Business Process Improvement Using Object-Process Methodology

Jason M. Casebolt
System Design and Management Fellow
MIT School of Engineering
Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
jason.casebolt@sloan.mit.edu

Dov Dori
Faculty of Industrial & Management Engineering
Technion – Israel Institute of Technology
Haifa 32000, Israel
and
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
dori@ie.technion.ac.il

***Abstract*—For decades, business process improvement (BPI) has been a persistent and expensive concern that spans across many industry sectors. We present OPM-BPI – a model-based method to improve business processes using Object-Process Methodology (OPM), the new ISO 19450. An aviation manufacturing company case study demonstrates the method and shows how a significant portion of the supporting objects and processes can be eliminated or merged, achieving significant model simplification.**

Keywords – Business Process Improvement; Manufacturing Processes; Model-Based Systems Engineering; Object-Process Methodology; Process-as-a-Product; Process Architecture; Process Improvement

I. INTRODUCTION

Emerging as a major focus in the 1980's and 1990's, Business Process Improvement (BPI) has been a steady undertaking for major companies around the globe [1] [2]. The definition of BPI used here is simply "improvement of a process [by] means [of] changing a process to make it more effective, efficient, and adaptable" [1]. The leading drivers of this movement have been the need to save money and to improve performance. Additional motivations include increasing customer satisfaction, improving organizational responsiveness, complying with regulations, such as Sarbanes-Oxley, and major events, like a merger or an acquisition [2]. These efforts have made BPI a big business, with process improvement departments, consultants, and practitioners who focus a large part of their time or resources on improving business processes. They use familiar products and methods, such as Lean, Six Sigma, Business Process Reengineering, Workflow, ERP software, and Business Process Management Suite software [2].

Despite over 20 years of focus, companies are still spending substantial sums on process improvement each year. For example, a 2013 survey of over 300 large companies revealed that 46% spent at least $500,000 that year on process improvement efforts. Nearly half of those companies (26% of the overall total) spent at least $1 million [2]. Of all companies surveyed, 31% classified BPI as a major strategic commitment.

The purpose of this paper is to apply "systems thinking" by using Model-Based Systems Engineering (MBSE), specifically Object Process Methodology (OPM), to perform a new method of BPI, called OPM-BPI. This approach applies MBSE to identify solution-neutral process improvement opportunities in a manner that accounts for the context of the system.

The rest of this paper is organized as follows: Section II introduces OPM. In Section III we describe OPM-BPI and apply it in Section IV to a real-life case study at a large manufacturing company. In section V we summarize the results and discuss ongoing research efforts.

II. OBJECT-PROCESS METHODOLOGY

OPM is a leading MBSE platform [4] due, in part, to its December 15, 2015 release by the International Organization for Standardization (ISO) as the ISO-19450 specification for "Automation Systems and Integration – Object-Process Methodology" [5]. Founded on the minimal ontology of stateful objects and processes that transform them as a set of necessary and sufficient building blocks, OPM is a holistic conceptual modeling language and cross-system lifecycle methodology, expressed graphically in a single kind of diagram and a complementary, auto-generated natural language text. It is different from other MBSE modeling languages in (i) the equal priority given to stateful objects and processes as the only two conceptual building blocks needed to represent systems in any domain – the minimal ontology, and (ii) the bimodal representation of the OPM model in both formal intuitive graphics and automatically generated text – simples sentences in a subset of English [4].

OPM is flexible in its application and has indeed been applied in a wide array of industrial domains, from defense and avionics through electronic consumer appliances to software engineering, Web applications design, and molecular biology. OPM has been used in the evaluation of complex socio-technical system in fields such as aerospace, defense, information systems, medicine, sciences, and space exploration [6]. Formal yet intuitive, OPM is learned quickly and enables involving the customer as a partner, starting from the early

product or system development phases all the way to deployment and maintenance, providing for the integration of risk and interoperability into the architecture and design of complex systems and systems-of-systems.

Using OPM to Create Models

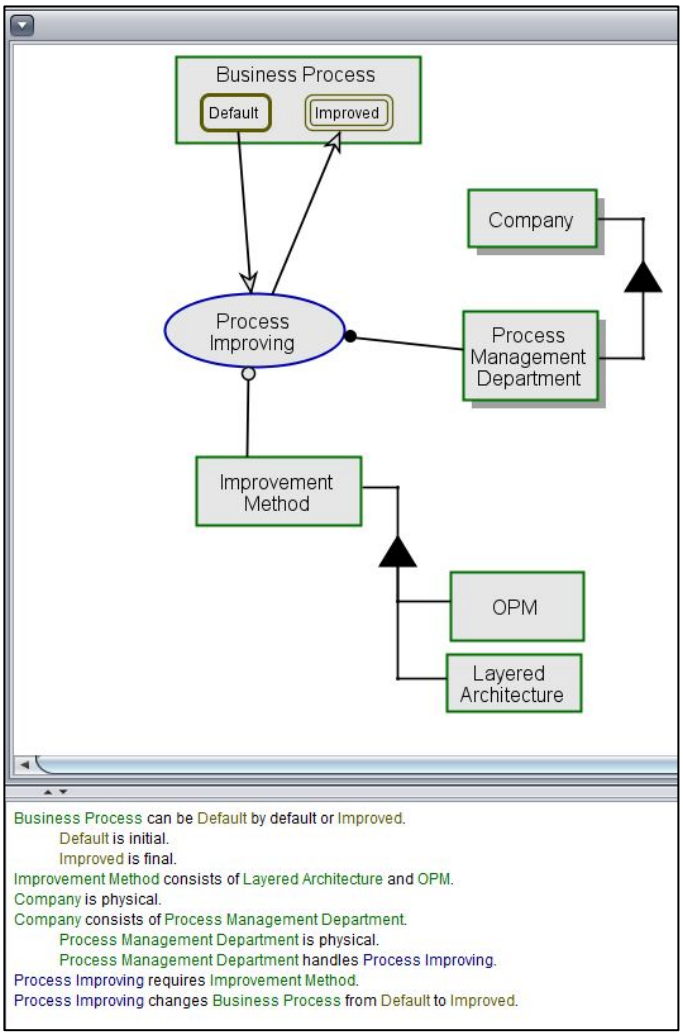

Figure 1. An OPM model of Process Improvement with the OPM-BPI method

To use OPM, the freely available CASE tool OPCAT[1] provides an environment that enables users to design OPM models, which are referred to as Object-Process Diagrams (OPDs) [7]. OPDs created in OPCAT automatically generate Object-Process Language (OPL) text in a separate panel, which is a textual description of the OPD in a subset of English. In addition to model creation, OPCAT enables model simulation through executing the model for behavior verification and validation. Figure 1 is a simple OPM model of OPM-BPI: Process Improving using OPM as a method—the focus of this paper—through OPCAT's OPD (top) and OPL (bottom) views.

[1] Downloadable from http://esml.iem.technion.ac.il/

	Visual Representation	Textual Form	Definition	Description
Entities	Object	Nouns; capitalized first letter in every word; if ending with "ing", "Object" is placed as a suffix	*An **object** is a thing that has the potential of stable, unconditional physical or mental existence.*	Static things. Can be changed only by processes.
	Process(ing)	Nouns in gerund form; capitalized first letter in every word; if not ending with "ing", "Process" is placed as a suffix	*A **process** is a pattern of transformation that an object undergoes.*	Dynamic things. Are recognizable by the changes they cause to objects.
	Object state	Nouns, adjectives or adverbs; non-capitalized	*A **state** is a situation an object can be at.*	States describe objects. They are attributes of objects. Processes can change an object's state.

Figure 2. The OPM entities (adapted from [4])

Within OPM, a system is comprised of physical (tangible) or informatical (intangible) things—objects and processes—that are represented by rectangles and ovals respectively [7], as presented in Figure 2. A key premise of OPM is that objects and processes are of equal importance and complement each other for providing a complete structural and procedural specification of the system [8]. Objects are things that exist in some state, and are represented by nouns. Processes, represented by verbs, preferably in their gerund form (ending with "ing"), are things that transform objects through creating or destroying objects, or changing object states.

Shorthand Name	Aggregation	Exhibition	Generalization	Instantiation
Symbol	▲	△	△	▲
Meaning	Relates a whole to its parts	Relates an exhibitor to its attributes	Relates a general thing to its specializations	Relates a class of things to its instances

Figure 3. Structural relation symbols (adapted from [4])

To supplement the objects, processes, and states, OPM supports structural and procedural relations, expressed graphically as links, as well as hierarchical organization for complexity management. The four fundamental structural links, represented and defined in Figure 3, are aggregation-participation, generalization-specialization, exhibition-characterization, and classification-instantiation.

While structural links connect objects to objects or processes to processes, procedural links connect processes to objects or to object states. Procedural links include transforming links (consumption, result, input-output, and effect), enabling links (agent and instrument), and control links (which are out of scope for this paper). Consumption implies that the process consumes the object. Result links indicate that the process generates the object. An input-output link pair denotes that the process changes an object from an input state to an output state. The effect link denotes that the process changes the object without specifying the input and output states. These are demonstrated in Figure 4 [4].

Type	Name	Semantics	Symbol	Source	Destination
Transforming	Consumption	The process consumes the object.	→	Object	Process
	Result	The process generates the object.		Process	Object
	Input[a]	The process changes from an input state.		state	Process
	Output[a]	The process changes to an output state.		Process	state
	Effect	The process changes the object.	↔	Object	Process
				(Both are source and destination.)	
Enabling	Agent	The human agent enables the process.	—●	Object	Process
	Instrument	The process requires the instrument.	—○	Object	Process

Figure 4. Procedural link (adapted from [4])

Enabling links, also presented in Figure 4, denote objects that are needed for the process to occur but themselves are not transformed. The agent link expressed the fact that the agent (a human) enables the process. An instrument link denotes a non-human enabler.

As noted, beyond visualization, OPCAT generates OPL to evaluate the system through textual description in English [9]. OPL has two purposes. First, it enables domain experts and systems architects to better analyze and design a system by providing a description-based model to validate or contrast their graphic-based OPD model [10]. Second, OPL establishes a firm basis for automatically generating the designed application. An OPL example is displayed in the bottom portion of Figure 1.

OPM Summary

OPM is a dual approach that uses graphic-based modeling with text-based validation to construct a system. Through the freely available OPCAT software and the minimal number of selectable entities, OPM is easy to obtain, learn, and use. Despite its simplicity, it enables robust system exploration beyond architecture, including states, aggregation, and zooming within systems-of-systems. Its recent emergence as an international standard provides for its use as a consistent method for the foreseeable future.

III. The OPM-BPI Method

To perform the OPM-BPI method, the design of the business process must be (i) decomposed, (ii) rationalized, and (iii) optimized. Modern systems architectural principles provide a basis from which OPM can be used for decomposition and rationalization [3]. This paper expands on these principles by providing a means for optimization or at least significant improvement of business processes.

1. Decomposition

The first step is to decompose the design into its entities so that it can be evaluated. Using OPM, each entity of the design is identified as either an object or a process [4]. The focus of the first step should be accuracy of the identification, not the relationships between the objects and processes; relationship association will take place in the next step.

2. Rationalization

The second step is to rationalize the entities that were identified in Step 1, namely, to express meaningful and useful relations among them. With OPM, this involves connecting the objects and processes that were identified with structural and procedural relations. Modern system architecting principles also provide the basis for this linkage [3].

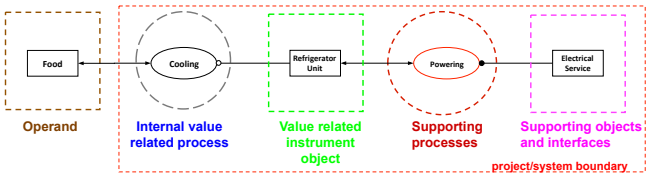

Figure 5. Layered architecture (adapted from [3, 15])

The concept of layered architecting within OPM is the starting point of performing BPI. Following this approach, we identify the system's objects and processes, and separate them into the operand object – the major object transformed by the system, value-related objects and processes, and finally supporting processes and objects [3]. Figure 5 provides an example of this rationalization approach, resulting in a layered architecture. This approach rationalizes not just the relationships, but also the value-added role that each object and process plays in the context of the system's intended function.

3. Optimization

After rationalization is complete, the OPM-BPI method takes a different point of view than the modern systems architectural principles. Where Crawley *et al*. suggest that supporting objects and processes provide structure that enables the value-related objects and processes to perform their respective functions [3], our OPM-BPI method asserts that the supporting objects and processes serve as both waste and complexity to a process being performed. The concept visualized in Figure 6 proposes that the operand, as well as the value-related objects and processes, are considered to be value-adding, and are therefore desired. The non-value-added waste that exists as the supporting objects and processes should be minimized or eliminated to maintain efficiency.

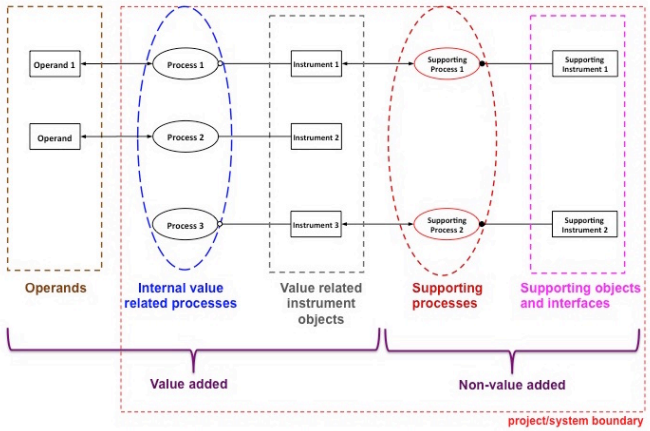

Figure 6. OPM-BPI-based layered architecture (adapted and modified from [3, 15])

The concept underlying this view is that the additional layers of architecture in a business or industrial process serve to complicate, add time, and otherwise hinder a process to perform its pure function. This departs significantly from product development, where such additional structures serve to support the system by design. The key is properly identifying value-adding entities as those that if removed would degrade or otherwise prevent the intended function of the business process from occurring. The remaining entities, those that do not meet this standard, are therefore considered non-value-adding.

Once the entities have been rationalized into value-adding and non-value-adding objects and processes, then optimization occurs through any combination of one or more of the following solution-neutral actions: (i) delete, (ii) combine, (iii) reduce/simplify, (iv) automate, (v) offload/outsource, and (vi) upgrade. These are evaluated for each individual entity, as well as groupings of entities together. Similar to other guiding concepts, such as TRIZ, this OPM-BPI method narrows the focus of a user's creativity to improving certain aspects of a process with specific types of optimization actions.

The non-value-adding objects and processes should be the major focal point, since their removal or simplification does not disrupt the functioning of the business process. Therefore, these things are potentially waste, so their deletion is preferred over other optimization options, since waste eliminated is preferred over waste reduced. Value-added objects and value-adding processes should be reviewed as well, but with an opposite intent, because deletion of a value-added thing undermines the proper functioning of the business process under design. In these cases, simplification activities, such as automation, are preferred over deletion, which, for value-adding objects and processes, is harmful almost by definition.

The result of the OPM-BPI method is an identified set of solution-neutral process improvements that optimize the system and preserve intended function. The solution-specific means of implementing the improvements should be determined by the expertise and resources available at a company using the method. Therefore, OPM-BPI does not provide solution-specific improvements. Rather, it identifies solution-neutral opportunities to generate such improvements.

IV. The AvAe Quality Assurance Case Study

Introduction

We demonstrate the OPM-BPI method through a case study, in which we had access to a large American aerospace manufacturing company identified with the pseudonym "Aviator Aerospace" ("AvAe"). In this case study we demonstrate the OPM-BPI method on a major quality assurance process of AvAe. The case study evaluates and validates a combination of concepts, including those that AvAe has already identified, along with new conclusions identified through applying the OPM-BPI method.

Background

AvAe is a manufacturing company that produces aerospace parts and assemblies in accordance with government quality system regulations [12] and aerospace industry standards [13]. Consistent with best practices [11], AvAe divides its internal control documentation into policies, procedures, and processes that drive its operations. AvAe has a nine-person work group that focuses just on management of quality assurance processes that Quality Assurance Inspectors and Factory Mechanics use to perform their work. BPI and process optimization are among the responsibilities of this quality process management group.

This case study applies the OPM-BPI method to a business process improvement for the "Uninstall Part or Assembly process" (shortly, the Uninstall Process), which was selected due to chronic audit failures, long performance time, and feedback from frustrated process users who claim that the process is too complicated. According to working team time trials, the Uninstall Process takes on average approximately 84 minutes to perform. The process involves 14 written steps, featuring frequent exchanges between Factory Mechanics and Quality Assurance Inspectors at different intervals.

The 14 steps in the Uninstall Process are as follows:

1. Either the Factory Mechanic or the Quality Assurance Inspector initiates both the Uninstall Record and Uninstall Order to begin the process.
2. The Factory Mechanic makes a request to the Quality Assurance Inspector for authorization to uninstall the part or assembly.
3. The Quality Assurance Inspector authorizes the Factory Mechanic's request for authorization.
4. The Factory Mechanic uninstalls the part or assembly.
5. The Factory Mechanic makes a request for authorization to the Quality Assurance Inspector to reinstall the part or assembly.
6. The Quality Assurance Inspector authorizes the Factory Mechanic's request for authorization.
7. The Factory Mechanic reinstalls the part or assembly.
8. The Quality Assurance Inspector verifies that the part or assembly reinstallation was performed correctly.
9. The Factory Mechanic and the Quality Assurance Inspector determine if a retest of the reinstalled part or assembly is necessary.
10. If necessary, the Factory Mechanic and the Quality Assurance Inspector retest the reinstalled party or assembly.
11. If necessary, the Quality Assurance Inspector verifies that the retest was performed correctly.
12. The Federal Aviation Administration ("FAA") Coordinator inspects the reinstalled part or assembly.
13. The Quality Assurance Inspector completes the Order.
14. The Factory Mechanic completes the Record to end the process.

Wait times between steps are noted as one role triggers another role to queue up to begin their next step.

Applying the OPM-BPI Method

Having described the details of the Uninstall Process, we apply the OPM-BPI method to this process.

Step 1: Decomposition

The decomposition of the entities (things, i.e., objects and processes) of the Uninstall Process is straightforward, because the 14 process steps were already identified, along with inputs and outputs for each step, the performers of each step, and the systems used by the performers. Such information is considered best practice [11] to include in business process documentation. In addition to objects and processes being identified in business process documentation, an object's beginning and end states are identified if they are changed through performing the process.

Each of the 14 steps listed in the Uninstall Process document were converted into separate processes in the OPM model. To help relate the written process steps, each process in the OPM model was numbered with the respective process step number. The workers and systems that perform or affect the process, specifically the (i) **Quality Assurance Inspector**, (ii) **Factory Mechanic**, (iii) **FAA Coordinator**, (iv) **Manufacturing Data System**, and (v) the **Requirements Data System**, are represented in the OPM model as objects. Lastly, state changes were modeled with careful attention to the orders and records being opened and closed. For example, in Figure 7, "**installed**" is both the initial and final state of **Part**. Step 4, **Uninstall Performing**, changes **Part** from "**installed**" to "**uninstalled**", and Step 7, **Uninstall Performing**, does the opposite.

Step 2: Rationalization

After the decomposition, the next step is to rationalize those entities into layered architecture. Figure 7 demonstrates the primary value-creation function of the process by selecting and grouping respective entities into operands—essential objects that the system transforms, thereby adding value (on the left), internal value-related processes to the left of the operands, value-related instrument objects next, then supporting processes, and finally auxiliary objects. The classification of these things was based on reviewing the process documentation to verify that the intended output is the reinstallation of an uninstalled part or assembly. Stated differently, that the state of the part or assembly changes from **installed** to **uninstalled**, and back to **installed**.

The selection of the internal value-related processes was more subjective from the point of view of the process improvement architect. The concept was to identify which of the process steps, if removed, would undermine the intended-function of the documented process. The ones selected as value-adding processes were: 1) **Order and Record Initiating**, 4) **Uninstall Performing**, 7) **Reinstall Performing**, 10) **Retest Performing (If Required)**, 12) **FAA Conformity Inspecting**, 13) **Order Completing**, and 14) **Record Accepting**.

By description, steps 10) **Retest Performing (If Required)** and 12) **FAA Conformity Inspection** may seem to be non-value-adding processes, since they are by definition either rework or verification. Still, we decided to list these as value-adding processes since AvAe classifies them within quality assurance processes, which therefore consider them value-adding, though from a manufacturing perspective such activities could be considered non-value-adding. The decision not to list steps 10 **Retest Performing (If Required)** and 12 **FAA Conformity Inspecting** as supporting processes, unlike the other verification-type processes, is that they are imposed by auditing entities [12, 13], and therefore are value-adding to the extent that they satisfy mandatory external constraints.

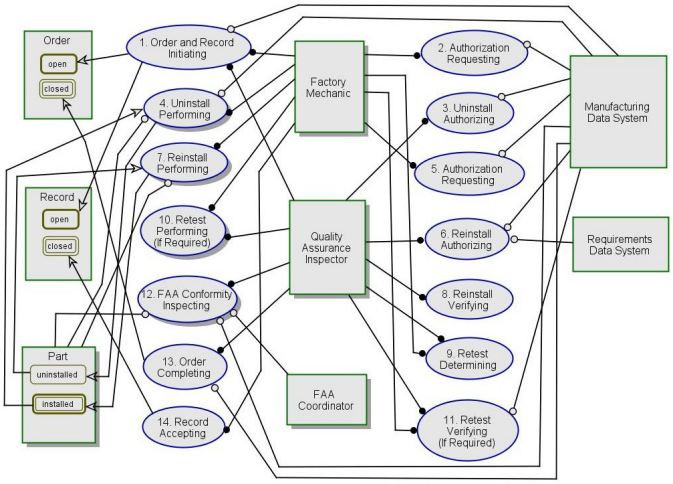

Figure 7. OPM Layered Architecture of Uninstall Process

The value-related instrument and agent objects (middle column) were then listed as those that either perform or are essential to function. This was straightforward from the process documentation that lists the performers as the OPM agents (humans) **Factory Mechanic**, **Quality Assurance Inspector**, and **FAA Coordinator**. The less straightforward one is the **Record**, which may seem non-essential, but is also a requirement-constraint imposed by external sources [12, 13].

Supporting processes and supporting objects/interfaces are those entities that, if they were simply deleted, would not disrupt the intended function of the documented process. For processes, these are all the remaining *requesting*, *authorizing*, *verifying*, and *determining* steps. These are differentiated from internal value-related processes because these steps are not imposed requirements that must be satisfied. The supporting objects/interfaces similarly assist the process, but would undermine the intended function if they were deleted.

Step 3: Optimization

For optimization to take place, we evaluated the non-value-added columns (fourth and fifth columns) to determine if the entities, individually or collectively, could be reduced or deleted. This means that the manufacturing and requirements systems, as well as the *requesting*, *authorizing*, *verifying*, and *determining* steps, were targeted for (i) deletion, (ii) combination, (iii) reduction or simplification, (iv) automation, (v) offload or outsource, and/or (vi) upgrade. Figure 8 demonstrates the three objects and five processes targeted to be optimized as pink (darker than the rest).

Optimization 1: Current AvAe Proposal – Delete Inspection by Quality Assurance Inspector (Supporting Processes): The improvement of Operator Self-Inspection ("OSI") has been

considered by AvAe leadership. OSI has been around for a few decades [14], but still has not been fully deployed into manufacturing companies like AvAe. This approach shifts responsibility of quality inspection from the **Quality Assurance Inspector** to the operator of the process, which in the case study is the AvAe **Factory Mechanic**. **Quality Assurance Inspectors** then perform a separate external function of monitoring the certification of the self-inspecting operators. One of the goals of the OSI is to eliminate the need for the operator to stop and wait for an inspector to come and inspect the product. Here, the incorporation of OSI would result in deletion of non-value-adding Step 8 **Reinstall Verifying**, where the **Quality Assurance Inspector** would otherwise inspect the re-**installed Part** or assembly, and instead merge that inspection back into Step 7 **Reinstall Performing** for the **Factory Mechanic** to perform during the reinstallation. This could be alternatively viewed as (i) deletion, (ii) combination, or (iii) reduction or simplification, depending on the perspective of the process architect.

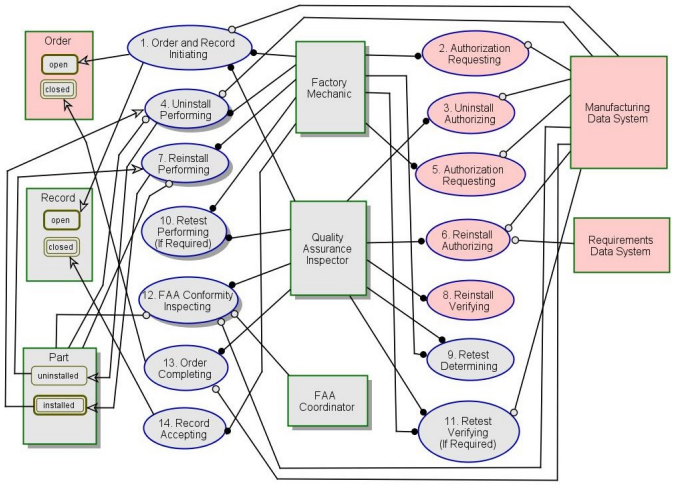

Figure 8. OPM of Targeted Improvements Highlighted

Optimization 2: Current AvAe Proposal – Delete Authorizations Conducted by Factory Mechanics (Supporting Processes): Proposed after considering OSI, the concept of Operator Self-Authorization (OSA) has been identified by AvAe leadership as a potential inverse to OSI. Here, **Factory Mechanics** would self-authorize themselves to perform the uninstallation, which would therefore eliminate the need for Step 2 **Authorization Requesting** (where the **Factory Mechanic** requests authority from the **Quality Assurance Inspector** to uninstall the part or assembly), Step 3 **Uninstall Authorizing** (where the **Quality Assurance Inspector** authorizes the request), Step 5 **Authorization Requesting** (where the **Factory Mechanic** requests authority from the **Quality Assurance Inspector** to reinstall the part or assembly), and Step 6 **Reinstall Authorizing** (where the **Factory Mechanic** requests authorization to reinstall the part or assembly). OSA would therefore result in three non-value-adding steps removed from the process. Similar to OSI, the perspective of the process architect will determine how the targeted improvement is classified.

Optimization 3: Current AvAe Proposal – Combined or Automated Data Systems (Supporting Objects): Another improvement being considered is not specific to a process entity. Rather, it is attributed to simplification of object entities. AvAe uses two different data systems to manage manufacturing and requirement data, **Manufacturing Data System** and **Requirements Data System** respectively. These systems both require manual input each time that information is accessed. While these systems are important, they are classified as non-value-added because the systems can be deleted without disrupting process function; though an alternative for accessing data to accomplish value-adding steps needs to be addressed.

AvAe is currently studying its requirements for a next generation data system. The OPM-BPI method presents a visual platform for which AvAe can model what types of requirements would also improve process simplification. Here, following the OPI-BPM method of finding a solution-neutral optimization, both systems could be (ii) combined into a single data system to reduce the architecture even further. Depending on preferences of the process architect, this could take the form of (ii) combination, or (i) deletion of one data system and (vi) upgrade of the other for the same requirements. In addition, other opportunities could exist for automating sub-processes to moderate inputs-outputs of the data system to increase efficiency further and eliminate waiting on manual inputs.

Optimization 4: Proposal by this Paper's Authors – Combining Order into Record (Primary Operand): Opening and closing both an **Order** and a **Record** are currently performed for two different purposes. **Order** signals work to be performed, while **Record** maintains configuration control, as required by regulations and standards [13, 14]. Though practical use varies, the conceptual usage of both these informatical objects is redundant when displayed through OPM. Therefore using the OPM-BPI method, we have identified a solution-neutral (ii) combination of the **Order** and **Record** to eliminate this redundancy. A solution-specific manner of performing this (ii) combination can now be explored by AvAe experts for the feasibility and specific means of implementation.

Optimization 5: Proposal by this Paper – Simplifying Order and Record Initiating into Record Initiating (Value Process): One effect of Optimization 4 above is that another optimization occurs: (iv) reducing/simplifying the Step 1 **Order and Record Initiating** from initiating both the **Order** and the **Record** to initiating only the **Record**.

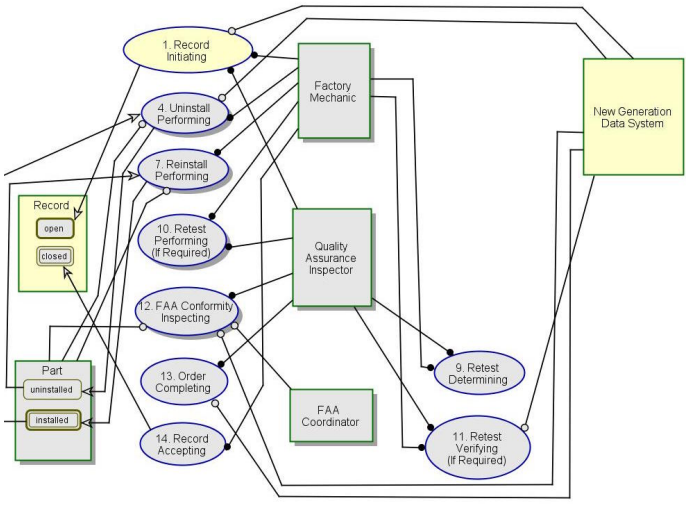

Figure 9. OPM of Optimized Uninstall Process

OPM-BPI Results in AvAe Practical Example

Depending on the choices to be selected by AvAe's senior management, OPM-BPI has identified or validated that up to five non-value-added process steps could be eliminated, four value-added and non-value-added objects could be combined into as little as two, and one value-added process could be simplified. These are identified in Figure 9 by the absence of the pink entities that were present in Figure 8, and highlighting yellow entities that consume the combined or simplified objects. OPM-BPI has identified that these solution-neutral improvements have an optimization effect on the process as a system, without disrupting the value-added function performed by the business process. Therefore, identification and validation of new options has indeed occurred. The next steps are for these solution-neutral opportunities to be explored by AvAe, its process management team, and technical experts to find solution-specific means to implement these solutions that are consistent with process function, resources, and other synergies throughout the company.

V. SUMMARY AND CONCUSIONS

This paper proposes a new method of conducting business process improvement by using the recently released ISO-19450 standard Object-Process Methodology. Through using layered systems architecting principles to create improvement models, the value-added and non-value-added objects and processes within the models are clearly identified. Once identified, creativity-guiding principles are applied to the entities within the model to determine which ones can be deleted or simplified without detriment to intended process function with the purpose of eliminating waste and complexity within the model. This results in solution-neutral business process improvement opportunities that can become the focus of solution-specific optimization efforts. A case study from an aviation manufacturing company was used to illustrate the application of our OPM-BPI method and the optimized results that were obtained.

Future work on the subject should evaluate quantifying the effects of optimization, since not all improvements have the same impact. In addition, there is potential to broaden the reach of the optimization modeling to take into account a wider range of sub-processes that are often evaluated in many lean or kaizen process improvement workshops. Lastly, there is potential opportunity to apply Axiomatic Design principles to ensure that the requirements of the process to be performed are also optimized, thereby reducing the likelihood of sub-optimized functional requirements becoming the basis around which the design parameters of an individual process is formed.

ACKNOWLEDGEMENT

The authors would like to thank Aviator Aerospace, for access and perspective into its business process improvement activities, and the MIT System Design and Management master's program for its support. Second Author would also like to thank Gordon Center for Systems Engineering at the Technion for its support.

REFERENCES

[1] Harrington, H. James. Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity, and Competitiveness. McGraw-Hill. 1991.

[2] Harmon, Paul and Celia Wolf. "The State of the BPM Market – 2014." Business Process Trends. 2014.

[3] Crawley, Edward, Cameron, Bruce, and Daniel Selva. Systems Architecture: Strategy and Product Development for Complex Systems. Prentice Hall. 2015.

[4] Dori, Dov. Object-Process Methodology – A Holistic Systems Paradigm. Berlin: Springer Verlag. 2002.

[5] International Standards Organization. "Automated systems and integration – Object-Process Methodology." ISO/PAS 19450:2015. Web. Accessed: December 13, 2015.

[6] Mordecai, Yaniv and Dov Dori. "Agile modeling of an evolving ballistic missile defense system with Object-Process Methodology." *Proc. SysCon 2015 ,9th Annual IEEE International Systems Conference*, Vancouver, Canada, pp. 839-846, April 13-16, 2015. DOI: 10.1109/SYSCON.2015.7116855

[7] Dori, Dov, Reinhartz-Berger, Iris, and Arnon Sturm. "OPCAT – A Bimodal CASE Tool for Object-Process Based System Development." *Proc. IEEE/ACM 5th International Conference on Enterprise Information Systems (ICEIS 2003)*, Angers, France, pp. 286-291, April 23-26, 2003.

[8] Blekhman, Alex, Dori, Dov, and Richard Martin. "Model-Based Standards Authoring." INCOSE. 2011.

[9] Soderborg, Nathan R., Crawley, Edward F, and Dov Dori. "System Function and Architecture: OPM-Based Definitions and Operational Templates." Association for Computing Machinery – Communications of the ACM Vol. 46, No.10. 2003.

[10] Blekhman, Alex, Wachs, Juan P., and Dov Dori. "Model-Based System Specification with Tesperanto: Readable Text from Formal Graphics." IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2015.

[11] Page, Stephen B. Establishing a System of Policies and Procedures. Process Improvement Publishing. 2002.

[12] U.S. National Archives and Records Administration, "Quality System" 14 CFR Part 21.137, 2012.

[13] SAE Aerospace, "Aerospace Standard AS9100C", 2009.

[14] Whittingham, P.R.B., "Operator Self Inspection." American Society of Quality - 41st Annual Quality Congress. Vol. 41. 278-286. 1986.

[15] Cameron, Bruce. "Functional Architecture and Operations." MIT Course ESD.411: Foundations of System Design and Management I. Lecture. September 23, 2014.