be used in real- time decision making to improve the performance of an expensive resource while optimizing the surgeon and patient experience.

We are currently using the framework for researching financial outcomes under different scheduling scenarios. We have also started working on some predictive analytics to assess optimization of case sequencing and the impact on flow into PACU. Finally, we have began to work on using embedded computers to collect and process real- time information such as anesthesiology gas flow to collect volatile anesthetic costs in an effort to make anesthesiology providers aware of these issues and drive change where appropriate.

Our goal is apply this framework to validate hypothesis about OR performance and uncover areas where improvements can be made so as to realize gains in efficiency and financial indices. A great deal of information is hidden within the data being collected in the perioperative period. Development of the tools to uncover and present such information brings potential to building a data-responsive operating room.

References:

1. The Python Quants Group (http://www.pythonquants.com/)
2. Python Software Foundation (https://www.python.org/)
3. iPython Interactive Computing (http://ipython.org/index.html)
4. Pandas python data analysis library (http://pandas.pydata.org/)