Wazee Group
Software Construction & Renovation

Business Process Optimization Domain Analysis

Version 1.0

Jim Ladd
Wazee Group, Inc.

May 26, 2022

Contents

Wazee Group
Software Construction & Renovation

Introduction

The paper presents Business Process Optimization (BPO) as the next phase in the evolution of process automation technologies. A definition is presented along with a domain analysis that provides a path forward in the realm of process automation.

Background

Process automation concepts have a long and extensive role in the history of software applications. Control of a process began when the first program was written and expanded as programs became applications and then distributed systems. As the business processes using these systems evolved so did the underlying software to help automate these processes.

A pyramid diagram is shown below that depicts the relative popularity and capabilities of the types of process control archetypes.

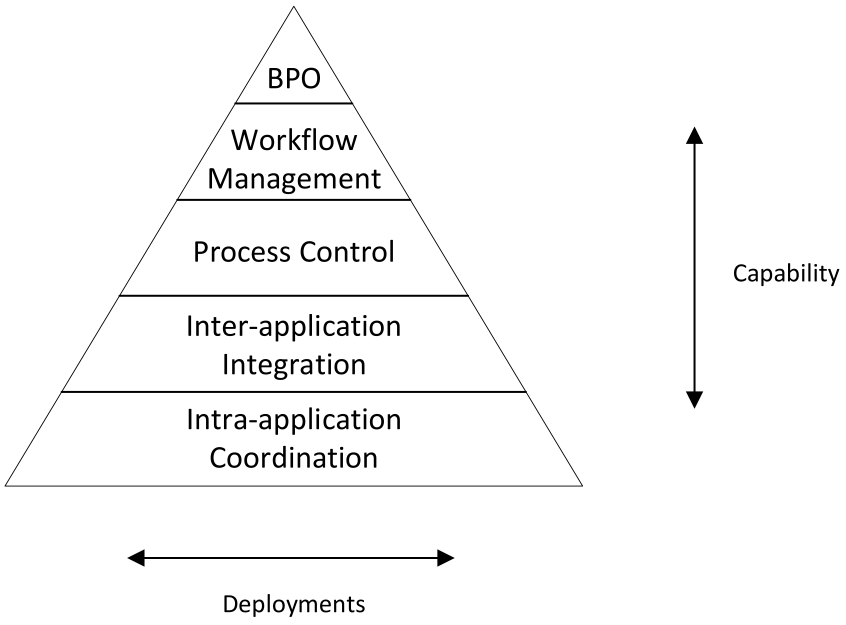

Figure 1 - Process Control Approaches

The five broad classifications of process control are described below in a general chronological sequence.

- ***Intra-application Coordination*** – The most basic and intrinsic method of process control, this classification deals with the internal design of a software system. The code is typically sequential in nature and is focus primarily on the entities within the application and those human operators with direct access. Interactions with different systems are through manual entry of data and generating reports for manual entry into those other systems.

- ***Inter-application Integration*** – This classification covers point-to-point interfaces between different systems. The interface can be 1) proprietary to an application, 2) based on open standards or practices, or 3) a third party solution. The key attribute is that there is no logic nor processing as the information flows from source to destination. This type of integration is basically a messaging layer.

- ***Process Control*** – This type has logic between the integration points. The capability of the processing ranges from simple to complex. The solution typically includes a "process controller" entity along with an underlying messaging system.

- ***Workflow Management*** – A workflow management solution not only includes a "process controller" between applications but also include humans in the process automation scenarios. "Tasks" are typically placed in a user's queue and the controller notified when the task has been "completed".

- ***Business Process Optimization*** – A system at this level not only interacts with human and non-human entities but strives to increase the efficiencies of the business process executions. The optimization maximizes the utilizations of the participants in the process executions. Tasks are created and assigned to resources in forecasted schedules. As time passes and deviations in the forecasts emerge, the tasks are dynamically reassigned in updated schedules.

Approach

The path to a capable and useful BPO system is not easy, quick, or for the inexperience. However, all journeys have a beginning and this one starts with a two part mantra:

- *Everything is an **object**...with a **lifecycle**.*

- ***Events** are everywhere, all the time.*

The Objects, Lifecycles, and Events (OLE) approach provide the guidance from concept to code. In this paradigm, the challenge for software architects is to identify the critical objects, to model the relevant parts of their lifecycles, and to determine the events that impact changes of state.

When the OLE paradigm is embraced, a modeling technology quickly emerges that facilitates the effort. Finite state models (FSM) are a natural choice to obtain a deep understanding of the problem space and provide process definitions that can be executed by a FSM engine in the solution space.

Finite State Models

Finite state models (FSM) have been used for decades in a wide spectrum of industries. Domains using finite state model technology include communication systems, automobiles, avionics systems, and man-machine interfaces. These problem domains share common characteristics: they are usually large in size, high in complexity, and reactive in nature. A primary challenge of these domains is the difficulty of describing reactive behavior in ways that are clear, concise, and complete while at the same time formal and rigorous.

Finite state models provide a way to describe large, complex systems. FSMs view these systems as a set of inbound and outbound events, conditions, and actions integrated together to understand, model, and manage the change of state within the application. Some FSM technologies also provides a set of rules for translating the problem artifacts or specifications into source code. The partitioning of the problem into the events, conditions, and actions, the structured processing environment, and the ease of expressing the processing logic are the foremost strengths of FSMs.

Several different FSM patterns and archetypes have been researched and developed over the years. From the simplistic approach of a multi-value system variable to extremely sophisticated frameworks incorporating concurrency, hierarchies, persistency, and historical information. The pattern chosen for this project has been used successfully on past commercial systems. It is a straightforward but capable model. The fundamental components of our finite state model include:

- ***State*** represents the "mode of being" the object at any given time. An instance of the object being modeled may change state via Transitions. A Transition may be traversed when a specific event occurs and a set of associated conditions are met.

- ***Transition*** describes a single pathway from a given state to another state. The set of all transitions describe all possible paths among the defined states. A transition contains an event that it is subscribing to a condition and an action. A transition also contains the state from which the transition is exiting (i.e., the "from" state) and the state to which the transition is entering (i.e., the "to" state).

- ***Event*** is the mechanism that the system uses to interact with external systems and with itself.

- ***Condition*** represents a set of logic that evaluates to a Boolean result. It is used to determine if a transition is to be activated.

- ***Action*** is the processing to be performed with a transition is activated.

A diagram of the major components of the FSM is shown below. This is a subset of the lifecycle model for a countdown timer.

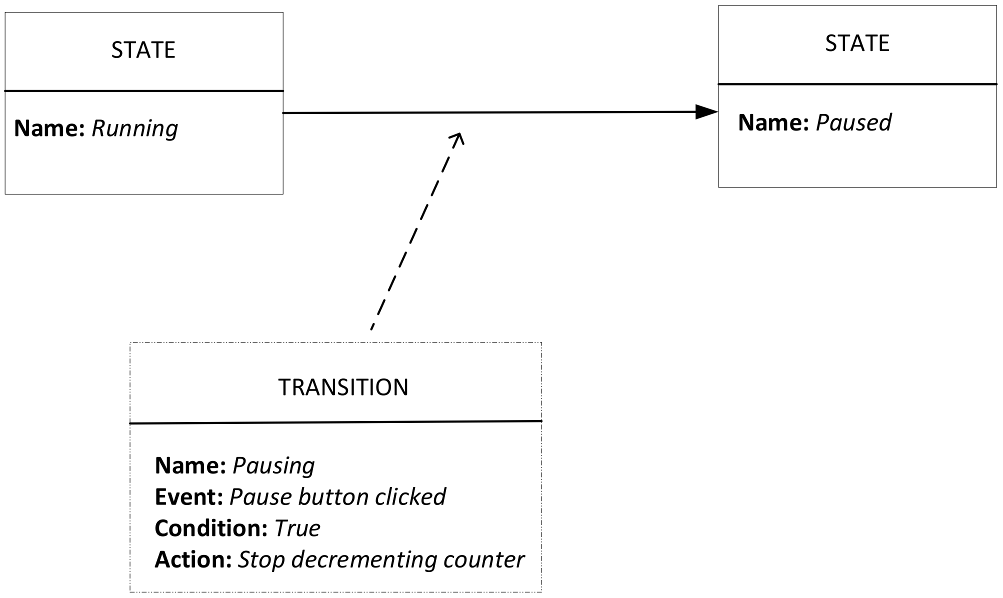

Figure 2 – Key elements of the finite state model

One of the more puzzling topics covered during the learning process of finite state models concerns events. What exactly is an event? How are they created? What's *their* lifecycle? Events can be considered as instances of trigger mechanisms that initiate code execution. In most problem domains, event generation can originate from four sources:

- ***User Input*** – Events can be created when a human interacts with a user interface. Mouse clicks, key strokes, etc., can all generate events that triggers processing.

- ***External Systems*** – Messages, responses, signals, etc. from external systems can also create events that may be relevant to the finite state models.

- ***Passage of Time*** - The simple act of time passing can generate events. The FSM may represent the passage of time in various ways.

- ***Change of Internal State*** – The occurrence of an FSM instance achieving or "entering" a new state may warrant a new event to be generated.

Domain Analysis

The first step of domain analysis is to define the boundary of the problem space. This activity identities the entities that are external to the BPO space and the subject matter that lies within. After the boundary is set, the problem space can be analyzed and partitioned into core domains and common services. A diagram of the boundary is shown below:

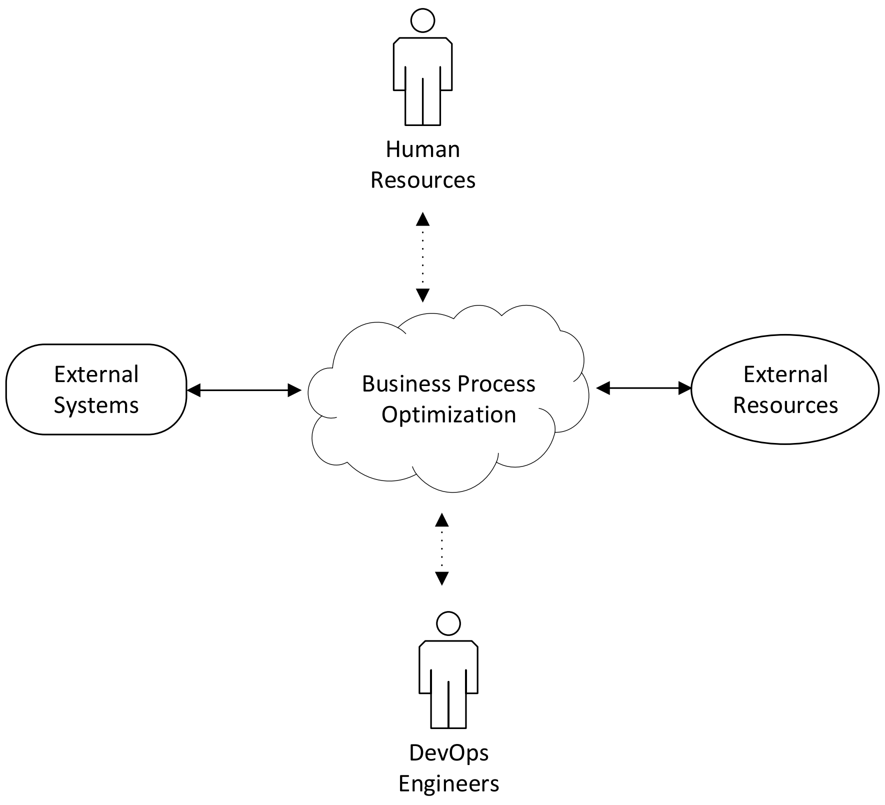

Figure 3 – Boundary of the BPO problem space

The external entities of the BPO problem space include:

- ***External Systems*** – This group consists of systems that interface with the BPO system. Events are received from and sent to external systems. While the interactions can range from simple to complex, the External Systems are considered to be loosely coupled in the process execution.

- ***External Resources*** – This set of resources are systems, machines, etc. that can be managed by the BPO system. Unlike External Systems, these entities can be controlled, impacted, or influenced by the BPO solution. External Resources can be assigned tasks and the timeline for task execution.

- ***Human Resources*** – Like External Resources, this group consists of human resources that can be managed or impacted by the BPO system. These actors are considered participants in the process execution.

- ***DevOps Engineers*** – These actors' program, configure, and monitor the BPO system.

The next step in the process is to decompose the problem space into the core domains and common services. Core domains represent the subject matter areas that are unique and critical to the problem space. Common services are the domains that are typical to other problems spaces and whose implementations in the solution space are widely available. A diagram of these core domains is show below:

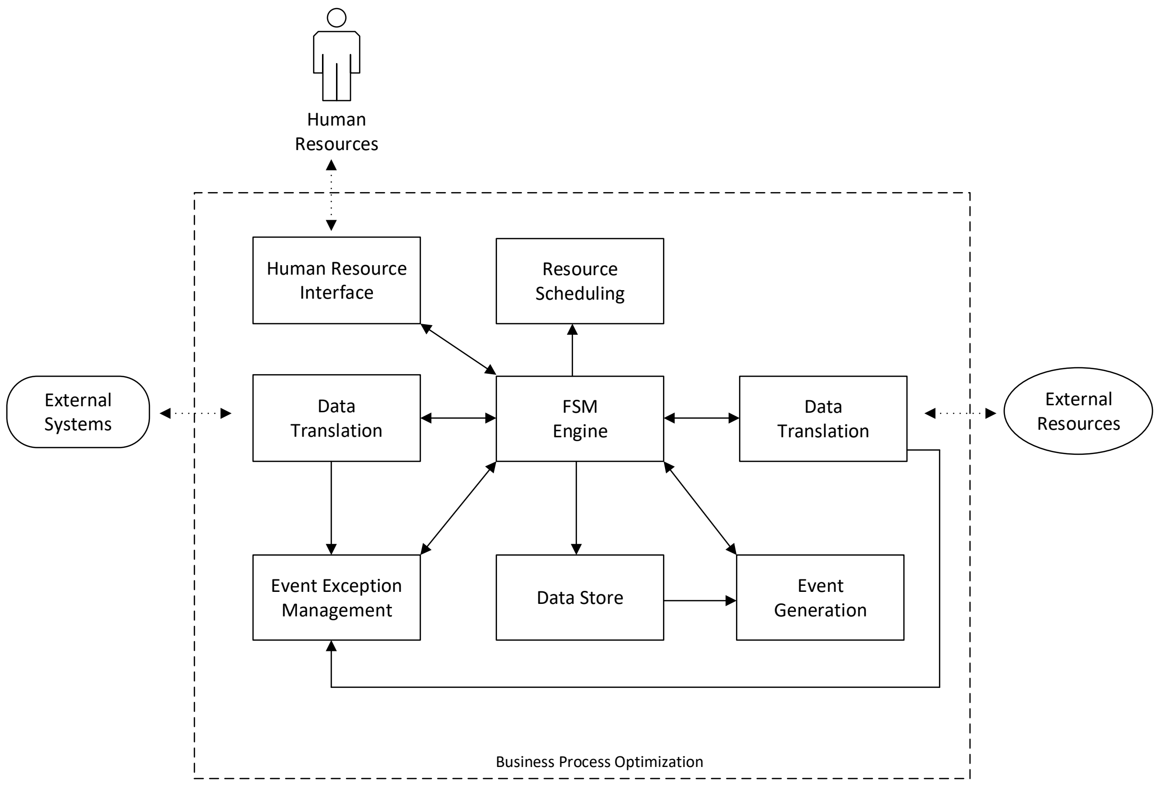

Figure 4 – Core Domains of Business Process Optimization

The core domains of the problem space include:

- ***Finite State Machine Engine*** – This domain is responsible for executing the finite state machine models that reflect the business processes. Events are received, conditions evaluated, actions executed, transitions traversed, and states persisted at the required performance levels.

- ***Resource Scheduling*** – The scheduling domain creates schedule forecasts for the human and non-human resources. Deterministic scheduling algorithms are used when the processing times for the tasks are known or can be estimated with confidence. Probabilistic-based algorithms perform the scheduling when the processing times are unknown or task requirements are vague or missing. The schedules are constantly monitored and can be re-created based on changes in the event horizon.

- ***Event Generation*** – This domain is responsible for creating derived events that help drive the system. The domain receives events from other sources and monitors state changes of objects. It constantly applies rules to the current dataset and, for those rules that are satisfied, publish new events of interest.

- ***Event Exception Management*** **–** The EEM subsystem receives exception events from the other entities in the system. It is responsible for automatically re-sending rejected events when the root cause is transitory. The EEM accepts new types of failures and determines whether the issue is transitory or permanent. If the error requires human intervention, it presents the error with supporting information for quick resolution and re-sending if needed.

- ***Data Translation*** **–** This domain maps inbound data (i.e., native) to a canonical representation for processing by the BPO subsystems. Likewise, the outbound data is converted from the canonical view to a native representation that can be consumed by the target external system.

- ***Data Store*** **–** The Data Store subsystem is responsible for fast, scalable persistency of the operational data for the core domains.

- ***Human Resource Interface*** **–** This domain provides the interface for human-based resources. The interface allows users to easily initiate and complete their tasks in an intuitive fashion.

The common services of the problem space include:

- ***Monitoring Service*** – This service gathers operational data from the different subsystems and provides both retroactive and proactive monitoring. When scenarios arise that require human attention, the service will notify the appropriate parties.

- ***Notification Service*** – The Notification Service provides mechanisms to send messages to people and machines.

- ***Messaging Service*** – This service provides asynchronous communications capabilities with point-to-point and publish/subscribe modes.

- ***Persistence Service*** – The Persistence Service is responsible for maintaining data throughout the lifespan of the need. This service covers a wide spectrum of data storage technologies.

- ***Secrets Service*** – This service protects data from unwanted use. It holds credentials to assets and is targeted to specific environments.

Looking Back and Moving Forward

The core concept for a BPO system has been around since the late 1990's when the Enterprise Application Integrations (EAI) tools emerged. Back then, I envisioned a next level system comprising of the messaging and process automation features of the EAI tools, the human task management capabilities found in the document management systems of the day, and the resource scheduling algorithms that emerged in the manufacturing industries in the late 1980's.

In 2002, I submitted a proposal to a client for building such a system. While the client was excited about the vision, the solution was simply too much for the near-term requirements. An alternative solution was created using Wazee Group's xBroker and Legacy Navigator tools.

Over the years, I have floated the concept to clients in various industries. While the interest was there, the business opportunity never materialized. Recently, however, I have seen the business processes in these industries become both more numerous and more complex with the need for automation and optimization drastically increasing. Now may be the perfect time to move forward with this technology.