Production Ramp-up of a Hardware Startup: Simulating Build to Order Strategies, Capacity Estimations and Impacts of Batching

by

Alexander Willem Anton van Grootel

B.Eng. Mechanical Engineering with Management
The University of Edinburgh, 2015

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF ENGINEERING IN ADVANCED MANUFACTURING AND DESIGN

AT THE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

SEPTEMBER 2016

© 2016 Alexander Willem Anton van Grootel. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part in any medium now known or hereafter created.

Signature of Author: ____________________________________________________________

Department of Mechanical Engineering
August 10, 2016

Certified by: ________________________________________________________________

Stanley B. Gershwin
Senior Research Scientist
Thesis Supervisor

Accepted by: ________________________________________________________________

Rohan Abeyaratne
Quentin Berg Professor of Mechanics & Chair, Committee of Graduate Students
This page is intentionally left blank.
ABSTRACT

When a startup scales up its enterprise the manufacturing system used to govern the production process should be robust enough to scale up with it. NVBOTS is in a position where it plans to scale up production by an order of magnitude within a few months. NVBOTS’s manufacturing systems should therefore be reevaluated to ensure its design will meet requirements. In order to model the production process at NVBOTS a discrete event manufacturing simulation is developed.

Both a Push system and a Build to Order (BtO) system are initially considered. The trade-offs that exist in these policies are identified and analyzed. A recommendation is made to adopt a BtO policy and to build in a batch size of one. Based on the feedback from NVBOTS an advanced model is then developed, which incorporates some benefit of batching. The effects of batch size on lead time, inventory level and capacity are investigated. Even when assuming a 30% setup time for assembly steps, the optimal batch size with regards to lead time or inventory levels remains at one. However, it is found that batching has a significant impact on the capacity of the system. The advanced model also introduces a new policy named Sell x Build y (SxBy, e.g. S1B6), which allows NVBOTS to operate in larger batches by minimizing the effect on lead time while keeping inventory roughly constant. With the current labor levels, and using a S1B6 policy, the model estimates a maximum capacity of around 23 printers per month. The expected 99 percentile lead time of this policy is roughly 11 work days.

The work described in this thesis covers roughly half of the project on manufacturing systems at NVBOTS. The other half is covered in Yugal Raj Jain’s thesis [1]. Jain’s thesis focuses on multiple product lines and late-stage differentiation, and analyses CONWIP and CONWIP-BtO policies. In contrast, this thesis focuses on the capacity at the current facility and the policies which NVBOTS should look to adopt in the short term. The policies that this thesis considers includes Push, BtO and SxBy policies.

Thesis Supervisor: Stanley B. Gershwin
Title: Senior Research Scientist
This page is intentionally left blank.
Acknowledgements

I would like to thank Professor Gershwin for his guidance and patience. Your confidence in our judgement helped me become a better engineer and our frequent digressions will be fondly remembered.

I would like to thank Professor Hardt for his clarity and insights. Your perspective was instrumental to help shape the project.

I would like to thank NVBOTS for sponsoring the project and its entire staff for being unequivocally approachable, friendly, passionate and knowledgeable. It was a very exciting time to be a part of NVBOTS, and I wish you good luck for the future.

At NVBOTS I would like to specifically thank Arjun Chandar for his advice and for setting clear expectations. Mateo Peña Doll for his unabating helpfulness and curiosity. AJ Perez for his enthusiasm for manufacturing and everything entrepreneurial. And Daniel Gerade for his admirable patience and willingness to teach.

I would like to thank my teammates Yugal Raj Jain and Rakshith Gokaram Narayana Murthy for being so easy to work with. You helped to make the whole experience almost too enjoyable.

I would like to thank my parents for their unconditional support.

And finally I would like to thank Abbie and her unrelenting optimism.
# Table of Contents

Acknowledgements  
---

Table of Contents  
---

List of Figures  
---

List of Tables  
---

Definitions:  
---

1. Introduction  
---

   1.1 Purpose  
   ---

   1.2 Overview of this Document  
   ---

   1.3 NVBOTS  
   ---

      1.3.1 Company Overview  
      ---

      1.3.2 The NVBOTS Facility  
      ---

      1.3.3 About NVPro and NVCloud  
      ---

      1.3.4 NVBOTS Growth Phase  
      ---

   1.4 Hardware Technology Startups  
   ---

      1.4.1 Overview of Hardware Startups  
      ---

      1.4.2 Production Ramp-up  
      ---

   1.5 Additive Manufacturing  
   ---

      1.5.1 Additive Manufacturing: Overview and Characteristics  
      ---

      1.5.2 Types of Additive Manufacturing  
      ---

      1.5.3 Applications  
      ---

      1.5.4 Market Growth  
      ---

2. Problem Formulation  
---

   2.1 Overview  
   ---

   2.2 Departments  
   ---

      2.2.1 Overview  
      ---
2.2.2 Sales and Marketing Team
2.2.3 Engineering Team
2.2.4 Manufacturing Team
2.2.5 Servicing Team
2.2.6 Software Development Team

2.3 Printer Build
   2.3.1 Purpose of Printer Build
   2.3.2 Data Collection
   2.3.3 Summary of Data and Learnings
   2.3.4 Sensibility Check of Data

2.4 Advice on Standard Operating Procedures

2.5 Quality Control Experiments

2.6 Problem Statement
   2.6.1 Problem Area
   2.6.2 Thesis Objective and Scope

3. Literature Review
   3.1 Manufacturing Systems and Scale-ups of Hardware Startups
   3.2 Push system
   3.3 Build to Order
   3.4 Batching

4. The Model
   4.1 Assumptions
      4.1.1 Operation Times
      4.1.2 Locked up Capital
      4.1.3 Inventory
      4.1.4 Workers
      4.1.5 Worker Schedule
4.1.6 Printer and Subassembly Demands

4.2 General Mechanisms

4.2.1 Overview of General Mechanisms

4.2.2 Modules

4.2.3 Dispatcher and Operators

4.2.4 Sales Modeling

4.2.5 Service Subassemblies

4.2.6 Work in Progress (WIP) Inventory

4.2.7 Released Parts Queue

4.3 Push Policy

4.4 Build to Order Policy

4.5 Simulation Run Settings

4.5.1 Warmup period

4.5.2 Run Time

4.5.3 Run Settings Summary

5. Analysis

5.1 Average Lead Time and Service Levels

5.1.1 Average Lead Time and 99th Percentile Lead Time

5.1.2 Relationship Between Average and 99th Percentile Lead Time

5.1.3 Relationship of 99% and Average Lead Time for BtO or CONWIP-BtO Systems

5.1.4 Relationship of 99% and Average Lead Time for CONWIP systems

5.1.5 Summary of the Simplification

5.2 Push Analysis

5.2.1 Overview

5.2.2 Basic Mechanics

5.2.3 Capacity versus Worker Utilization

5.3 Build to Order Analysis
List of Figures

Figure 1: NVBOTS Logo [4] ................................................................................................................. 17
Figure 2: Facility Layout ..................................................................................................................... 18
Figure 3: NVPro Printer [6] ................................................................................................................ 19
Figure 4: NVCloud Dashboard and Live Feed [7] ................................................................................ 19
Figure 5: Print Preview Feature [7] .................................................................................................. 19
Figure 6: Time-lapse (a-d) of two whistles being printed and the final part (e)................................. 24
Figure 7: Pie chart illustrating the applications of AM in industry. [18] ............................................... 26
Figure 8: BoM tree of a Rev G printer with the corresponding times. Note that this does not include the hour for picking, nor the time (~40 min) required for post processing of printed parts ......................... 34
Figure 9: A small section of the simulation showing four Sources, four Queues, four Processors and a Combiner ........................................................................................................................................ 49
Figure 10: A screenshot of the basic Sales module as used in the simulation models...................... 50
Figure 11: Example of a Released Parts Queue (RPQ) ....................................................................... 51
Figure 12: Average content of a CONWIP buffer over time for multiple replicates. ......................... 53
Figure 13: Average lead time in a CONWIP-BtO policy over time for multiple replicates. ................ 54
Figure 14: Average lead time in a BtO policy over time for multiple replicates. ............................. 54
Figure 15: Histogram of the waiting time of a Build to Order policy with a batch size of one. The dotted line represents the wait time that will result in a service level of 99%. ................................................................. 58
Figure 16: 99% lead time versus average lead time for Build to Order and CONWIP + Build to Order systems. ........................................................................................................................................ 59
Figure 17: 99% lead time versus average lead time for CONWIP systems. ...................................... 61
Figure 18: Released Parts Queue in a Push Model with a monthly demand of 15 printers ............... 63
Figure 19: Net content for a push system over time. ........................................................................... 64
Figure 20: RPQ content over time in a push system for different demand scenarios. Including scenarios where demand is less than capacity (top), demand is roughly equal to capacity (middle) and demand is greater than capacity (bottom). .............................................................................................................. 65
Figure 21: Maximum capacity versus worker utilization. ................................................................. 66
Figure 22: Net content of the finished goods buffer over time for a Build to Order policy ............... 68
Figure 23: Average lead time versus batch size for a BtO system. Demand is 15 printers per month and the worker utilization is 70%................................................................................................................. 69
Figure 24: Average lead time versus batch size in a BtO system with a monthly demand of 5, 10, 15, 20 and 25 printers. ..................................................................................................................... 69
Figure 25: Average batch fill-up time versus demand for different batch sizes ............................. 70
Figure 26: Lead time versus batch size for numerous demand scenarios. This plot is adjusted to remove the batch fill-up time ........................................................................................................................................... 71
Figure 27: Average inventory on the production line in a Build to Order system over different demand scenarios and batches. ................................. 72
Figure 28: Average WIP versus demand for a BtO system ................................................................. 73
Figure 29: Batch benefit as a function of batch size assuming different setup times. ............................. 78
Figure 30: Average lead time versus the Batch Benefit ......................................................................... 79
Figure 31: Net content in a Sell 1 Build 6 model ................................................................................. 81
Figure 32: A BtO model showing the benefits of batching. Demand is set to 15 printers per month .......... 82
Figure 33: Average lead time and average inventory against various thresholds in a Sell x Build 6 model. ........................................................................................................ 83
Figure 34: Average finished goods inventory level for different batches and demand levels ............ 85
Figure 35: Performance measures of a S1By model for different levels of y .......................................... 86
Figure 36: Capacity versus Worker utilization in a S1B6 model ......................................................... 87
Figure 37: CV_minute on the x-axis versus the corresponding output CV_month on the y-axis ............ 101
Figure 38: Normal distribution (left) and truncated normal distribution (right), with a CV of 0.1 and a mean of 384. Note that there is no discernable difference between the two distributions. ............................. 102
Figure 39: Normal distribution (left) and truncated normal distribution (right), with a CV of 0.3 and mean of 384. Any negative values are set to zero ................................................................................. 103
Figure 40: Normal distribution (left) and truncated normal distribution (right), with a CV of 1.00 and mean of 384. ........................................................................................................................................ 103
Figure 41: Input mean vs. Output mean, using the Set to Zero truncating policy. The slope of the curve is 1.26. ................................................................................................................................. 105
Figure 42: Input CV vs. resulting actual CV. The red horizontal line indicates where the actual CV should ideally be for a low variability scenario (CV_minute = 0.97), and the green vertical line shows what the CV should therefore be before it is fed into the truncating policy. Note that reaching the high demand scenario (Actual CV = 5.2) is impossible ............................................................. 106
Figure 43: Input mean vs. Output mean, using the Re-throw policy. The slope of the curve is roughly 1.08. ................................................................................................................................. 106
Figure 44: Input CV versus output CV, using the Re-throw policy. Note that a limit is reached somewhere around an effective CV of 0.8. This is insufficient to simulate a low variability scenario .......... 107
List of Tables

Table 1: AM industry revenue and growth. [18] ................................................................................................................. 26
Table 2: Output of a linear regression of 99% lead time against average lead time for Build to Order and
CONWIP + Build to Order policies. ................................................................................................................................. 60
Table 3: Output of a linear regression of 99% lead time against average lead time for CONWIP policies.
....................................................................................................................................................................................... 61
Table 4: Regression models of Figure 27. ............................................................................................................................ 72
Table 5: Percentage of total setup time that can be removed using different batch sizes.............................. 78
Table 6: Average finished goods inventory level in a SxB6 model. .................................................................................... 84
Table 7: Regression analysis of CV_{minute} vs. CV_{month} .............................................................................................. 101
Table 8: Summary table of the effects of truncation on the mean and CV of a distribution............................. 104
Table 9: Monthly demand versus the input parameter and the resulting CV ................................................................. 108
Definitions:

- **Batch fill-up time** - This is the amount of time customers spend waiting for other customers to fill up a batch of sales before the production process begins.

- **BtO – Build to Order or Build to Order.** This is a manufacturing policy where the production process only starts once a sale (or a batch of sales) is made. Parts on the production floor in a BtO system always have a buyer and therefore no finished goods inventory exists.

- **CV – Coefficient of Variation.** This is a measure of the variance in a distribution, and is defined as the standard deviation of a distribution divided by the mean of the distribution.

- **CONWIP – Constant Work in Progress.** This is a manufacturing policy where the work in progress is held constant by releasing parts into a system once a part has left. This policy is covered in Jain’s thesis [1].

- **RPQ – Released Parts Queue.** A queue used in the simulation to ensure work-orders are properly accumulated. Parts are released into the RPQ and then start the production process.

- **SxBy – Sell x Build y.** A policy where a batch (the y in SxBy) of printers is released into the system once the threshold (the x in SxBy) is made. For example, S1B6 implies that six printers are built once a single unit is sold. The remaining five printers need to be sold before another set can be released into the system.

- **WIP – Work in Progress.** This is the amount of in-process inventory.
1. Introduction

This section gives an overview of the thesis and provides context for the project.

1.1 Purpose

There are three purposes of this document. The primary purpose is to communicate the results of a simulation to NVBOTS. NVBOTS is a startup that manufactures 3D printers. The simulation is designed to test different manufacturing policies. Specific points include informing NVBOTS about the trade-offs that exists between and within different policies, and the rough lead time and inventory levels they can expect under specific settings. The secondary purpose is to detail sufficient information of the model and its workings should NVBOTS or a future MEng team decide to do further testing with this simulation. The tertiary purpose is to communicate startups, including NVBOTS, some potential problems which could prevent them from scaling up their business.

1.2 Overview of this Document

The project at NVBOTS is done in a group of three students, referred to as the MEng Team. While this thesis is distinct from the other two, in some sections the theses will be similar if not identical. All theses from the MEng Team involve the scale up of manufacturing at NVBOTS. Besides this author the group includes Rakshith Gokaram Narayana Murthy [2], whose thesis considers inventory policies at NVBOTS, and Yugal Raj Jain [1], whose thesis is on the subject of manufacturing systems. Jain’s thesis differs from this thesis as its focus is on a different set of policies and on longer term strategies. sections that are done in collaboration with Gokaram Narayana Murthy and Jain will be pointed out.

In Section 1 this thesis will first give context to NVBOTS by introducing NVBOTS as a company, reviewing typical challenges for hardware startups and discussing the field of additive manufacturing.

Section 2 then discusses the problem formulation process, and the problem statement, and gives justification for focusing on the manufacturing systems in this thesis. Note that the content of Section 1 and Section 2 are shared in all three theses, although the others may take on a different structure.

Section 3 discusses the literature on some manufacturing policies, and points out the lack of literature on manufacturing systems for startups.
Section 4 then introduces the model by going over the assumptions made and the general mechanisms that are used across the different models. The workings of the Push and the Build to Order models are also specifically addressed. Note that these are referred to as the base models, as these will be expanded on in a later section. The assumptions and general mechanics parts of this section are shared between Jain’s and this thesis.

Section 5 provides the analysis of the Push model and the Build to Order model. The capacity of NVBOTS is estimated, and it is shown that large batch sizes have negative impacts on both lead times and inventory levels. Together with the analysis of Jain’s models, some initial suggestions for NVBOTS are made.

In Section 6 the model is developed further to a Sell x Build y policy. Also, the argument is made that the per-unit processing time should fall as the batch size is increased, and this feature is also added to the model. Both the development and the analysis of these features are discussed in this section. This section ultimately fails to prove a benefit from batching on the grounds of lead time or inventory levels, but makes a case for batching based on an increase in capacity.

In Section 7 the findings of this and Jain’s thesis [1] are summarized and some suggestions are made for NVBOTS. Note that most of this section is shared with Jain’s thesis [1].

Finally, in Section 8 future work and other projects to overcome potential obstacles are detailed. This section is shared with Jain’s thesis [1].

**1.3 NVBOTS**

1.3.1 **Company Overview**

New Valence Robotics Corporation (NVBOTS) is a Boston-based manufacturing startup founded in March 2013 by four former MIT students. NVBOTS is the world’s first and only manufacturer of automated 3D printers that can function without needing human intervention. It achieves this through an automated removal system and a cloud-based software platform.

The mission of NVBOTS is to build a globally distributed network of intelligent automated additive manufacturing equipment [3]. NVBOTS’ 3D printers are connected to a cloud-based interface through which users can send print jobs that queue up and are processed in the system. Once a print job has been released and completed, the printed part is automatically removed from the print bed using a patented removal mechanism and the printer is ready for the next print job. The NVBOTS Logo is as shown below in Figure 1.
Since its inception, NVBOTS has been committed to delivering on three fronts: “education, innovation and commercialization”. Their flagship product the NVPro markets to the education sector which is currently their primary market, although NVBOTS has begun breaking into the commercial space.

NVBOTS is planning to extend its products lines. NVLABS, the R&D division at NVBOTS is developing an ultra-high speed, multi-metal 3D printing solution. This has been described as an “automated factory in a box” capable of building multiple metals in the same build [5].

NVBOTS offers its NVPro customers a seamless interfacing platform – NVCloud, through which print jobs can be monitored and approved from any device, at any time, from anywhere. Customers have full access to the NVLibrary, which is a comprehensive digital database of curricula, lesson plans and ready-to-print parts, orientated to the education sector. The extensive library and the ease of use of the machines makes NVBOTS a popular choice in this market.

1.3.2 The NVBOTS Facility

At the time of writing NVBOTS employs 18 full-time employees divided across engineering, software development, production, operations and sales. The production floor is about 1200 square feet.

The workbenches are organized for specific subassemblies and the workers move from one workbench to another to complete the assembly of the printer. All the raw-material stock keeping units (SKUs) are stored in the inventory room until a work order is issued for making new printers. After the work order is issued, the workers kit the required materials into a box before bringing them onto the production floor for assembly.

NVLABS is where the R&D activities for improving metal 3D printing technology are carried out. A schematic of the plant layout is shown in Figure 2.
1.3.3 About NVPro and NVCloud

The first NVPro 3D printer to be released into the market was the REV E (Revision E) version in 2014. With constant technological advancement, engineering development and study of other existing printers, many design improvement iterations have been made. The revisions are numbered alphabetically, and the current revision of the NVPro is REV H.

The NVPro 3D printer shown in Figure 3 is an FDM printer that has a built volume up to 570 cubic inches (9340 cubic centimeter) and has a layer resolution of 100 microns. The material extruded through the nozzle is PLA (Polylactic Acid), which is laid out layer-by-layer at a maximum print speed of 180mm/s [6]. Since the printer has the automated part removal feature and a camera that allows real-time viewing, the need for an external operator is mostly eliminated. This allows NVBOTS to stand out in the marketplace.

Users can interface with the printer using the NVCloud platform. The dashboard, as shown in Figure 4 and Figure 5, provides a live feed of the current print, indicates temperatures of the nozzle and the print bed, provides ability to queue and manage multiple prints, indicates the amount of filament left and also allows access to the NVLibrary of printable parts [7].
Figure 3: NVPro Printer [6]

Figure 4: NVCloud Dashboard and Live Feed [7]

Figure 5: Print Preview Feature [7]
This page is intentionally left blank.
1.3.4 NVBOTS Growth Phase

NVBOTS is currently experiencing high growth and is looking to ramp up production of their NVPro from 20 printers a year to about 200 a year. NVBOTS is planning to restructure their manufacturing and operations at the current facility to accommodate this transition. Within a few months, NVBOTS is planning to move to a bigger facility which will not only assemble the NVPro printers but also metal 3D printers, and a printer of flexible plastics. Both of these are currently under development.

1.4 Hardware Technology Startups

1.4.1 Overview of Hardware Startups
An important distinction needs to be made between two broad categories of startups: software and hardware. Hardware startups refer to those startups that produce physical products such as consumer goods and high-technology products. Software startups produce products that are some variations of software or services based on software. Hardware startups need to address areas of production and logistics including manufacturing, packaging, shipping, and customs [8], and therefore typically require substantially more capital compared to software startups. Compared to software development, hardware product development takes longer [9]. For instance, setting or achieving significant milestones in software development may be on the order of weeks or days. For hardware products such as consumer electronics, moving from concept to mass production may take as long as six to nine months after the associated technologies have been proven [9].

Regardless of the type, working at a startup inherently involves careful risk management [10]. These risks may take multiple forms and include financial risks, technology risks or market risks. Effort should be expanded to mitigate these risks in a cost effective manner.

A big part of day-to-day life at a startup is ‘firefighting’, which refers to tackling unanticipated problems [10]. In such cases, prioritization and time management become important drivers of success. This culture of constant firefighting is characteristic to startups.

In most startups, there is little organizational hierarchy in the company, leading to a flat organizational structure. Flat organizational structures work well for startups since they allow for fast response times [11]. The ability to pivot quickly is one of the main strengths of startups.
A flat of organization structure can allow a startup to remain nimble. However, paired with consistent firefighting an informal structure can also lead to an unstructured system. With constant pressure to solve a myriad of other problems, it becomes difficult to reflect on a particular way of doing things and determining whether that way is the most effective. As a startup enters its growth phase this lack of structure and protocol in the company can hinder further growth [12]. Some of these issues were identified at NVBOTS. For example, the lack of a clear manufacturing policy leads to an unstructured production floor which is difficult to control and scale up.

1.4.2 Production Ramp-up

Production ramp-up refers to the time period between the development cycle of a company’s product and the company’s full capacity production [13]. In other words, for a hardware startup, production ramp-up refers to the process of rapid growth of production once the startup has validated its business model and product lines for repeated revenue generation.

The phase of production ramp-up is marked by many risks and challenges, including threats from competitors, legal risks, and market risks. This is on top of internal challenges such as having inefficient policies which become too costly as the company grows.

Scale-ups require significant extra cash for additional parts, manufacturing equipment and staff. If the company does not generate a positive cash flow soon enough it may become bankrupt.

Scale-ups can only happen if the entire company scales up with it. This means that servicing and marketing will need to grow alongside production and sales.

A startup’s product might go through design and manufacturing modifications in preparation to cater to bigger markets. This would require working closely with the startup’s suppliers and vendors. Implementing these modifications also requires careful communication with the manufacturing team and the engineering team.

A hardware startup therefore has to juggle many factors when looking to scale up. Almost all facets of a startup have to be aligned. Finance, marketing, servicing, engineering, manufacturing and supply chain all need to move forward together and each of these departments need to understand that a scale up will impact areas outside their own field.
1.5 Additive Manufacturing

1.5.1 Additive Manufacturing: Overview and Characteristics

NVBOTS is in the Additive Manufacturing (AM) industry. AM has seen many startups in recent years, and it is beneficial at this point to explain what specifically is meant by AM.

AM is a collection of manufacturing processes wherein parts are built up layer by layer. AM differs from conventional manufacturing processes that typically involve material removal (such as cutting, milling, grinding), material formation (such as casting, injection molding), or deformation of an existing shape (such as bending). This distinction gives AM certain characteristics that are important to note. Some advantages of AM are listed below:

- **AM is flexible.** Many different types of parts can be made in succession with very little or no setup time between them. Little tooling and little changeover costs are required. The exception to this may be a cost associated with switching materials, which often involves something as minor as mounting a different spool onto the machine.

- **AM can make almost all geometries.** This includes geometries which are impossible to make via conventional methods, opening up new design possibilities such as making parts hollow. Figure 6 shows the printing of a whistle. Note that this would be impossible to make as a single part using conventional methods, as these methods do not allow for the ball to be manufactured within the rest of the whistle.

- **There is little material waste.** Only the material that is required is put down. The exception are the supports sometimes used to aid the printing process, however this make up a small proportion of the total material use. [14]

There are notable disadvantages of AM. Some of these disadvantages are listed below.

- **The process control is challenging.** There are many input settings that have to be fine-tuned for a part to come out according to specifications [15]. Minor deviations in ambient temperature or humidity can have major consequences in the final part quality.

- **Throughput is low.** Compared to mass production, AM is slow. For sufficiently large production quantities it may therefore be more effective to adopt processes like injection molding.

- **There are materials limitation.** Not all materials are available for AM processes. The list of useable materials is steadily growing, but this may still be a limitation for niche applications looking to use a specific material [15].
1.5.2 Types of Additive Manufacturing

There are many different types of AM, and many companies give new names to their processes even though they are arguably the same as existing ones. Many laymen believe the terms 3D printing and AM to be interchangeable, although this is not accurate. ASTM, an international standards organization, released a standard for terminology pertaining to AM. In it they define seven categories of AM [16]. As ASTM defines them, the types are:

1. **binder jetting**, *n*—an additive manufacturing process in which a liquid bonding agent is selectively deposited to join powder materials.

---

*Figure 6: Time-lapse (a-d) of two whistles being printed and the final part (e).*
2. **directed energy deposition**, *n*—an additive manufacturing process in which focused thermal energy is used to fuse materials by melting as they are being deposited.

3. **material extrusion**, *n*—an additive manufacturing process in which material is selectively dispensed through a nozzle or orifice.

4. **material jetting**, *n*—a additive manufacturing process in which droplets of build material are selectively deposited.

5. **powder bed fusion**, *n*—an additive manufacturing process in which thermal energy selectively fuses regions of a powder bed.

6. **sheet lamination**, *n*—an additive manufacturing process in which sheets of material are bonded to form an object.

7. **vat photopolymerization**, *n*—an additive manufacturing process in which liquid photopolymer in a vat is selectively cured by light-activated polymerization.

Note that at NVBOTS the process of choice is material extrusion, but that internally it is referred to as Fused Deposition Modeling (FDM). According to ASTM, 3D printing is defined as ‘the fabrication of objects through the deposition of a material using a print head, nozzle, or another printer technology’ [16]. As FDM adheres to this definition, it can be classified as 3D printing. Therefore, 3D printing, material extrusion, AM and FDM can all be used to describe the process used in the NVPro.

### 1.5.3 Applications

There are many applications of AM, ranging from purely visual purposes, such as visual aids for designers or teachers, to functional parts like production parts. Figure 7 shows the use of AM technology in industry. This is based on a survey conducted by Wohlers asking AM companies what their customers use AM technology for. Note that functional parts account for a third of all uses, and is the largest segment. This was not true even as recently as 2011. In the 2011 report by Wohlers, this section only counted for roughly 15% [17], indicating high growth for this segment.
NVBOTS has realized that there is demand for functional 3D printed parts. As materials are a major constraint in this field, there is a strong motivation to develop a metal printer. The trends therefore support NVBOTS decision of moving into this space.

A detailed analysis of these applications is far beyond the scope of this thesis. Instead, readers are redirected to other sources such as the annual Wohlers Report, or one of many textbooks available.

1.5.4 Market Growth

It is estimated that in 2015 the AM industry grew to $5.165 billion, a growth of 25.9% from the year before [18]. As Table 1 shows, a growth of 25.9% is relatively low in comparison to prior years, but is nonetheless an incredible statistic relative to many other industries.

<table>
<thead>
<tr>
<th>Year</th>
<th>Industry Revenue ($ billion)</th>
<th>% Growth</th>
</tr>
</thead>
<tbody>
<tr>
<td>2011</td>
<td>1.714</td>
<td>29.4%</td>
</tr>
<tr>
<td>2012</td>
<td>2.275</td>
<td>32.7%</td>
</tr>
<tr>
<td>2013</td>
<td>3.034</td>
<td>33.4%</td>
</tr>
<tr>
<td>2014</td>
<td>4.102</td>
<td>35.2%</td>
</tr>
<tr>
<td>2015</td>
<td>5.165</td>
<td>25.9%</td>
</tr>
</tbody>
</table>

Table 1: AM industry revenue and growth. [18]
Wohlers is expecting this growth trend will largely continue, and that by 2017 the industry will reach $8.8 billion – an average of 30% annual growth for the next two years. It also estimates that by 2021 the industry will be at roughly $26.5 billion. [18] The trends in AM technology are therefore very positive.

Using AM technology for rapid prototyping is a relatively mature application [18]. The future lies in production, with major benefits such as reduced reliance on tooling, high flexibility, and unique part geometries for better performance. However, there remain significant challenges for this application such as quality control and expanding the list of available materials.

What this means for NVBOTS is that there appears to be plenty of room to grow for the company as well as the industry as a whole.
2. Problem Formulation

This section details the team’s process of identifying obstacles that may impede a large scale-up. It then justifies the decision to focus on manufacturing systems in this thesis.

2.1 Overview

At the onset of this project there were no obvious symptoms of errors at NVBOTS. Printers were being assembled roughly within their promised time frame and were of satisfactory quality for NVBOTS’ customers. Many existing issues were about to be resolved with the release of an updated version of the NVPro – the Rev G. This model would be easier and quicker to assemble and service. However, the lack of obvious symptoms does not mean that a company is ready to ramp up its production by an order of magnitude.

The first step of the project was to identify potential obstacles that could hinder NVBOTS’ ability to scale up. Identifying these issues required a survey of the entire company. The survey was done by conducting interviews with some of the key employees at NVBOTS, going through documentation, and building a Rev G NVPro. The NVPro was built with significant help from one of NVBOTS’ engineers. During this process the importance of sales and servicing is highlighted, but it is observed that some strategies are in place to enable scaling up of these areas. There are further concerns for the production and inventory management which were not yet being sufficiently addressed.

It is recommended for companies which are looking to scale up to do a similar company-wide survey and to involve all departments. Some factors may appear trivial. For example, it may be easy to see that more parts will need to be brought and assembled during a scale up. However, there are other changes that are not obvious. Some examples are given in this section.

The section concludes with a problem statement, which also sets the scope of the thesis.

2.2 Departments

2.2.1 Overview

The first part of the project is to get familiar with the different departments in order to understand their priorities and concerns. The following sections summarize the roles, responsibilities and concerns of the departments at NVBOTS.
2.2.2 Sales and Marketing Team

The strategy of NVBOTS is to carve out a niche market position in the education sector and to develop the product before moving into the commercial market. NVBOTS’ initial target market was schools in and around Boston and New York. NVBOTS is now shifting its focus to the commercial space because it is larger and because customer acquisition will likely be less expensive.

Notable obstacles in the educational space is the low budgets of schools, and the amount of effort required for customer acquisition. Teachers often know very little about 3D printing, and have to be convinced of the virtues of it. Educating teachers and school officials about the utility of 3D printing is a large proportion of the job for the sales and marketing team. The main selling point over competitors is the ease of use. The ease of use comes from the automated part removal and the NVLibrary which included lesson plans, curricula and ready-to-print parts.

The NVPro currently does not have UL certification. UL is a type of safety certification that some companies or organizations require. In many cases the lack of this certification is not an issue, but there were some potential customers who would only consider the product if it had this safety certification. The sales team would therefore like to get the UL certification, as it would help them generate more sales. This has been put on the engineering timeline and will be addressed within the next few months.

2.2.3 Engineering Team

The engineering team works to improve the functionalities and quality of the printer. They receive feedback from the customers, production team, sales team or servicing team, and use this feedback to make improvements to the design. Since 2014 the NVPro Printer has gone through multiple design iterations to make it more robust, easier to assemble, and quicker to service.

Members of the engineering team also work along with NVLabs to develop new materials for printing. Most notably, they are working on developing the high-speed, multi-metal 3D printer. Also in development is a printer that can print flexible filament. The engineering team is making the design as modular as possible. This would allow NVBOTS to design all printer types with significant overlap in hardware which will make a late-stage differentiation strategy possible.

At the time of the interviews the engineering team was developing Standard Operating Procedures. These are used to ensure every member of the production team is assembling the printers in the same, correct way.
Establishing a set of Standard Operating Procedures is an important step, and is discussed in more detail in Section 2.3.

One serious time sink for the engineering team is filament calibration. NVBOTS receives filaments from a supplier, and then sends it to its own customers. The filament that NVBOTS receives is prone to material changes across batches, meaning that they have slightly different properties. A shipment usually comprises of one batch, and typically lasts a few weeks. However, when a new shipment is received it often means that the printer settings need to be adjusted to compensate for the material change.

The settings that have to be changed for a particular spool of filament can include printing temperature, first layer off-set, and printing speed, amongst a few others. Having the wrong settings causes the printed part to either warp or be hard to peel off the raft. This is a critical problem because a main selling point of NVBOTS is that their products are easy to use. The customer base of NVBOTS frequently does not have the technical knowledge to try and adjust the settings themselves and rely on NVBOTS to supply the correct settings.

The engineering team is running experiments to understand the important factors that affect print quality. Improvements to the calibration process are being made, but the process remains slow and can sometimes take days or weeks. This calibration issue could be an expensive problem if the amount of filament required increases substantially. Only a few people at the company have the technical knowledge to do filament calibration. However, these are the same people that NVBOTS should have working on other things like product development.

2.2.4 Manufacturing Team

The manufacturing team is responsible for the assembly of the products. The team prefers building in batches, as they feel it is easy to read a procedure once and repeat it multiple times before moving onto the next step.

There are currently three people working on the production floor. Members also have some other responsibilities like servicing or doing quality control on incoming parts.

During the interview it was found that the main problems of the production team are the limited number of tools, space, and little formal structure on the production floor. Also, the engineering team regularly introduces changes and this becomes difficult to keep track of.
2.2.5 Servicing Team

The main role of the servicing team is to ensure good customer service. Printers sometimes jam or have other issues, and this team helps to fix these issues for customers. Servicing typically includes troubleshooting some hardware or software issue or replacing a part.

Most servicing problems are usually dealt with over the phone. Sometimes it is absolutely necessary to send servicing men to customer’s location. This can be expensive considering that NVBOTS offers free servicing for three years. The team actively keeps track of servicing costs and reasons for failure. This information is passed onto the engineering team so that they can take corrective action.

The current method of doing servicing is unsustainable once NVBOTS’ customer base grows to other geographical locations. While it may be feasible for employees to go out to somewhere around Boston, going to California will be too expensive. If NVBOTS has customers all over the US, it will be imperative to have local servicing options which can be achieved through outsourcing. This is a strategy that NVBOTS is currently developing.

Failure tracking and the costs of servicing has been a topic of a previous thesis at NVBOTS [19], and readers are referred to this for more information.

2.2.6 Software Development Team

Print jobs and queues can be managed only through the NVCloud platform that interfaces with the 3D printer. An important role of the software development team is to ensure that all printers are connected to the cloud and function effectively.

The software can be used to read temperatures of the print bed and nozzle, provide live camera feed of the printer, and indicate filament levels. The software team is also optimizing the slicer software and is devising a unique coding system so that failures can automatically be tracked in the field.

As with any software there are updates to make and glitches to debug. To make this easier, NVBOTS partnered with Resin IO, a startup that provides a platform to encompass client, server and device software. This will help the software team to update the code of the customers’ printers.
2.3 Printer Build

2.3.1 Purpose of Printer Build

Doing a survey of the company makes some potential obstacles clear. However, it is suspected that there are issues on the production floor that did not come up during the initial survey.

Two printers were built. The first one was only partially built by the MEng team, as there was significant help from NVBOTS engineers. The purpose was to gain familiarity with the product. The second build was done based on written Standard Operating Procedures, and with minor help from NVBOTS employees. The purpose of the second build was to fully understand the problems on the production floor and to collect data.

There are a few questions that NVBOTS was interested in answering with the second build. What is the capacity at the current facility? How long does it take to fully assemble a printer? Is it possible to hand a set of procedures to a worker, and have them build a printer on their own or is more explanation necessary? In addition, there are some questions the MEng team are looking to answer, such as what is the bottleneck? Are the quality checks robust? Are the procedures intuitive and detailed? Which procedures are difficult to execute, and can they be improved? Are there other problems that have not been considered? The build helped to answer these kinds of questions.

2.3.2 Data Collection

The most important data to collect for this build is the times taken for each procedure. A procedure may involve something like: 'Using a 2mm hex wrench, bolt SA055 onto SA067 using 6x FR023'. (Note that SA055, SA067, and FR023 are hypothetical examples of part numbers.) There are multiple procedures for each subassembly, so the time required to build a subassembly can be found by adding the times for all the relevant procedures.

The timings are strictly only for the execution of the procedure. Before the timer starts, all the parts and necessary tools are placed on the work surface, the procedure is read in detail and sometimes clarification is sought with one of the engineers. The timer is only started once everything is in place and the required tasks are understood. The timer is stopped once the procedure is done. The process is then repeated for another procedure. It is always the same person who performs all the procedures.

The times therefore do not include time looking for tools, looking for parts or trying to understand the procedures. This implies that the times are optimistic; in reality the processes would take
longer. However, in the longer run when the workers are experienced and know the procedures, and assuming that tools are available and easily accessible, the actual operation time should tend towards our recorded time.

The procedures for this build is done by someone with little assembly experience. Someone with more experience, like a full time worker at NVBOTS, would take significantly less time. This implies that the times are conservative.

In the long run, when considering both the optimistic and conservative assumptions, it is expected that our timings will be a net conservative estimate of the actual timings. In other words, it is expected that the capacity will be underestimated. However, we expect that this underestimation is small.

During the build other information is noted down as well. This included equipment requirements, number of times Loctite is to be applied, number of times other consumables are applied, notes on the clarity of the Standard Operating Procedures, and notes on the mechanisms in a procedure and potential suggestions. This data was collected to help answer questions like how many sets of equipment are required, or how much time is spent on using Loctite.

### 2.3.3 Summary of Data and Learnings

Figure 8 below summarizes the most important timing data. Note that the figure below does not include picking nor post processing. Figure 8 also gives a crude view of the BoM tree. Eight frame subassemblies come together to form the frame. This is paired with 13 other subassemblies to make the core. After calibration the core is placed in the enclosure. This concludes the production process.

It takes roughly 10.5 man-hours to build a printer, including all the assembly steps, picking and post processing for printed parts. Note that these 10.5 hours do not include steps that do not require an operator. For example, numerous times the procedures call for a 24-hour wait period to let epoxy set. This is not included in the stated time. Also, there is a final quality control step, where the products printing for roughly two days straight. This is also not included in the time. However, the final quality control step is included in the model.
Data is collected on roughly 400 procedures, across 39 items (36 subassemblies, and 3 printed parts requiring post processing). Loctite has to be applied over 120 times per printer. The vast majority of procedures need little specialized equipment. 1.5mm, 2.0mm and 2.5mm hex wrenches, a Philips head screwdriver, an arbor press, and needle-nose pliers are amongst the most used tools. There are 231 SKUs, and 720 parts that go into a printer. 70 SKUs, or 450 parts, are dedicated to fasteners.

A significant amount of time is spent on locating tools before a procedure. For some frequently used tools, like the 2mm hex wrench, only one or two exists on the production floor. As a result, not every workbench had its own and it can be difficult to locate these tools.

Numerous times the procedures call out to use a fixture or spacer. For example, when pressing an insert into a printed part, a fixture is used to support the part so that it does not break once the press is used. Another example is the use of a spacer for when the procedure called out for a specific distance between two features. However, the location of these fixtures or spacers is not
always obvious, leading to significant searching time. The workers usually know where the tools or fixtures could be found, but this was not always true.

It will become more difficult to locate tools with more workers on the production floor. This should be considered in preparation for the scale up. Time could be saved if these items are always available. This can be achieved by having a sufficient number of tools available, and by ensuring there is a convenient place where they should return after a procedure.

A concern for production is that procedures are constantly and continuously being updated. In many cases the parts are updated alongside it. This causes confusions and disruptions.

Another issue is that inventory stock-outs occur frequently. Often multiple parts are out of stock at the same time. These stock-outs require an adjustment of the workflow, both for our build as well as for the production team’s build. The current inventory management methods therefore lead to disruption to the production floor, forcing the schedule to be dynamic and constantly readjusted.

Finally, the total workload is unbalanced. Some weeks are quiet, without anything occurring on the production floor. The following week employees could suddenly be very busy. Partly this unbalanced workload is as a result of the inventory stockouts, but it is also partly due to unexpected demand spikes and rework.

2.3.4 Sensibility Check of Data
We asked the employees at NVBOTS to roughly time themselves when they were building a few subassemblies. As the builds are done sporadically it is difficult for this data to be collected by the MEng team. It should be noted that the data does not just include the actual operation time, and is therefore different from the MEng team’s data. Rather, these timings are started when an operator starts working on a batch of subassemblies, and finishes when the batch is completed. It is therefore a more realistic estimate of production times as it also includes time spent reading procedures and looking for tools.

One employee built ten removal assemblies in roughly six hours. This translates into a per-unit time of roughly 36 min, while the MEng team estimates it as 31 minutes. At another point in time an employee built these same six removal assemblies for an average of 35min per assembly. Finally, ten printer cores were assembled in 33 hours, for around 198 minutes, while the MEng team’s data predicted it would take 189 minutes.
It was assumed in Section 2.3.2 that the data would be slightly conservative, and that the actual production rates would be slightly higher than predicted. Comparing the data with the production team’s times indicates that the data is actually optimistic. However, the prediction that the data is conservative relies on the assumption that workers do not have to consistently check the procedures and that tools are easily accessible. As the procedures are still relatively new, the timings of the production team will fall over time. In addition, it is still expected that work benches with sufficient tools would help bring the assembly time down further. It is therefore still expected that the data is a good approximation, or perhaps still slightly conservative.

### 2.4 Advice on Standard Operating Procedures

After using some early stage procedures to go through a build, some feedback is given to the engineering team. This feedback is summarized below.

Note the codes, like FM0021, refer to part numbers. AD#### refers to a grease or a glue. FR#### refers to a fastener. SA#### refers to a subassembly.

- **Remove ambiguity**
  - Be specific with dimensions. For example, instead of stating ‘approximately 10mm’, use ‘10mm ± 1mm’. Avoid relying on someone’s judgement as this will likely vary from person to person. Being 5mm off might be approximate to some, but not to others.
  - ‘Apply AD0002 to two sides of the part’ is not specific. Detail how much (thin coat, thick coat) and how (fingers, brush), and specifically where (e.g. as indicated on the diagram). For example; ‘Using a brush apply a thin coat of AD002 to the two sides of SA0020 as indicated on the diagram’

- **Use the sequence of how the operation is done to structure the sentence.**
  - A procedure like ‘Tighten FM0021 to FM0046 using FR0024 after applying AD0008’ is confusing. In this procedure it is possible to start following the steps before reading the entire sentence. AD0008 has to be applied before the parts are tightened together, so put this at the beginning of the procedure. For example: ‘Dip FR0024 into AD0008 and use the 1.5mm Hex Wrench to tighten FM00021 to FM0046’
- **Mention the tools in the procedure**
  o Instead of saying ‘Tighten FR0022’ state what tool is required. This stops the operator from trying different tools. For example, ‘Using a 2.5mm Hex Wrench, tighten FR0022’

- **Standardize the layout and format**
  o Design procedures to have the instructions, tools and parts list at the same location on every page.
  o Name things the same way. For example, do not switch between calling something an Allen key or a hex key. Decide on one.
  o Use the same phrasing everywhere. For example, if it is decided to use ‘dip FR0021 into AD0032, and insert it into SA0029’ then don’t use different phrasing somewhere else to describe a similar procedure.

- **Standardize procedures that are frequently repeated in a separate document**
  o For example, on a separate document it should be stated how to apply Loctite or how to operate the arbor press. This is to ensure that operators are doing these procedures in the same way.

### 2.5 Quality Control Experiments

A lack of a definition of ‘goodness’ is an issue for NVBOTS. As part of a separate project, a statistical analysis experiment was conducted on three NVBOTS printers in an effort to provide a starting point for checking the variation in performance of the printers.

On an analytical level, the objective of the statistical analysis experiment was to study the variation in performance among different printers, at different locations within a printer, and within the inherent printing process. An analysis of variance and a nested variance analysis was performed for different samples printed on the three printers.

The results indicate that there is minimal variation among printers. This suggests that the newly written Standard Operating Procedures are effective. Whether the performance of the printers is satisfactory would need to be determined by NVBOTS. NVBOTS implemented a more rigorous quality control test for printers following this experiment. It was therefore decided by the MEng team to shift focus to other problem areas.
2.6 Problem Statement

2.6.1 Problem Area
Numerous obstacles that could impede NVBOTS’ scale-up are identified during the survey of the company and the build of the two printers. These obstacles are detailed in Section 8, to serve as reminders for NVBOTS, and to help other hardware startups to identify some issues that exist in their own companies.

Ultimately a different project was chosen as the main focus of this thesis and also for Jain’s [1] and Gokaram Narayana Murthy’s [2] thesis. Other projects were found to be of relatively low priority or were already being worked on by others who may be in a much better position to handle these obstacles. The exceptions to this were the manufacturing system and the inventory management. Employees at NVBOTS either did not have the time or the background to redesign a manufacturing system or an inventory management system. The lack of a well-designed manufacturing or inventory management system was found to be a critical obstacle to scale-up, and it was decided to make these the topics of the theses.

Section 2.6.2 below details the specifics of the problem, and sets the scope of this thesis.

2.6.2 Thesis Objective and Scope
Some issues exist on the production floor. First, there is little structure as to when printers should be made and how many should be made. Decisions on new builds are made ad hoc. Batch sizes of 3, 6, 10 or 12 printers are made sometimes as a result of, or sometimes in reaction to demand. An explicit policy should be developed in order to create more structure on the production floor. The policy should be sensible and clear.

Failing to have a well-defined policy will lead to confusion. Confusion may lead to some workers spending time on work that has a low priority, or to do work that does not have to be done at all. It needs to be made abundantly clear what has to be done, and when it has to be done by. The sales team will be making promises on delivery times, and if those promises are not kept it will tarnish the brand value of NVBOTS. NVBOTS currently promises a lead time of four to six weeks. However, six weeks is a long time, and with competition growing fiercer the lead time may become an important competitive advantage. The lead time should therefore ideally come down to less than four weeks. The first objective is therefore to design a clear policy that will allow NVBOTS to add clarity to the production floor, and reduce the lead time of the NVPro.
Secondly, the capacity is unknown. This is problematic as it makes it difficult to plan for expansions, to project the feasibility of sales volumes, or to develop a recruiting plan for hiring new members of the manufacturing team. The second objective is therefore to estimate the capacity of the production floor.

In order to find solutions to these concerns this thesis and Jain's thesis [1] will develop a set of simulations which will model the effects of different manufacturing policies. This thesis will consider a Push policy, a Build to Order policy, and a variation on the Build to Order policy (named Sell x Build y), and give estimations for the expected lead time and inventory levels. It will also give suggestions of what batch size to use. Additionally, this thesis will estimate the capacity under different conditions.

Jain's thesis [1] will focus on longer term strategies, including a CONWIP policy and a late stage differentiation policy.
3. Literature Review

This section will give a brief overview of some literature involving manufacturing systems, including a Push system, a Build to Order system, and batching.

3.1 Manufacturing Systems and Scale-ups of Hardware Startups

Manufacturing systems are defined as ‘a set of machines, transportation elements, computers, storage buffers and other items that are used together for manufacturing’ [20]. A manufacturing system has a set of performance measures, like the average Work in Progress (WIP), lead times, flexibility or capacity [20]. The objective when designing a manufacturing system is to optimize some desired combination of performance measures. For example, the lead time will be set at some target and the system is designed to minimize inventory levels while still adhering to this lead time. Numerous papers and textbooks have been written on the topic of manufacturing systems [20]–[25], however little literature appears to be available specifically for startups looking to scale up production.

Three pieces of work have been found on the topic of scaling up production for startups. These are the three theses written at NVBOTS in 2015 [19], [26], [27]. However, while these theses do make some mention of the need for a structured manufacturing system, they do not specifically address the problem of designing manufacturing systems for startups.

This thesis will borrow from the extensive literature on manufacturing systems, and attempt to apply their learnings to a startup environment.

3.2 Push system

There is some contention as to what distinguishes a Push system from a Pull system. Some authors [28] base their definitions on the direction of material flow – information flowing with production is Push, information flowing against it is Pull. Others consider Push to be a system where machines produce parts without regard to the downstream machines [29]. In contrast, in a Pull system a machine only produces once it has been tasked to do so by a succeeding machine. Non-academic sources can go by other definitions, like by defining Push as a system where WIP is not controlled via policy while for Pull the WIP is actively managed [30].
The reality is that for the purposes of this thesis it matters little what definition is used as long as the meaning is made clear. This thesis will define Push as a manufacturing strategy where parts are produced purely in anticipation of demand, and not in response to demand.

A benefit of a Push system is that production can begin in anticipation of demand, rather than reacting to it. This allows companies to plan in when they will build orders, possibly giving them more flexibility for machine or worker scheduling.

Some products take a lot of time to build, but the delivery of them may be time-sensitive. In this scenario a Push model can be a good choice. By the definition used, practically all existing supply chains include some push section. For example, mining companies do not start taking minerals out of the ground only once Boeing receives a new order from an airline. Instead, the supply chain will store many parts in anticipation for demand.

The main disadvantage of Push systems is that they heavily rely on accurate forecasts, but there are limits to forecasting methods [31]. The likelihood and cost of excess inventory at the end of a selling season have to be weighed against the likelihood and cost of missed sales. For some products it is relatively easy to get useful information as there is sufficient historical data. But for other products there may not be any historical data available to aid in predictions. A startup typically falls in the latter category, and will therefore have difficulties predicting the sales of their products.

### 3.3 Build to Order

A Build to Order (BtO) policy differs from a push system because it does not rely on forecasting at all. The policy is to only schedule and build a product after an order has been confirmed [32]. Therefore, a BtO strategy involves responding to demand, rather than anticipating it.

The main benefit of a BtO system is that it can reduce inventory and therefore reduce the amount of capital tied up in inventory [32]. In a BtO policy there should not be a finished goods inventory as all finished products are already sold before the production process begins. This has the potential to free up a substantial amount of capital which can provide a return if alternatively invested.

A BtO model is especially attractive if customers are likely to want a customized product. For example, a car can be sold with different kinds of optional features, with some examples including a sunroof, an in-built navigation system, luxury seating, or a rear-view camera. On top of that
cars can be sold in a large range of colors. Holding stock for every permutation of color and features will quickly become very expensive and tie up a lot of capital. Additionally, not all permutations may be as desirable as expected. This could lead to a high scrap rate at the end of a selling season. At the same time, some permutations may be more desirable than expected, leading to a loss in potential revenue. It may be more effective to let a customer place an order, and to build a car with the specified features and color.

A drawback of a BtO system is that it increases the lead time. Customers will have to wait for the entire production process before they can have access to their product.

If there are multiple variations of a product being made, then one strategy to reduce the lead time is to adopt a late stage differentiation policy. In this system, part of the production process is done in anticipation of demand and stored. Once a sale is made these parts are then combined to give the desired permutation. This type of system often requires a modular design to work effectively [33].

NVBOTS is planning to adopt a modular design for their future products. A late stage differentiation policy could therefore be very reasonable. For further information about late stage differentiation at NVBOTS, readers are referred to Jain’s thesis [1] where the feasibility is developed in more depth.

3.4 Batching

Batching is used to describe the practice of processing multiple units of something before the group, or ‘batch’ moves on to the next stage. A batch size is the number of items that constitute a batch [34].

There exists a large amount of literature on the topic of optimizing a batch size when considering the setup costs. Large batch sizes will take longer and lead to higher Work in Process (WIP), but will also reduce the total amount of setup time [35]. Having a batch size of 20 versus a batch size of 2000 means that the setup time has to be incurred 100 times as often. Batching can therefore lead to increases in throughput and the maximum capacity, but it comes at the cost of higher WIP.

NVBOTS prefers to work in batches when assembling their printers. No literature has been found on the topic of batch sizes when it comes to a manual assembly process, and therefore there is little prior work that can be considered when building a model for batching. However, it will be
argued in Section 6.1.1 that a simple model based on a setup time and a processing time will be reasonable for the purpose of this simulation.
4. The Model

This section gives an introduction to the model, including its assumptions and general mechanisms. It also details the mechanisms behind the Push policy and the BtO policy. Finally, it addresses some decisions made on simulation run time, and the process for simulating demand.

Note that Section 4.1 and 4.2 are shared between this and Jain’s thesis [1].

4.1 Assumptions

4.1.1 Operation Times

The operation times for almost all subassemblies and processes are assumed to be normally distributed. The mean values of all operation times are taken as the operation times recorded during the initial printer build. Section 2.3.2 discusses the methods used to collect this data.

Since these operation times are based on a single data point (i.e. a single printer build), it is reasoned that the recorded operation times may not represent true operation times. To account for deviations and variability in operation times a coefficient of variation of 0.3 is assumed for all procedures requiring a worker. The MEng team deemed this to be a reasonable representation of the variability in an assembly process.

The only exception to the normal distribution is the printing time during the final quality control. This procedure requires no operators, and this operation is set to be deterministic.

As discussed in Section 2.3.2, the recorded operation times only include the times spent performing the actual operations. Time spent looking for tools, or assimilating and reading the procedure steps, are not included. This implies that the times used are optimistic.

As discussed in Section 2.3.3 and Section 2.3.4, it is expected that full-time staff at NVBOTS is more skilled at assembling tasks. This implies that the times used in the model are conservative.

We assume that these optimistic and conservative assumptions roughly balance, or perhaps lead to slightly conservative estimations.

Lastly, for simplicity of the models, it was assumed that no rework is required as little rework was observed.
4.1.2 Locked up Capital

Inventory is expressed as the average number of printers that exist on the production floor, which includes finished goods inventory. This is done to avoid giving away NVBOTS' costs.

For the Build to Order policy, the locked up capital is estimated using the WIP FlexSim module as explained in Section 4.2.6.

4.1.3 Inventory

For the purpose of the model it is assumed that raw material inventory is always available. The scope of the simulation models is limited only to the production floor, and does not account for the absence of raw material inventory. For detail on inventory policies at NVBOTS, readers are referred to Gokaram Narayana’s thesis [2].

Time spent picking parts from inventory is assumed to be independent of the batch size. This implies that picking for a batch size of two would be the same as the time spent for a batch size of six. It is found that the majority of time spent on inventory picking comes from operator movement when moving to different locations as opposed to picking multiple parts from the same location. Consequently, different batch sizes did not significantly affect inventory picking times.

4.1.4 Workers

For simplicity, it is assumed that all workers work full time and are completely interchangeable in their roles. This means that any worker is capable of producing any subassembly on the production floor, and that there are no specialized skillsets associated with any worker.

4.1.5 Worker Schedule

It is assumed that there are 20 workdays in a month, and that a workday constitutes of 480 work-minutes (8 hours). Worker utilization, which is the fraction of time workers spend on production, is assumed to be 70%. Worker utilization is covered in more detail in Section 5.2.3.

4.1.6 Printer and Subassembly Demands

Initially we tried to use a distribution of inter-arrival time which would allow the adjustment of the variability as well as the mean of demand. However, no adequate solution was found. The investigation into alternative distributions is detailed in Appendix A. Our process is detailed there to prevent others having to repeat the same work.

In all simulations the inter-arrival time between orders for a printer is assumed to be exponentially distributed. For example, an inter-arrival time of 480 minutes simulates a demand of 20 printers per month.
There are some subassemblies that are needed for servicing. For example, the Hot End, which is essentially the orifice that extrudes the material, sometimes needs to be replaced in the field. The other service item that is included in the model is the Encoder, which registers whether the filament is moving through the orifice. For these subassemblies the demand is assumed to be twice that of the printer demand. This means that the inter-arrival time of service items is half that of the inter-arrival time for printers. Note that the inter-arrival time for service items is also assumed to be exponentially distributed.

4.2 General Mechanisms

4.2.1 Overview of General Mechanisms

The simulation is run using an academic version of FlexSim, which is a discrete event simulator designed to be used for manufacturing and material handling [36].

The models are mostly comprised of modules which in aggregate represent NVBOTS' production process. The main modules used in the simulations include the Source, Sink, Processor, Combiner, Separator and Queue. Every model also has a single Dispatcher and three Operators unless stated otherwise. By combining these modules, and by using triggers if certain conditions are met, complicated policies can be simulated.

This section will first introduce the five main modules, as well as the Dispatcher and Operator, and will then show how these are combined to simulate different parts of the model.

4.2.2 Modules

Note that material flows from left to right in all figures in this section.

Source

The Source is a module that generates material, and is the only way parts can enter the system. For example, parts coming out of inventory come out of a Source. In addition, demand is simulated using a Source by having it create a work order.

Sink

The Sink is the module that allows material to leave the system.
Processor

Every subassembly has its own processing station. If a subassembly required parts from just a single upstream module, then a Processor is used. The Processor takes one input and provides one output. The model operates in batches, meaning that an entire batch is taken as the input, and a batch is then also the output.

Most Processors require a Worker to be present. The processing time can start once the required part and the Worker are available, and the processor is no long occupied. For some Processors, like the Final Printing Processor, no Worker is required, and the processing time will start without a Worker. In reality a Worker must initiate the process, but this takes up a small amount of time and is ignored for simplicity. Once the processing time has elapsed the output batch can go on to the next module, assuming it is not blocked. For Processors that do require a Worker, the Worker is required to be at the module for the entirety of the processing time.

Note that Operators are used synonymously with Workers in this thesis.

Processing times are modeled as a normal distribution. The mean is a function of batch size, procedure time and picking time. Procedure time is equal to the sum of all procedure times that are required to build that subassembly. In the model it is the processing times that are used as inputs to the modules. The processing time includes both procedure time and picking time.

To simulate batch production, the procedure time is multiplied by the batch size to give the total processing time for a batch of subassemblies. Note that as this is a direct multiplication it means that no improvement is made on the per-unit processing time. There is only a slight decrease in the per-unit processing time when batch size increases as the inventory picking time is only counted once, regardless of batch size. However, this effect is relatively minor.

If the mean processing time is $\mu_{\text{processing}}$, then it is calculated as shown in Equation 1 below.

$$\mu_{\text{processing}} = \text{Batch Size} \times \text{Procedure Time} + \text{Picking Time}$$  \hspace{1cm} \text{Equation 1}

The standard deviation is calculated using the Coefficient of Variation (CV) which is defined in Equation 2 below.

$$CV = \frac{\sigma}{\mu}$$  \hspace{1cm} \text{Equation 2}
We assume that CV is set to 0.3 in all random models. Note that it is assumed that picking is deterministic and is therefore not included in the calculation of standard deviation. It is assumed that the different procedures in a batch size have independently distributed processing times. Therefore, the variance of the distributions should be multiplied by the batch size. Or in other words, the standard deviation should be multiplied by the square root of the batch size. If $\sigma_{\text{processing}}$ is the standard deviation of the processing time, then it is expressed as shown in Equation 3.

$$
\sigma_{\text{processing}} = 0.3 \times \text{Procedure Time} \times \sqrt{\text{Batch Size}} \tag{Equation 3}
$$

The processing time is then a normally distributed random number, with a mean of $\mu_{\text{processing}}$ and a standard deviation of $\sigma_{\text{processing}}$. Or as shown in Equation 4:

$$
\text{Processing Time} \sim N(\mu_{\text{processing}}, \sigma_{\text{processing}}) \tag{Equation 4}
$$

Note that the picking time factors into the mean processing time, but not its standard deviation.

**Queue**

A queue is a temporary holding place for parts, and is sometimes referred to as a buffer. In the figure it is represented by a grey rectangle.

**Combiner**

If a subassembly has multiple subassemblies that precede it, then the subassembly will be modeled as a Combiner. An example is shown in Figure 9. Subassembly SA0100, shown on the right in yellow, requires an input from each of the upstream Processors connecting to it. These are the SA0119, SA0120, SA0121 and SA0124 Processors shown in green.

Figure 9 also shows the sources on the left hand side, represented by a blue square with a green arrow. The Queues are represented by a grey rectangle.

Note that the Combiner uses the same approach to processing times as the Processor.
Figure 9: A small section of the simulation showing four Sources, four Queues, four Processors and a Combiner.

Separator

A Separator is essentially the opposite of a Combiner. It takes a single input, but generates multiple outputs connecting to multiple downstream modules. The Separator can also be used to deliver multiple outputs to the same downstream module. The main use of Separators in the model is as a ‘de-batcher’. This means that when a batch of six enters the Separator, six individual units will go to a single downstream module.

4.2.3 Dispatcher and Operators

Most Processors and Combiners require an Operator before the procedures can be started. The exception of this is the final printing stage, which does not need an Operator. Operators are directed to a workstation using a Dispatcher.

The Dispatcher delegates task sequences to all Operators. Decisions are based on a priority system. Once an Operator finishes a task, the Dispatcher will direct it to another workstation depending on what has the highest priority at that time. Note that it is only after an Operator completes a task that he can move on to another task, even if the other task has higher priority.
Workstations relating to service items are given highest priority. Besides this the further downstream workstation has the highest priority. For example, doing the final set of procedures in a printer’s assembly is given a higher priority than an early subassembly. However, making a Hot End for a service job is given higher priority still.

4.2.4 Sales Modeling

Sales are generated according to an exponential distribution. A sale is represented by a work order, which is generated in a Sales Source, and waits in a Waiting Queue. The work order has to be matched with a finished printer before either leaves the system. The modules representing this sales process are shown in Figure 10. The To Sell Queue (holding finished printers) and the Waiting Queue (holding work orders), are paired in the Sales_Fulfilling Combiner. Note that immediately afterwards the Timer_Separator Separator splits the material flow into two Sinks. This separation is needed to collect data on the individual waiting times for each of the work orders. The Sales_Sink Sink takes the difference in time between the work order creation, and the time the work order enters the Sink. (i.e. the production lead time) This is then recorded in a table.

Figure 10: A screenshot of the basic Sales module as used in the simulation models.

4.2.5 Service Subassemblies

In addition to demand for complete printers, NVBOTS also has field servicing requirements for certain subassemblies such as the Encoder (SA0099) and the Hot End (SA0103), which frequently requires replacement. As mentioned these subassemblies have a higher priority in order to ensure high customer satisfaction.
The demand for service items is modeled in much the same way as is done for the printer sales. There is a work order that has to be combined with a Hot End or an Encoder.

4.2.6 Work in Progress (WIP) Inventory

The WIP is calculated by taking the difference of the total number of printers that have gone into the system and the total number that have left the system. The net WIP is averaged over the entire run time to provide the average WIP content.

4.2.7 Released Parts Queue

The Released Parts Queue (RPQ) is used to model the beginning of production when a batch of demands arrives in the system. It accumulates parts that have been released from inventory, and are ready to begin assembly.

The main purpose of the RPQ is to ensure that work orders are not missed. The alternative is to have the inventory (represented as a source) go straight into the first Processor. However, without a RPQ sometimes quick consecutive sales can be made. The first batch is then released onto the system, followed closely by the second batch. However, if there is no RPQ, and if the first batch has not yet passed the first workstation, a blockage will arise. This can cause the second batch to not enter the system, meaning that it is never matched with a printer. The RPQ resolves this by accumulating released parts such that every sale is matched with a printer.

An example of an RPQ is shown in Figure 11. Source2 will receive a message (the specifics of which depend on the policy), and will then release a batch of parts into RPQ_SA0110. Once a Worker is available and the SA0110 Processor is free, the part will be taken out of RPQ_SA0110 and start the production process.

![Figure 11: Example of a Released Parts Queue (RPQ)](image-url)
4.3 Push Policy

The Push policy replicates a build-to-forecast system. This model serves a useful purpose because it is simple and allows for an intuitive interpretation of the capacity at NVBOTS.

The policy is that every month (i.e. every 20 workdays or 9600 simulated minutes) a set of parts are released onto the production floor. This set is equal to the expected monthly sales. The inter-arrival time of the sales are exponentially distributed, with a mean such that on average a predetermined number of sales are made each month. Assemblers work until all printers are finished, and if there is nothing left to do they remain idle.

4.4 Build to Order Policy

The Build to Order (BtO) system works by only releasing parts onto the production floor once a sale is made, or when a batch of sales is made. For example, if the batch size is set to three, then once three sales are made, one batch is released into the RPQ. The entire batch is then taken through the system. Because of batching a more accurate term for this policy would be Build to Order with Fixed Batch Size. However, it will be referred to as Build to Order or BtO.

The main lever for this policy is the batch size. The production team prefers to work in batches, but as the analysis section will illustrate, this increases both the WIP and the lead time.

4.5 Simulation Run Settings

4.5.1 Warmup period

The model will have a starting bias that should be ignored in the analysis. For example, in the CONWIP model used in Jain’s work the buffers are injected with finished printers at the very beginning of the simulation. The injection provides a starting point that is necessary for the simulation to run. However, this should not be considered in the simulation as it may skew the results. In addition, some performance variables are logged in such a way that the values are initially nonsensical. These should be ignored. For example, the Average Lead Time label in all models is set to zero every time the simulation resets. Until the first printer is paired with its customer the model therefore shows a value of a zero average lead time, which is not realistic for Build to Order models where there by definition will be some waiting time. However, the value of this label is nonetheless recorded. The first parts of the recording should therefore be discarded.
These initial biases should be removed from the analysis, which is typically done by setting up a warmup period. The warmup period allows the model to settle before any data is taken and anything that happens during the warmup period is ignored in any analysis. The next step is to determine how long this warmup period should be.

The main performance measures of the models are the average lead time and the average content in buffers. The average lead time and average content should therefore be the main determination of the warmup period. Figure 12 below shows the average content of the core subassemblies inventory in a CONWIP-BtO system for four replicates. The CONWIP-BtO model is used in Jain’s thesis [1]. At the very beginning of the plot the values go from zero (at time zero) to roughly ten. This is the injection mentioned earlier. Over time this value then quickly drops. Somewhere before 100,000 simulated minutes the values appear to stabilize.

Figure 13 below shows a similar plot, but now shows the average lead time in a CONWIP-BtO system. In the first few thousand simulated minutes the average lead time quickly rises from zero to roughly 1600 minutes. This is due to the data logging in the simulation; if no sales have been made yet, the data logs a lead time of zero. Within the first ~20,000 minutes the model appears to begin logging sensible data.
The other policies behave in much the same way; there is an initial adjustment period but it is relatively short. For example, in Figure 14 below the values appear to settle around the 100,000-minute mark.

Figure 13: Average lead time in a CONWIP-BtO policy over time for multiple replicates.

Figure 14: Average lead time in a BtO policy over time for multiple replicates.
Following this analysis, it appears as though the initial adjustment period is typically on the order of $10^4$-$10^5$ minutes. In order to be safe, and because the warmup period is computationally cheap in the software (as data is not logged), all data runs will be done with a warmup period of $10^6$ minutes.

4.5.2 Run Time

Once the warmup period is finished, it is assumed that the model will start collecting usable data. The run time should be long enough to collect sufficient meaningful data during this state.

A BtO model is run under a particular set of settings for 60 million simulated minutes (approximately 120,000 printers) and is repeated for a total of five replicates. The average is found to be 2247 min, with a standard deviation of 8.9 min. It is assumed that an average lead time of 2247 min is therefore the long term average, and is the ‘true’ output of this model under these settings. The same model is then run numerous times for three million simulated minutes. Three replicates are averaged together from these runs to give an estimate of the performance measure. This is repeated until there are 43 estimates based on 129 runs of three million simulated minutes each. Out of the 43 estimates of the performance measure, 42 (or 98%) lie within the band of 2247 ±5%. This will be deemed sufficient for the model’s purpose.

A similar approach is done on a CONWIP-BtO model to confirm whether three replicates of three million simulated minutes each would result in an estimate that lies within 5% of the ‘true’ estimate. The model is left to run for five replicates of 60 million simulated minutes. Across these five replicates, the average of the average lead times is found to be 1634 minutes with a standard deviation of 3.9. Once again this is assumed to be the ‘true’ value. As above, 43 estimates of the average lead time are derived from 129 runs of three million simulated minutes each (three replicates are averaged to give an estimate). 43 out of the 43 estimates (100%) lie within the band of 1634 ±5%.

Finally, the average content in the frame subassemblies buffer in a CONWIP-BtO policy is recorded. The long term average content is 6.31 with a standard deviation of 0.016. Out of the 43 estimates, 43 (100%) lie within the band of 6.31 ±5%

Running the simulation for three million minutes, and averaging across three replicates would therefore be justified. However, in order to be conservative, a run time of five million minutes per replicate will be used instead.
4.5.3 Run Settings Summary

Based on conservative calculations the recommended warmup period is one million minutes, and the recommended run length is five million minutes. Three replicates should be averaged to give the estimate of the performance variable.

Depending on the specifics settings it takes between one and four minutes for a typical run of five million simulated minutes. Simulations were on a laptop with an Intel Core i7-4600 CP 2.70 GHz processor, and 8.00 GB of RAM.
5. Analysis

This section first shows a relationship between the average lead time and the 99% percentile lead time, which is useful for analysis. It then analyses the Push and BtO models. It includes a brief summary of Jain’s work on CONWIP and a CONWIP +BtO model [1], and makes some suggestions based on the results. It concludes with some feedback from NVBOTS.

Unless stated otherwise, all simulations assume three workers, with a worker utilization of 70%, a demand of 15 printers/month and a batch size of one.

All figures are based on data from the simulations, unless stated otherwise.

5.1 Average Lead Time and Service Levels

5.1.1 Average Lead Time and 99th Percentile Lead Time

A distinction should be made between the average lead time, and the 99th percentile lead time. The danger with using the average lead time when giving quotes to customers is that roughly half of the customers will not receive their orders in time (depending on the distribution on the waiting times).

The 99th percentile lead time (99% lead time for short), is the lead time in which 99% of customers get their printer produced. Setting a 99% lead time as the lead time quoted to customers implies that only 1% of customers will get their orders late. This method is frequently used in industry, where the 99% value is referred to as the service level.

In the simulation the average lead time can easily be determined as it is a standard output of the model. The 99% lead time is more difficult and has to be derived from the data collected on each customer in the simulation. The waiting time for each customer is recorded, and this data is then fed into a MATLAB script. The MATLAB script is added to Appendix C.

The script works by ordering all the waiting times from shortest to longest, and then taking the 99th percentile value. An example is given below in Figure 15 for a Build to Order system. Note that there are no occurrences of lead times less than 1500min. This is because it takes at least this long to build a printer with three workers. However, sometimes customers have to wait longer because the workers are already working on an order. In Figure 15 the 99th percentile is represented by the dotted line. In this scenario the average waiting time is roughly 1600 min (or 3.3 workdays), but the wait time NVBOTS should quote is 2400 minutes (or 5 workdays).
Figure 15: Histogram of the waiting time of a Build to Order policy with a batch size of one. The dotted line represents the wait time that will result in a service level of 99%.

5.1.2 Relationship Between Average and 99th Percentile Lead Time

As shown in Section 5.1.1 there is an important and significant difference between the average and 99% lead time. This is undesirable from the perspective of running numerous replicates, scenarios and policies. Collecting the 99% lead time to do an analysis requires the MATLAB script to be run manually each time. This requires considerable additional time. If instead the average is used, then data collection will be much quicker.

It is possible to use the average in lieu of the 99% lead time if there is a reasonably strong relationship between the two. For example, assume that the 99% lead time is always double the average lead time. If it is required to find settings such that the 99% lead time is less than 20 workdays, then instead the settings for average lead times less than 10 workdays can be looked for. This greatly simplifies data collection.

The simplification of using the average lead time as a proxy for the 99% lead time requires a strong correlation to exist between the variables. Applying a regression model on all the available data initially resulted in an unusable relationship. However, it was then discovered that the initial result is due to there being two categories of policies. The first group includes the BtO policies. The second include the pure CONWIP policies. Once the data is separated into these two groups the relationship between average lead time and 99% lead time is strong enough to use the average lead time as a proxy.
Section 5.1.3 will analyze the regression of a BtO for this thesis and a CONWIP-BtO system as can be found in Jain’s thesis [1]. Section 5.1.4 will analyze the regression for a pure CONWIP system as can be found in Jain’s thesis [1].

5.1.3 Relationship of 99% and Average Lead Time for BtO or CONWIP-BtO Systems

Figure 16 below shows the relationship between the 99% lead time and the average lead time for the BtO or CONWIP-BtO system. The data used is from a range of different scenarios and replicates. There appear to be two outliers, shown as red triangles, which correspond to scenarios where the demand is close to capacity. After further investigation the max capacity for these scenarios is found to be around 26 or 27 printers per month, and the demand is set to 25 printers per month. This simplification should therefore be used carefully in scenarios where capacity is approximately equal to demand.

![Figure 16: 99% lead time versus average lead time for Build to Order and CONWIP + Build to Order systems.](image)

Other than these two points there appears to be a strong correlation between the average lead time and the corresponding 99% lead time. This is confirmed when a linear regression model is applied to the data, the results of which are shown in Table 2 below. The adjusted R-Squared value is 0.98.
<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>1431.2</td>
<td>145.38</td>
<td>9.8442</td>
</tr>
<tr>
<td>x1</td>
<td>1.3911</td>
<td>0.0318</td>
<td>43.747</td>
</tr>
</tbody>
</table>

Table 2: Output of a linear regression of 99% lead time against average lead time for Build to Order and CONWIP + Build to Order policies.

The results of this analysis suggests that for BtO or CONWIP-BtO systems the 99% lead time can be reasonably well approximated from the average lead time. The formula is as shown in Equation 5 below. It is recommended that the capacity should be above (by at least 10%) the demand when this approximation is used. Note that the units are in work-minutes. There are 480 work-minutes in a workday.

\[
99\% \text{ Lead Time} = 1.4 \times \text{Average Lead Time} + 1400
\]

*Equation 5*

Note that this relationship only holds for the current facility and the current procedures. Once changes are made to the production line or the product, this equation will no longer apply.

### 5.1.4 Relationship of 99% and Average Lead Time for CONWIP systems

The analysis of 99% lead time against the average lead time is now done for the CONWIP systems. The plot of this data is shown below in Figure 17.

Note that the data in Figure 17 deviates from a linear regression when the average lead time is smaller than roughly 100 minutes (or about a fifth of a workday). This is because in these scenarios most customers do not spend any time in the system as there is typically sufficient stock in the finished goods inventory. At the same time, it is not possible to have negative sale times. If negative sale times were possible, then the average lead time would likely be far lower than it currently is. The average lead time is therefore skewed to be higher than it normally would be, but the 99% lead time, representing the extreme cases, is not. The closer the average sales time is to zero, the more skewed the results will be. This would explain why the slope of the curve is significantly higher at low average lead times, and why it drops down at longer lead times.

Even at longer lead times the slope is higher than in the BtO system— i.e. the difference between the average and 99% lead time is higher in the CONWIP as compared to the BtO system. In the CONWIP system the average lead time will be lower, as production does not need to wait for a
sale to be made. However, there is still a chance of rare events like surges in demand, which will dictate the value of the 99% lead time. If there is a rapid succession of sales, then the customers will be forced to wait for a significantly longer duration than normal. This difference between typical and unexpectedly high demand is simply more extreme than it is in the BiO systems. This causes the slope to be steeper.

An argument can be made about doing a more sophisticated regression to adjust for the region where average lead time is low, or to alternatively do two kinds of linear regressions; one when average lead time is < 100min, and one when average lead time is >100 min. However, in reality NVBOTS would not quote a lead time of 100 minutes regardless of the manufacturing policy. As a result, using a single linear model will be a reasonable approximation.

![Graph showing 99% Lead Time versus Average Lead Time for CONWIP systems.](image)

*Figure 17: 99% lead time versus average lead time for CONWIP systems.*

A linear regression done on the data results in an adjusted R-Squared value of 0.94. The result of the regression is as shown in Table 3. The resulting equation is as shown in Equation 6. Note that the units are in work-minutes. There are 480 work-minutes in a workday.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>1407.2</td>
<td>227.29</td>
<td>6.1911</td>
<td>3.13E-06</td>
</tr>
<tr>
<td>x1</td>
<td>5.4703</td>
<td>0.29069</td>
<td>18.818</td>
<td>4.74E-15</td>
</tr>
</tbody>
</table>

*Table 3: Output of a linear regression of 99% lead time against average lead time for CONWIP policies.*
5.1.5 Summary of the Simplification

It has been shown that the 99% lead time can be approximated reasonably well using a linear model and the average lead time. If a BtO or CONWIP-BtO model is used, the model as expressed in Equation 5 should be applied. The exception to this is if the monthly demand is within ~10% of the monthly capacity, as it appears as though this causes the linear model to fail. If a CONWIP system is used, the model should be as expressed in Equation 6, however the linear approximation degrades when the average lead time is very small (i.e. less than 100 minutes).

What this means in the analysis of the simulation is that when trade-offs are investigated, the average lead time can be used as a proxy for the 99% lead time. For example, instead of determining whether the batch size has a significant effect on the 99% lead time, the relationship between the batch size and average lead time will be investigated. This means that the average lead time will be the standard performance measure, rather than the 99% lead time.

If NVBOTS is looking at data from this report that it would like to convert into the 99% lead time, then equations Equation 5 and Equation 6 will suffice. If the 99% lead time for a particular setting is required, then the model will have to be run and the data will have to be fed into the MATLAB script as discussed in Section 5.1.1.

5.2 Push Analysis

5.2.1 Overview

The Push strategy is relatively straightforward and does not include any policy levers. However, the push model does give some valuable insights as it can be used to provide an estimation of capacity at the current facility. There is a clear point at which the demand is above capacity which allows an estimate of the capacity to be made. As is shown in this section, the capacity is estimated to be around 18 printers at the current facility. This is assuming a batch size of one and that workers spend 70% of their time on production.

5.2.2 Basic Mechanics

Figure 18 below shows the number of parts in the RPQ. In this graph, 15 printers are released every month (or every 9600 simulated minutes). These parts are then taken out of the RPQ and
onto the production floor. Note that with a demand of 15 printers / month almost half of the time is spent with no parts in the queue, indicating that there is spare capacity.

Figure 18: Released Parts Queue in a Push Model with a monthly demand of 15 printers.

Due to the variable nature of demand, and the naïve approach to forecasting in this model, it is expected that this policy will frequently lead to over or under production. This can be seen in Figure 19 below, which shows net content over time. Net content is defined as the number of printers available in the finished goods inventory minus the number of customers waiting for a printer. On average the sales per month is equal to the number of parts released per month. However, note that due to the random distributions, the net content can vary wildly. Over the long term this may lead to the right number of printers to be built, but in the short term demand can be much higher or lower than anticipated.
5.2.3 Capacity versus Worker Utilization

If demand is lower than the capacity, then there will be some idle time between the last printer finishing production and the next batch of parts being released onto the production floor. All printers are then cleared from the RPQ before the end of the month. However, once the demand is higher than capacity the production team will be unable to clear the RPQ before another set of parts is released. The result is that over time the parts accumulate in the RPQ. There is also a region between these two scenarios. If demand is roughly equal to capacity, then the RPQ is often cleared before the start of another month, but sometimes one or two parts will be carried over to the next month. This is due to processing times and inter-arrival times being randomly distributed instead of deterministic.

Figure 20 below shows these three scenarios. Note that there are three workers and that worker utilization is assumed to be 100%. When demand is at 25 printers per month the RPQ has a content of 0 for a significant portion of the total time. This indicates that the RPQ is cleared every month. If demand is 30, there are some months when all the parts are not cleared. As soon as the demand goes up to 31 printers per month the plot becomes distinctly different because there is a definite positive trend indicating that demand now exceeds capacity.

The maximum capacity of a Push system is defined as the demand level prior to the first instance of sustained increasing content in the RPQ system. In the example given it would mean that the maximum capacity is 30 printers / month.
The same procedure can be repeated for other levels of worker utilizations. Worker utilization is essentially the amount of time that workers spend on production, versus the time being spent on other things. It is modeled using a Mean Time to Failure (MTTF) and Mean Time to Repair (MTTR) in the model. Both these times are exponentially distributed, and the mean depends on the worker utilization level. The sum of the means is always equal to 480 minutes. For example, a worker utilization of 70% is modeled using a MTTF of 336 minutes, and a MTTR of 144 minutes.

For a more thorough analysis of worker utilization, readers are referred to Jain’s thesis [1].

The assumption is made that the worker utilization at NVBOTS is 70%. Unless stated otherwise this is the presumed worker utilization in the model.

![RPQ Content over Time](image)

*Figure 20: RPQ content over time in a push system for different demand scenarios. Including scenarios where demand is less than capacity (top), demand is roughly equal to capacity (middle) and demand is greater than capacity (bottom).*
The maximum capacity can therefore be found for other levels of worker utilization. Plotting the maximum capacity results in the graph as shown in Figure 21. A linear relationship may be expected, which gives a good approximation of the results with an adjusted R-squared value of 0.975. However, a quadratic model is technically a better fit, with an adjusted R-squared of 0.997. The resulting equation is given in Equation 7 below.

\[ Capacity = 15.0 U^2 + 14.5 U + 0.2 \]  \hspace{2cm} \textit{Equation 7}

A quadratic relationship appears unintuitive, as it implies that an additional percentile of worker utilization result in a larger increase in capacity when worker utilization is high, compared to when it is low. However, this phenomenon can be explained due to the disruptions caused by blocking or starving machines.

What is critical in the modelling worker utilization is that a worker will block other workers from assembling at a station if the first worker fails. If Worker A is assembling SA0101, and he is called somewhere else like a service job or a meeting (which is represented by the worker ‘failing’ in the simulation), then the production of SA0101 will not continue until Worker A is back. Worker B cannot finish this procedure for Worker A. This is a fairly reasonable assumption as some employees in the production team are often called away at short notice, and it is unusual for another worker to come in and finish the procedure he was working on. However, this disruption can be a problem if the production line is waiting for SA0101 to continue production.

The assumption of blocking other workers is important. Not only is an individual worker more likely to work when MTTF is high, but it also means that there is less chance for the rest of the
production system to become blocked or starved. Increasing the worker utilization is therefore beneficial by increasing the output of an individual worker, and by increasing the output of all other workers by decreasing the chance of them being blocked or starved. This explains the non-linear relationship.

What this means for NVBOTS is that a worker solely dedicated to production is of more value than two workers who spend half their time. However, this finding should be used with some caution, as the assumptions made in this simulation may not always hold true.

In reality workers will have some flexibility in deciding when to schedule meetings or go to a service call; it is not purely random. If a critical part has to be finished to continue production, then the meeting could be pushed back an hour. Or if a worker anticipates that he will have little to do on the production floor because he is waiting for other procedures to be completed, then during this time he can do other work like schedule meetings or fill out paperwork. Still it should be something to keep in mind for NVBOTS’ hiring plan; a full time production worker will lead to less disruptions compared with multiple workers who only spend a fraction of time on the production floor.

From Figure 21, and using NVBOTS’ estimates that worker utilization averages around 70%, the maximum capacity is roughly 18 printers per month. Manufacturing policies have little impact on maximum capacity. This figure should therefore hold true for the other policies as well.

5.3 Build to Order Analysis

5.3.1 Overview

In the BtO model the batch size plays an important role in determining the lead time and the inventory costs. This section will explore the relationship between batch size and these performance variables. Although the production team prefers to work in larger batches, the model indicates that this is sub-optimal.

The BtO model is one of the models brought forward to NVBOTS as a potential contender for a new manufacturing policy. The BtO model is used as a comparison against the CONWIP or CONWIP-BtO models Jain discusses in his thesis [1]. Note that the BtO model assumes that batching has no benefits; there is no improvement in processing time due to batching. This assumption was challenged during the meeting with NVBOTS, and is addressed in the advanced models in Section 6.
5.3.2 Basics

In a BtO model no finished goods inventory is expected; only printers that already have a buyer are produced. Parts will only be released into the RPQ when a full batch of sales is made. If the batch size is three, then no build is started until three new sales are made. Net Content will never be positive because printers will go straight from production to the customers and completely skip the finished goods inventory.

An example plot of Net Content over time can be seen in Figure 22 below. Here the batch size is one and the demand is set to 15 printers per month. Note that at times there are nine customers waiting for a printer but that there is never a surplus of printers.

Figure 22: Net content of the finished goods buffer over time for a Build to Order policy.

5.3.3 Batch size and Lead Time

There are two effects that determine how batch size impacts the lead time. First the aggregate results will be shown, after which the individual effects will be isolated to provide their intuitions.

As can be seen in Figure 23, there is a linear relationship between the batch size and the average lead time. As found in Section 5.1 this implies there is also a linear relationship between the batch size and 99% lead time. NVBOTS would like to keep their 99% lead time to within 20 days, which from Equation 5 would mean that the average lead time should be no greater than 12 days. As can be seen in Figure 23, in order to achieve this lead time, the model indicates that a batch size of five or less should be used when demand is 15 printers per month.
Figure 23: Average lead time versus batch size for a BtO system. Demand is 15 printers per month and the worker utilization is 70%.

However, as can be seen in Figure 24 the relationship between batch size and average lead time also depends on the demand. In this plot a batch size of five will not allow NVBOTS to reach an average lead time of 12 days if demand at 10 printers per month or lower. Note that the worker utilization is increased in order to allow a larger range of demand scenarios. The higher worker utilization would only decrease the lead time, meaning that if worker utilization is reset to 70% a batch size of five would still not work if demand is 10 printers per month.

Figure 24: Average lead time versus batch size in a BtO system with a monthly demand of 5, 10, 15, 20 and 25 printers.
Essentially there are two influences that batching has on lead times. The first effect is that the larger the batch size, the longer customers have to wait until the batch is filled up and parts are released into the RPQ.

Readers are reminded that for this policy parts will only be released into the RPQ when a full batch of sales is made. If demand is 20 printers / month, and assuming there are 20 workdays in a month, then on average a printer will be sold every day. If the batch size is six, then the first customer will on average have to wait five additional workdays before production even starts. This waiting time is named the batch fill-up time.

Figure 25 below shows the average batch fill-up time for a customer as the demand changes. This graph is plotted using a MATLAB script that is added in Appendix E. Multiple batch sizes are plotted. Note that if the batch size is one, then the batch fill-up time is zero. However, if the batch size is 12 then the average batch fill-up time can be as great as 22 workdays if demand is five printers per month. Customers have to wait this amount of time on average before their printer begins production.

![Average Batch Fill-Up Time vs. Demand in a BtO System](image)

*Figure 25: Average batch fill-up time versus demand for different batch sizes.*

Overcoming this drawback of batching is the main motivation for developing a Sell x Build y model, as discussed in the advanced models in Section 6.

The second effect is that building in a batch size of six versus a batch size of one means that any process (with the exception of the printing during quality control) will take six times as long. Customers therefore do not just wait on their own printer, but on five other printers being made. The result is that there is a significant extra waiting time per person for larger batch sizes.
Figure 24 is re-plotted with an adjustment for the batch fill-up time. This isolates the second effect. The resulting plot is shown in Figure 26.

Note the magnitude in Figure 26; the lead times drop significantly for the larger batch sizes. Secondly, the different demand scenarios do not cross over like they did in Figure 24.

There is still a difference between different demand scenarios in Figure 26. Higher demand scenarios result in longer lead time. This is because if demand is higher there is a higher chance that workers will not immediately start working on a new batch of printers, as they are still working on an old batch.

![Figure 26: Lead time versus batch size for numerous demand scenarios. This plot is adjusted to remove the batch fill-up time.](image)

The bottom line of this analysis is that increasing the batch size has a detrimental effect on the lead time.

### 5.3.4 Batch size and WIP

Besides lead time, the other important performance variable is the inventory level. For a BtO policy there should be no finished goods inventory, but there will be WIP.

As can be seen in Figure 27, there is a linear relation between the batch size and the average WIP. For a given demand level, increasing the batch size will increase the WIP. This is because there will be more parts at each station. The equations and adjusted R-Squared values are given in Table 4.
The relationship between batch size and WIP has a non-negative y-intercept. This arises from
the processes that do not scale with batch size such as the printing during quality control, and the
picking of parts (which is constant regardless of batch size).

Figure 27: Average inventory on the production line in a Build to Order system over different demand
scenarios and batches.

<table>
<thead>
<tr>
<th>Demand</th>
<th>Regression Model</th>
<th>Adjusted R-Squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 printers / month</td>
<td>$\text{Avg WIP} = 0.41 \times \text{BatchSize} + 1.32$</td>
<td>Equation 8</td>
</tr>
<tr>
<td>10 printers / month</td>
<td>$\text{Avg WIP} = 0.68 \times \text{BatchSize} + 1.99$</td>
<td>Equation 9</td>
</tr>
<tr>
<td>15 printers / month</td>
<td>$\text{Avg WIP} = 2.68 \times \text{BatchSize} + 2.27$</td>
<td>Equation 10</td>
</tr>
</tbody>
</table>

Table 4: Regression models of Figure 27.

In addition to WIP varying with batch size, WIP also varies with demand. If demand is very low
there will only ever be a single batch of printers on the production floor. In this scenario it will be
possible for a single worker to do the final assembly while the others are idle (as only one worker per workstation is allowed). Once demand starts to pick up there will be less idle time, and workers will need to start working on other batches before the first is done. Consequently, as demand increases the WIP will grow. However, there is a limit because only a single batch of WIP can exist at a workstation. Once demand grows beyond the maximum capacity the production floor becomes saturated. This can be seen in Figure 28 below.

From the analysis of the Push model in Section 5.2, it is expected that the maximum capacity is around 18 printers / per month if the worker utilization is set at 70%. This roughly corresponds to Figure 28, where the WIP levels off after demand exceeds around 18 printers/month.

![Average WIP versus Demand for Different Batch Sizes](image)

**Figure 28: Average WIP versus demand for a BtO system**

### 5.3.5 Service Items

There are a few items that are needed for servicing. For example, a new Hot End is required for a customer's printer if there has been a severe jam. These items need to be assembled before they can be taken out on a service call. Response time is critical for these kinds of items, and while a few weeks of lead time may be acceptable for printers, this will not be acceptable for service items.

It is assumed that the demand for service items is always twice the of demand for printers. With a printer demand of 15 printers per month, and therefore a service item lead time of 30 per month, the 99% lead time for service items is predicted to be around three days. This is relatively high
for service items. A CONWIP model for the service items would be better suited, and this is discussed in more detail in Jain’s thesis [1]. Using a CONWIP model increases the inventory slightly, but drastically improves the response times.

5.3.6 BtO Conclusions

The BtO model has one main lever; the batch size. The batch size has a detrimental effect on both the lead time as well as the average WIP. Increasing the batch size increases the lead time for two reasons. First, it increases the time that customers wait for a batch of sales to be made before the entire batch is released onto the production floor. This waiting time is referred to as the batch fill-up time. Second, the lead time increases with batch size because it takes longer to make six printers than one. Both effects combine to give a linear relationship between batch size and lead time. The average WIP also varies linearly with the batch size as a larger number of parts will be on the production floor at a time.

Both these relationships are also dependent on demand. That is, the linear relationship between lead time and batch size, or WIP and batch size, holds true for a given level of demand. When considering the lead time, it is beneficial to have a smaller batch size if the demand is low because it reduces the batch fill-up time.

Higher demand always has a negative impact on the WIP, as multiple batches will need to be worked on to keep up with demand. Once the demand exceeds the maximum capacity the average WIP plateaus.

NVBOTS should adopt a CONWIP model for the service items, even if it plans to adopt a BtO model for the rest of the production system. The added costs are negligible compared to the benefit of having near instant lead time on service items. For more analysis readers are referred to Jain’s thesis [1].

NVBOTS has set a target of keeping the 99% lead time to within 20 workdays. Based on this target, the maximum batch size should be less than six (note, this is assuming demand is 15 printers per month), and preferably as small as possible. From the perspective of WIP, it would be best to keep batch sizes as low as possible.

The results therefore indicate no benefits to batching. It has been argued by the production team that batching reduces the assembly time. This has not been included in the model at this time. The benefit of batching feature is added to the model in Section 6.
5.4 Presenting results to NVBOTS

Note that this section is shared between this and Jain’s thesis.

5.4.1 Summary

The results of the models discussed in this thesis, as well as the models discussed in Jain’s thesis [1] were presented to NVBOTS for feedback. Below follows a short summary. Note that this section is shared with Jain’s thesis [1].

The capacity at NVBOTS is roughly 18 printers per month. This is assuming the three workers spend, on average, 70% of their time on production. If all three workers spend 100% of their time on production, this can go up to 30 printers per month, however this means that the production team spends no time on servicing, meetings or paperwork, and is unrealistic. Based on the models, NVBOTS should look to hire additional employees when demand surpasses 18 printers per month.

The lead time requirements are not strict, and as a result a BtO system would be a good choice, as it typically has low WIP. However, there is a strong trade-off between the WIP and batch size, as well as the lead time and batch size. As a result, a low batch size is preferred. The model predicts that a batch size of one is the most efficient option.

The CONWIP-BtO system, as described by Jain [1], fails to provide much benefit over a pure BtO system. The CONWIP-BtO system has gained some support by the production team. However, there is no logical placement of the boundary between the CONWIP and the BtO systems that would result in favorable outcomes. The only place where it would make sense to place the boundary is very early on in the assembly process. This includes a significant proportion of the total costs, but a small proportion of total production time. The expected reduction in lead time is less than a day. It is argued that the added inventory levels are not worth a minor reduction in 99% lead time.

The CONWIP system, as described by Jain [1], predicts a 99% lead time of a few days. NVBOTS would have to allow an inventory of 24 printers once NVBOTS moves to its new facility in a few months, and an inventory of 12 printers at the current facility as a result of space constraints. This would be the best system for NVBOTS to adopt if it needs to be competitive on the lead time.
5.4.2 Feedback

NVBOTS chose to move forward with the BtO system, but expressed their desire to operate in batches of around six printers. Additional functionalities will be added to the model to simulate the benefits of batching, and to get a more accurate picture of the trade-off between batch size and the performance variables.

As described in Section 5.3, a major drawback of batching in the BtO model is the batch fill-up time. A Sell x Build y (SxBy) model is proposed as a counter to this drawback. Both of these additional functionalities will be developed and analyzed in Section 6 of this thesis.

In addition, NVBOTS expressed further interests in the CONWIP-BtO model, and asked whether it would make sense to adopt this policy if the boundary is shifted somewhere else. This policy is particularly interesting to NVBOTS as they are looking to bring numerous product lines onto the market, and are hoping to adopt a late-stage differentiation system for their production line. The development and analysis of the shifted CONWIP-BtO model is discussed in Jain’s thesis [1].
6. New Policies

This section adds additional features to the models. The benefit of batching is incorporated into the model, and a Sell x Build y policy is developed. This section addressed both the development and the analysis of these additional features.

6.1 Model Extensions

6.1.1 Extent of Batch Benefit

In the base models as described in Section 4, there is no advantage derived from batching other than a slight benefit given by spreading the picking time over a larger amount of parts. The picking time is assumed to be constant regardless of how many parts are picked.

The production team prefers to work in larger batch sizes, and claims that it has a significant benefit on the per-unit processing time. Hard data is unavailable, and no previous studies have been found. However, a per unit improvement on the order of 20%-40% is expected based on the experience of the production team.

The difficulty lies in determining the specific relation between processing time and batch size. Does the per unit processing time decrease by the same fraction when going from a batch size of one to a batch size of two, compared with the ratio if it goes from four to eight? This seems unreasonable, as it would imply that the benefit would never diminish. It is unrealistic to propose that the capacity of a plant can go to infinity as long as the batch size is sufficiently large.

It is pointed out that the main benefit of batching is that it removes a setup time. For example, take a bolt which is screwed into a subassembly. It would be faster to read and understand the procedure once, and to pick up a tool once, before then screwing in the bolts on five different subassemblies, rather than having to read the procedure and picking up the tool five times. Considering that there are over 400 procedures, and that these frequently change, it is reasonable to assume that the operators at least consult the procedures before doing each operation. Setup time is therefore not negligible.

Take an arbitrary procedure, with a total processing time of 1 min, and assume that the setup time is 30% of this 1min. Building in a batch of one would require 1 min, but building in batch sizes of two would take a total of 1.7 min (1min + 0.7min), or a per unit processing time of 0.85min. This is because the setup only has to be done once. Building in batches of three would require a total of 2.4 min, or 0.8min per unit. This calculation can be repeated assuming a setup time of 30%,
50% and 70%, for a batch size of one through to a batch size of 12. The results are plotted, and are shown in Figure 29 below.

![Batch Benefit as a Function of Batch Size](image)

*Figure 29: Batch benefit as a function of batch size assuming different setup times.*

Note that regardless of setup time, the per unit processing time quickly levels off. Using a batch size of three captures 67% of the theoretical benefits, while a batch size of six captures 83% of the total benefit. Therefore, there are diminishing returns. This is true regardless of what setup time is assumed. The percentage of the total benefit that is captured is shown in Table 5 below. It illustrates that it may not make sense to move from 10 to 11 printers in a batch as the improvement in processing time is negligible, but perhaps moving from a batch size of one to two or three would be beneficial.

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of Total</td>
<td>0%</td>
<td>50%</td>
<td>67%</td>
<td>75%</td>
<td>80%</td>
<td>83%</td>
<td>86%</td>
<td>88%</td>
<td>89%</td>
<td>90%</td>
<td>91%</td>
<td>92%</td>
</tr>
</tbody>
</table>

*Table 5: Percentage of total setup time that can be removed using different batch sizes.*

A setup time of 30% will be assumed, which means that if a batch size of two is used then the per-unit processing time should fall by 15%, and if a batch size of five is used the per-unit processing time should fall by 24%. Section 6.1.2 will discuss how these numbers are implemented in the model.
6.1.2 Batch Benefit Mechanism

Another functionality is added to the model, referred to as the Batch Benefit, or (BatchBenefit in the simulation). Batch Benefit is the percentage reduction of the per unit processing time. This takes the form of a variable in the model which can be adjusted. For example, to represent a batch size of two the Batch Benefit should be set to 15%. For a batch size of five it should be 24%. This can be calculated as explained in Section 6.1.1.

A trade-off between the average lead time versus the Batch Benefit in a BtO system is shown in Figure 30 below. For this run the service requirements are switched off, and the inter-arrival rate of demand is turned down to a deterministic value of 2000. This is done to isolate the effects of Batch Benefit. Note the linear relationship; if Batch Benefit goes from 0 to 20% the improvement in average lead time is 273 minutes, while if Batch Benefit goes from 20% to 40% the improvement is also 273 minutes. This behavior is as expected.

![Average lead time (min) vs. Batch Benefit in a BtO model](image)

*Figure 30: Average lead time versus the Batch Benefit.*

At zero processing time (100% Batch Benefit) there remains a lead time. This is because there are two factors which are not impacted by the Batch Benefit. The first is the printing time, which is part of the final quality control step and takes 933 minutes. The other factor is the batch fill-up time, the average time that customers wait for a sales batch to fill up. For Figure 30 the batch size is set to three and the inter-arrival rate is exactly 2000 minutes. Therefore, in this scenario the first customer will wait 4000min, the second 2000min, and the third 0min, to fill up the sales batch. The average is therefore 2000min.
If the Batch Benefit is 100% (processing time is zero), then the expected average waiting time would be 2000 + 933 = 2933 min. The data outputs an actual time of 2978min, which is within 1.5% of 2933. The discrepancy is attributed to random error. Therefore, the conclusion is that the Batch Benefit variable is working as expected.

6.1.3 Sell x Build y

Section 5.3 introduces the phenomenon of batch fill-up time. This is the amount of time customers spend waiting for other customers to fill up a batch of sales before the production process begins. The time spent waiting for other customers to fill up a sales batch can be a significant proportion of the total lead time.

An alternative policy is proposed to address the batch fill-up time. This policy is referred to as the Sell x Build y policy, or SxBy for short. For example, S3B6 would be to release a batch of six after three sales are made. A S3B5 model would be to release a batch of six when three sales are made. The remaining printers will be sold before another set of parts is released.

The BtO policy of Section 5 can be thought of as a SxBx policy.

The benefit of a BtO policy is a reduction in lead time. By adopting a S1B6 policy instead of a BtO policy with a batch size of six, the lead time is reduced by removing the batch fill-up time. This reduction could enable the production team to work in larger batch sizes and still meet the 99% lead time target of 20 days.

The drawback of a SxBy policy is that the finished goods inventory will no longer be zero as it is in a BtO model. The inventory costs are therefore higher for a SxBy model.

The SxBy model works by using a counter and a threshold. Every time a sale is made, a counter is stepped up. Once this counter meets the threshold (the ‘x’ in SxBy), a message is sent to the RPQ where a batch (the ‘y’ in SxBy) of printers is released onto the production floor. The counter continues until it hits the ‘y’ value, at which point it will reset to zero. This is to ensure that printers will not accumulate. The mechanism then repeats itself.

Figure 31 below shows a snapshot of the net content in a S1B6 model. Note that most of the time the net content is negative but that sometimes the net content is positive. This has an effect on the total inventory levels, but as will be shown the effect is almost negligible.
The following section will analyze the combined effects of the SxBy and the Batch Benefit additions to the model.

**6.2 New Policies Analysis**

**6.2.1 Batch Benefit Analysis**

Figure 32 below illustrates the benefit of building in batches. In blue the graph shows the average lead time (left axis, line) and the WIP (right axis, bar) for a BtO model without Batch Benefit. In red the same is shown but now includes the batch benefit.

The Batch Benefit causes the lead time to reduce by a few days or even a week when the batch size is high. The WIP reduces as well because the demand is the same, but the printers spend less time on the production floor.
Looking purely at this trade-off indicates that there still is no real benefit to batching. The lowest lead time and the lowest WIP are still achieved with a batch size of one. However, because the Batch Benefit reduces the processing time, it will increase the maximum capacity at the plant. For example, it is expected that a batch size of three versus a batch size of one would boost capacity by roughly 15%. This should be included in the considerations. However, before a decision is made on the batch size, the effects of a Sell x Build y model should be understood.

6.2.2 Sell x Build y Analysis

The analysis done in Section 5.1, which relates the average lead time to the 99% lead time, does not account for a SxBy policy. Therefore, it is unreasonable to take the average lead time as a proxy for the 99% lead time using Equation 5 or Equation 6. Instead the 99% lead time values should be used for any SxBy policy, meaning that the MATLAB code as explained in Section 5.1 will need to be run.

There are two levers in a SxBy model: the threshold (x) and the batch size (y).

The impact of the threshold is shown below in Figure 33. As the line-plot shows, the lead time increases as the threshold increases. However, there is only a minor change in the inventory levels as represented by the columns.

Figure 32: A BtO model showing the benefits of batching. Demand is set to 15 printers per month.
Two types of inventory are included in this measurement of average inventory; average WIP and average finished goods inventory. The WIP portion remains unchanged because the production team is working with a constant batch size of six. However, if the WIP portion of inventory remains constant, then it must mean the finished goods inventory also stays mostly constant. This may appear counterintuitive because if printers are made before demand arrives, then one would expect printers to be stored in the final printer inventory. This does happen, but on average this number is on the order of around 0.1 printers.

![Sell x Build 6 model for different levels of x](image)

**Figure 33: Average lead time and average inventory against various thresholds in a Sell x Build 6 model.**

Table 6 below shows the average content in the finished goods buffer for different thresholds and demand rates. As can be seen from this table, for every demand scenario the average finished goods inventory peaks when the threshold is set to one. Setting the threshold any higher reduces the average content. Also note that the average content is never higher than one printer, which means that the impact of thresholds on the finished goods inventory is small.

The threshold has little impact on the finished good inventory levels because most printers are sold before they have finished production. By the time the printers enter the finished goods inventory they will likely already have a buyer, meaning they will not spend any time in the finished goods buffer. Those that do stay in the finished goods buffer will be sold before another batch is released onto the RPQ. This means that at lower demand levels, the average content will be

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Average Inventory (of printers)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>1</td>
<td>5.0</td>
</tr>
<tr>
<td>2</td>
<td>10.0</td>
</tr>
<tr>
<td>3</td>
<td>15.0</td>
</tr>
<tr>
<td>4</td>
<td>20.0</td>
</tr>
<tr>
<td>5</td>
<td>25.0</td>
</tr>
<tr>
<td>6</td>
<td>30.0</td>
</tr>
</tbody>
</table>
higher. This can be seen in Table 6. However, even at low levels of demand the average content remains low.

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>\text{Average Final Printer Content (# of printers)}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>\text{Demand = 8 printers / month}</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>0.69</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>0.34</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>0.15</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>0.05</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>0.01</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0.00</td>
</tr>
</tbody>
</table>

*Table 6: Average finished goods inventory level in a SxB6 model.*

Some rudimentary calculations can verify this finding and help provide some intuition. Note that this calculation is highly simplified, as it assumes that workers are not held up by bottlenecks and that the work can be split evenly.

Assume a S1B6 policy. A printer can be built in roughly 10.5 workhours. With three workers, and a worker utilization of 70%, this means that a printer will take roughly five hours to build. Building six therefore takes roughly 30 man hours or about four days. On top of that printers have to go through quality control, and therefore have to print for an additional two days. The total is six working days or around 2,900 work-minutes. If the monthly demand is 15 printers per month, then this equates to 640 minutes per printer. Therefore, during the 2900 minutes of time required to produce a batch of six printers, 4.5 printers will be sold already. Considering that one was already sold at the beginning of this process, on average only 0.5 of a printer will be going to the finished goods inventory. Once this queue is empty once again, it will stay empty for around six days, until the next batch is received. The result is that the average content in the finished goods inventory remains very low.

This reasoning would suggest that the final goods inventory level is dependent on batch size as well as the demand level. Repeating the above calculation for a S1B12 model predicts that instead of 0.5 printers going into the finished goods inventory without a buyer, five unsold printers will go into the finished goods inventory.
Running the simulation confirms the intuition that demand has an impact on the finished goods content. Figure 34 below shows the tradeoff between various batch sizes and the average finished goods inventory level for different demand scenarios. Note that a S1By policy is the worst case scenario when it comes to finished goods inventory; increasing to a S2By or higher would reduce the average finished goods inventory level, for a given batch size.

As can be seen in Figure 34 a smaller batch size leads to a lower finished goods inventory. However even at larger batch sizes the inventory levels are quite low. NVBOTS is interested in batch sizes of around six, and although it would not be optimal, an additional printer in finished goods inventory will do little to impact NVBOTS bottom line.

![Figure 34: Average finished goods inventory level vs. batch size in a S1By Policy](image)

The lesson from this analysis is that if NVBOTS wants to adopt any SxBy policy it makes most sense for them to pursue a S1By policy. This results in a large benefit in terms of lead time, but the extra inventory level is minor, especially for smaller batch sizes.

### 6.2.3 Batch Size

As shown in Section 6.2.2, a S1By policy has the potential to reduce the lead time while adding almost negligible inventory costs. A threshold of one is recommended, as this allows the greatest reduction in lead time but adds little inventory costs. However, this does not aid in deciding the batch size.
Figure 35 shows the performance (99% lead time and average inventory) of a S1By policy and a BtO policy across various batch sizes. The Figure suggests that a S1By policy could be preferred compared to a BtO policy for a given batch size. By switching to a S1By policy the 99% lead time can drop by days or weeks, while leading to less than one additional printer in inventory.

The plots also indicate that there is no point of inflection nor a new minimum in the graph. Even when batch benefit is included, the model indicates no advantage to batching. Increasing the batch size beyond one is always detrimental to both the 99% lead time and the average inventory level.

![Performance of S1By model for different levels of y](image)

**Figure 35: Performance measures of a S1By model for different levels of y.**

The simulation therefore predicts that a batch size of one is most efficient. However, the price to pay for a higher batch size is not that steep. The simulation estimates that any S1By model up to a batch size of 12 will achieve the 99% lead time target of 20 workdays. Technically any batch size up to at least twelve would therefore satisfy NVBOTS requirements. Moving from a batch size of one to six only increases the expected inventory by three printers.

Figure 35 does not illustrate the improvement in capacity. It is expected that building in a batch size of 6 would increase capacity by 25% because all processing times are reduced by that amount. This is shown in Figure 36 below.
This same graph is drawn for the Push model (Figure 21, page 66) and predicted a maximum capacity of 18 printers per month assuming the worker utilization is 70%. In a S1B6 model the model predicts a maximum capacity of 23 printers per month, which is a 27.7% increase in capacity. It might be possible to go a little higher, but at that point the expected lead grows quickly. This is because a manufacturing system will become very volatile when demand is roughly equal to capacity. The increase in capacity is slightly larger than the predicted 25%, but this variation is attributed to the low resolution of one printer/month when determining capacity.

![Capacity versus Utilization in a S1B6 model](image)

*Figure 36: Capacity versus Worker utilization in a S1B6 model.*

In addition to this extra capacity, there may be other benefits to batching that are not captured by the model. For example, larger batches reduce the amount of paperwork that has to be filled in or larger batches may keep the production team more motivated because they prefer to work in larger batches. On the other hand, there may be costs to batching that have not been included like difficulties keeping track of all the different printers in a batch or having difficulties trying to find a place to work on twelve printers instead of on three printers.
7. Recommendations and Conclusions

This section summarizes the most important findings of this thesis as well as Jain’s thesis [1], and gives some suggestions for future work.

7.1 Summary of Findings

The model suggests that the current capacity at NVBOTS is around 18 printers per month if the batch size is one. This is assuming that the three workers in the production team each spend 70% of their time on production. The capacity can be increased when using larger batch sizes. For example, with a batch size of six the estimated capacity rises to around 23 printers per month. The capacity increases for higher batch sizes because the setup time for each procedure can be spread across a larger number of printers. This essentially causes the per-unit processing time to drop.

Due to the non-linear effects of disruptions it is more beneficial to hire a single worker who spends all of his or her time on production, compared to two workers only dedicating half of their time to production.

The simulations indicate that a BtO model with a batch size of one would give the best results in terms of inventory costs and lead time. Larger batch sizes are detrimental to lead time and inventory costs. This holds true even when considering the reduction in the per-unit processing time associated with batching, or when adopting a SxBy model to reduce the batch fill-up time. However, using a S1By policy, large batch sizes can meet the lead time requirement of 20 days. This comes at the cost of an increase in average inventory levels, but the additional inventory is relatively minor.

Batching increases the maximum capacity. A batch size of six increases capacity by roughly 27% in the model, to around 23 printers per month. This might delay the hiring of another employee, or allow the production team to spend more time on other work and is a significant benefit. Saving 25% in salary of the production team far outweighs the inventory holding cost of an additional few printers.

There are other factors relating to batch size that have not been considered. For example, at large batch sizes it will be more difficult to manage inventory, or manage the space on the production floor. Considering these tradeoffs, the relatively minor increase in inventory, and the large improvement in capacity, a batch size of six would be a reasonable option for NVBOTS to adopt.
If a larger batch size is decided on, it is strongly recommended to adopt a Sell 1 Build y policy. For example, a Sell 1 Build 6 policy if the batch size is set to six. The improvement in lead time associated with such a policy is substantial but it has very little impact on the total inventory. This is because most printers will likely be sold by the time they finished construction. For example, shifting from a BtO policy with a batch size of 6 to a S1B6 model drops the 99% lead time from 21 workdays to 11 workdays, while only increasing the average inventory level from 6.7 to 6.8.

Finally, regardless of the policy for printers, it is recommended to adopt a CONWIP policy for the service items. The lead time for these items is critical. A sufficiently low lead time can only be achieved using a policy where these subassemblies are held in inventory. As Jain describes, the added inventory costs are small [1].
8. Future Work and Other Problem Areas

This section details future work to be done to continue with this project. It also details other problem areas that were identified during the problem formulation. This section is shared with Jain’s thesis [1].

8.1 Future Work

Future work should include implementing a policy and collecting data to see whether observed values align with predicted values. There is insufficient time for this data to be collected by the MEng team, but it will be useful to verify the simulation. If observed values do not align with the simulation, then some of the assumptions may have to be reconsidered.

It may also be worthwhile to revisit the models once more information is known about the other product lines. As shown in Jain’s thesis [1] a late-stage differentiation policy is sensible. However, the batch size that NVBOTS should choose or the lead time it can expect may change depending on the design of the products. In general, the more similarity there is across the products, the better it will be for the manufacturing system.

8.2 Identification of Problem Areas

The initial survey of the company and the build of the two printers helped to identify potential problem areas at NVBOTS. The MEng team decided to focus on inventory management and manufacturing systems as it is believed there is substantial low hanging fruit for someone with expertise in this area. However, other problems are identified and discussed in this section.

Many of these other potential problems could each be the focus of a thesis, although some problems may be solved or the circumstances may have changed substantially by the time a new MEng group arrives.

In addition, each of these issues could prevent a startup in a similar position from going through a successful scale up. It is therefore worth detailing these obstacles in order to aid others in identifying similar problems at their companies.

Below the other obstacles are explained, and brought to a larger point.
8.2.1 Quality Definition and Quality Testing:

Product quality is incredibly important for a startup because low product quality will tarnish the reputation that a startup is trying to build. It may be possible to manually check every aspect of a product when only a handful are produced, however once the throughput increases this strategy will be infeasible. For example, it will simply take too much time to check every aspect of hundreds of printers. Besides having robust Standard Operating Procedures there needs to be a test that gives a company confidence that the product is of sufficient quality. This test needs to be effective and efficient. If the quality of the output of a production line begins to fall it needs to be addressed well before it causes issues for the customers.

At the onset of the project at NVBOTS, there did not exist a rigorous formal definition of quality used to quantify the ‘goodness’ of a printer. Defining the quality of a printed part relied mostly on visual inspection of it. While no serious complaints have been made, the quality of printers coming off the production floor was mostly unknown.

Did the design change impact the ability of the printer to print high quality parts? Does the quality of printer drop as production is ramped up? These kinds questions are unanswerable, and could expose NVBOTS to risk.

A project on this topic would include the development of quality control tests. In addition, the project scope would include the development of a formal framework to continuously monitor and try to improve quality issues.

The lesson for other companies is to set a standard for their products. What is considered good enough to be sold, and what is not good enough to be sold? What tests can be done, or what measurements can be taken, to find out whether a particular product is good enough? What should happen once a mistake is identified?

Note that since the beginning of the project NVBOTS has made significant steps to address this issue.

8.2.2 Procedures which are Difficult to Scale

Many parts of the company will necessarily need to scale when more sales are made. This will lead to increased expenditure in some areas. For example, a larger manufacturing team will be required in order to achieve a significantly higher throughput. However, for some aspects the increase in sales will not lead to commensurate costs, but in much higher costs.
There are some procedures or policies that are more difficult to scale than others. For example, some procedures take up a lot of space, are expensive or take a lot of time. Scaling this up by one or two order of magnitude may take up so much space that another facility is required. Or it may be prohibitively expensive, eating up all revenue. Or it may require a lot of time of a particularly valuable member of the company.

Some examples of practices at NVBOTS which are difficult to scale include filament calibration, servicing, shipping and the 3D printing of production parts. These are discussed below.

**Filament Calibration:** An example of a process at NVBOTS that does not scale is filament calibration. This is a job that the VP of engineering has taken responsibility for. The process is complex, requires experience, and does not have a set procedure. The problem in short is as follows: NVBOTS’ printers are based on the FDM additive manufacturing process. The printer draws a plastic filament from spools, which are sourced from a supplier. There is notable variation in material characteristics from batch to batch, and compensating for this variation requires calibration of the settings. Calibration is a very time consuming process.

Every few weeks this process can take a few hours, or potentially a few days. Currently this is not an issue, as the benefit far outweighs the cost. However, if the demand for filament increases this process will need to be done more frequently. This is problematic. The production floor uses filament that needs to be calibrated and if the calibration is delayed it also delays production. In addition, the calibration could start taking so much time that the VP of Engineering will have little time to do other things. The VP of Engineering is one of the most valuable engineers at the company, and it would be much preferred if he could spend more time on R&D or other projects.

NVBOTS should find an alternative to the current process for calibrating filament. This can either be in the form of defining a procedure for filament calibration so that other people can work on it, or by removing the need for manual calibration in the first place. A project in this area could go via either or maybe both routes.

**Servicing:** NVBOTS sends out its own personnel for servicing. NVBOTS’ servicing capabilities would have to scale up roughly linearly with the number of active NVBOTS printers. If NVBOTS has customers all over the US, it will require long and expensive trips to personally provide servicing. As an alternative, an outsourcing strategy could be pursued.
This project involves identifying the needs of NVBOTS with regards to third party servicing companies, and would result in making recommendations to NVBOTS. It could also build upon a previous thesis written at NVBOTS on the topic of servicing and the costs of servicing [19].

**Shipping:** As NVBOTS expands its distribution to areas far outside of Boston its shipping methods will become ineffective. Currently printers are typically delivered by a member of the sales or servicing teams at NVBOTS. However, this becomes infeasible when delivering to remote places in the US. A preferable strategy may be to hire a delivery company.

At the beginning of the project NVBOTS did not have a structurally robust shipping container for safely shipping its printers overseas and had to be addressed. In addition, different shipping strategies or companies should be considered and weighed against each other.

**3D Printed Parts:** NVBOTS currently uses about 20 printed parts in its own printer. This has allowed the engineering department to be flexible and to quickly make changes to the design. In addition, many of these parts are complex and would be expensive to fabricate using conventional manufacturing methods. However, by relying on parts printed in house NVBOTS exposes itself to some risks. Whenever a new filament batch is used there are often problems with warping or other printing errors caused by incorrect calibration. This disrupts the production line. In addition, while printing with the NVPro requires little supervision it still requires an operator for some aspects of the process. For example, the filament spools need to be replaced and some parts need post-processing.

There will be a point where it no longer makes sense to print all 20 parts. Instead, all or some of these parts could be sourced from a supplier. It is not certain when NVBOTS should change these parts, but the current methods do cause disruptions on the production floor. In this project a cost benefit analysis would be done to determine when, or if, NVBOTS should switch over to conventional manufacturing methods for these parts.

Just like the four examples above, other companies should identify practices that will take much more time after a scale up. They should then decide whether it is still worth using the same methods in the future, or whether a change should be made.

### 8.2.3 Product Overhauls

During a scale up there will be a lot of extra work to do like training more people and preparing to move into other markets. Furthermore, it is difficult to manage both an increase in the throughput of a production floor and the implementation of engineering changes to the product. This means
that jobs like major product overhauls will likely be postponed until the later stages of the scale up. However, not making these changes can be costly, as it may reduce efficiency and it may become more expensive or convoluted to change things at a later stage.

Releasing a major design revision before scaling up should therefore be considered. The main contender for this at NVBOTS would be a Design for Assembly project. There are many instances of inconvenient and difficult assembly steps that sometimes required several attempts to do correctly. This substantially increases the printer assembly time, which could adversely affect the production rate during ramp up.

Significant improvements have already been made with the release of the Rev G. This project is therefore not included with filament calibration, servicing or sales as something which will become excessively expensive if the same practice is scaled up. However, the points made are still important to note for other startups and it is therefore included here.

8.2.4 Standard Operating Procedures & Engineering Change Orders
At the onset of the project the Standard Operating Procedures for the assembly of a printer was in its nascent stage. The lack of proper standardization could impact the quality of printers by increasing the variation from printer to printer. If one employee does a procedure differently from another, then if an issue arises it will be difficult to track what caused the problem or what other printers may have the same issue. In addition, one of the methods of doing a procedure may be significantly more efficient than another. It therefore makes sense to standardize assembly procedures.

If a change is made to a part or a method of assembly, it should be reflected in the procedures. However, these changes should not be implemented continuously. Changes should be batched together, and released as a set. This implies that Engineering Change Orders will need to be considered. Engineering Change Orders is a term used to describe the system for making engineering changes. A startup is flexible, which is one of its great strengths as it allows for quick changes in the product after receiving feedback. However, frequent changes are undesirable from a production perspective as it causes disruptions. By standardizing the implementation of changes through Engineering Change Orders, the production floor should be minimally impacted while still allowing the product to be easily adjustable.

Addressing this problem area would involve implementing a system for the writing of new procedures, and would also address Engineering Change Orders.
8.2.5 Problem Identification Summary

This section gives some examples of issues that should be considered by a startup in order to prepare for a scale up. The main points include defining quality, identifying and addressing procedures that will not scale well, considering doing product overhauls before scaling up, and defining Standard Operating Procedures and Engineering Change Orders.

The focus of this section has been predominantly on manufacturing related obstacles. However, there are many other issues that should be considered by a startup looking to scale. For example, it does not make sense to scale up if there is no market to expand into or if no further sales can be made. Additionally, the finances of a company are something that will need to be given sufficient thought. Note that these concerns are not addressed here.
References


Appendix A: Simulating Demand

This Appendix details attempts at simulating demand in such a way that both the mean and variance can be adjusted. These attempts ultimately did not result in a usable method, and instead a simple exponentially distributed inter-arrival time was used. The process is detailed here to prevent others from repeating the same work.

A.1 Challenge of Simulating Demand

The distribution of demand for a product greatly influences the optimal manufacturing policy. A highly variable demand will require a different approach compared to very steady demand. However, there is little usable demand data that can be used to determine the distribution of demand at NVBOTS. A solution would be to run a highly variable demand scenario, as well as a steadier demand scenario, and to see what policy performs well in both instances. The outcomes could then be compared to see whether a particular policy is highly susceptible to variable demand. This would be some sort of sensitivity analysis. The ideal policy would perform well regardless of the variability.

The objective of this section is to find out how to model different demand variability levels. However, as will be shown, no solution is ultimately found to suitably model the desired variabilities.

The assumption is to have an average of, say, 25 printers / month, normally distributed, with a Coefficient of Variation (CV) of 0.2 for the low variability scenario, and a variation of 1.0 for the high variability scenario. CV is defined in Equation 11 below, where $\sigma$ is the standard deviation and $\mu$ is the mean. Equation 12 and Equation 13 are then the low and high variable demand scenarios respectively. Note that these units are in printers/month.

\[
Coefficient of Variation = CV = \frac{\sigma}{\mu} \quad \text{Equation 11}
\]

\[
Demand \sim N(25,5) \quad \text{Equation 12}
\]

\[
Demand \sim N(25,25) \quad \text{Equation 13}
\]
This needs to be translated into the units of minutes per printer to make sense in the model. If on average 25 printers are sold per month, then the average number of minutes per printer can be calculated. It is assumed that there are 20 workdays in a month, with 8 hours per workday, and 60 minutes per hour. That means there are a total of 9600 minutes per month. Selling one printer would then, on average, require 384 minutes.

The issue lies with the CV; what does a CV on the printers/month level translate into on the minutes/printer level? No literature was found on this problem, and an analytic solution was not immediately obvious. A simulation based approach was chosen as an alternative to solving this analytically, and has produced unexpected results. The MATLAB code used has been documented and added to Appendix B for future engineers running into similar problems.

Note that from here onward the CV on the printers/month level will be referred to as CV$_{\text{month}}$, and CV on the minutes/printer level will be referred to as CV$_{\text{minute}}$. Note that the CV$_{\text{minute}}$ is what is put into the main simulation, and that this should result in a CV$_{\text{month}}$ of 0.2 or 1.0 when looking at the number of printers sold each month.

### A.2 Translating CV$_{\text{minute}}$ to CV$_{\text{month}}$

Instead of converting CV$_{\text{month}}$ to CV$_{\text{minute}}$ analytically, a simulation was run in MATLAB. As mentioned, the relevant code has been added to Appendix B. What the code does is determine how many events are generated in one month (9600 minutes), assuming that the time between events is normally distributed, with a mean of 384 minutes, and a varying CV. This simulates the generation of sales. The resulting graph is shown in Figure 37.
Note that there appears to be a linear relationship between CV<sub>minute</sub> and CV<sub>month</sub>. A regression analysis is shown in Table 7. The resulting equations are shown in Equation 14 and Equation 15. The regression implies that there is roughly a factor of five difference between CV<sub>minute</sub> and CV<sub>month</sub>. The adjusted R<sup>2</sup> of the regression is 0.997, which confirms that this is a good fit.

![Figure 37: CV<sub>minute</sub> on the x-axis versus the corresponding output CV<sub>month</sub> on the y-axis.](image)

Note that there appears to be a linear relationship between CV<sub>minute</sub> and CV<sub>month</sub>. A regression analysis is shown in Table 7. The resulting equations are shown in Equation 14 and Equation 15. The regression implies that there is roughly a factor of five difference between CV<sub>minute</sub> and CV<sub>month</sub>. The adjusted R<sup>2</sup> of the regression is 0.997, which confirms that this is a good fit.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>t-Stat</th>
<th>p-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>0.012539</td>
<td>0.004235</td>
<td>2.9606</td>
<td>0.00476</td>
</tr>
<tr>
<td>x1</td>
<td>0.1923</td>
<td>0.001445</td>
<td>133.04</td>
<td>2.69E-63</td>
</tr>
</tbody>
</table>

Table 7: Regression analysis of CV<sub>minute</sub> vs. CV<sub>month</sub>

\[
CV_{month} \approx 0.1923CV_{minute} + 0.0125
\]

Equation 14

\[
CV_{minute} \approx 5.200CV_{month} - 0.065
\]

Equation 15
If a $CV_{\text{month}}$ of 0.2 is required, a $CV_{\text{minute}}$ of 0.97 should be fed into the simulation. Similarly, if a $CV_{\text{month}}$ of 1.0 is required, a $CV_{\text{minute}}$ of 5.14 should be used.

So far these are useable results; however, the assumption of a normally distributed inter-arrival time is invalid, as it allows negative numbers. This should be adjusted, and is discussed in the next section.

### A.3 Curtailed Distributions

It is impossible to have a negative time between events. FlexSim sets any arrival time less than zero to zero, and the same policy will be used here. In later parts of the analysis an alternative truncation policy is also considered.

Most of the time the truncation is of little concern. For example, in Figure 38 below there are two normal distributions (based on a sample of $10^6$), with a mean of 384 and a CV of 0.1. The left has all negative values remaining as they were. The right is truncated: it has all negative values set to zero. There is no noticeable difference between these two charts because there are so few negative values.

![Normal Distribution](image1.png)

*Figure 38: Normal distribution (left) and truncated normal distribution (right), with a CV of 0.1 and a mean of 384. Note that there is no discernable difference between the two distributions.*

However, if the CV increases, the fraction of negative numbers increases and the differences between a truncated and non-truncated plots become discernible. The above distribution is run with a CV of 0.3, and the results are as shown in Figure 39. Note that there is now a clear cut-off:
the truncated distribution has become skewed. Increasing the CV even further has a dramatic influence on the distribution.

According to Equation 15, if a $CV_{\text{month}}$ of 0.2 is to be simulated, then the $CV_{\text{minute}}$ should be set to 1.0. The above distributions are redrawn, but this time with a mean of 384 and a CV of 1.0 to represent the highly variable demand scenario. The results are as shown in Figure 40. Note that for the highly variable demand scenario, where the $CV_{\text{minute}}$ would be 5.14, this effect would be even more exaggerated.

Figure 39: Normal distribution (left) and truncated normal distribution (right), with a CV of 0.3 and mean of 384. Any negative values are set to zero.

Figure 40: Normal distribution (left) and truncated normal distribution (right), with a CV of 1.00 and mean of 384.
It can be argued that there is nothing inherently wrong with having a distribution that deviates from normal. However, these deviations also influence the observed mean and standard deviation (and therefore the CV). The mean will increase because any highly negative values are set a larger value of zero. This skews the mean to be greater than it would be before truncation. In addition, all the extreme values on one side of the distribution are now put closer towards the mean, therefore reducing the CV.

Using MATLAB, the truncated normal is analyzed for its actual mean and actual CV. The results are shown in Table 8. Note that Table 8 also shows the result for an alternative truncating policy. In this alternative policy, if a number is found to be negative the random number generator is run again. The original policy will be referred to as Set to Zero, while the alternative is referred to as Re-Throw.

The purpose of Table 8 is to show the effects of truncation by looking at the target or input CV and mean, and the resulting actual or output CV and mean. The difference is substantial. For example, for the high variability scenario using a Set to Zero policy the mean goes up by 160%, and the CV drops by a factor of four. Also note that the re-throw policy is strictly worse than the original policy. The Re-Throw policy leads to a larger shift in mean and a larger shift in the CV.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Truncate policy</th>
<th>CV$_{\text{month}}$</th>
<th>CV$_{\text{minute}}$</th>
<th>Target CV</th>
<th>Target Mean</th>
<th>Actual CV</th>
<th>Actual Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Variability</td>
<td>Set to Zero</td>
<td>0.2</td>
<td>0.97</td>
<td>0.97</td>
<td>384</td>
<td>0.79</td>
<td>413</td>
</tr>
<tr>
<td>High Variability</td>
<td>Set to Zero</td>
<td>1.0</td>
<td>5.2</td>
<td>5.2</td>
<td>384</td>
<td>1.3</td>
<td>1002</td>
</tr>
<tr>
<td>Low Variability</td>
<td>Re-Throw</td>
<td>0.2</td>
<td>0.97</td>
<td>0.97</td>
<td>384</td>
<td>0.61</td>
<td>487</td>
</tr>
<tr>
<td>High Variability</td>
<td>Re-Throw</td>
<td>1.0</td>
<td>5.2</td>
<td>5.2</td>
<td>384</td>
<td>0.73</td>
<td>1740</td>
</tr>
</tbody>
</table>

*Table 8: Summary table of the effects of truncation on the mean and CV of a distribution.*

Because negative inter-arrival times are impossible, truncation is required. However, this truncation changes the mean and standard deviation of demand. For example, if a mean of 385 minutes (25 printers per month) is fed into the simulation, then for a highly variable demand
scenario the observed mean will be 1002 minutes (or 9.6 printers per month). The CV becomes 1.3 instead of the target 5.2, which translates into a $CV_{\text{month}}$ of 0.26 instead of the target $CV_{\text{month}}$ of 1.0. The input CV and mean should therefore be adjusted to account for the effects of truncation.

A.5 Adjusting the Normal Distribution

In section A.4 it is shown that the normal distribution representing the arrival time between printers has to be adjusted in order to result in the desired monthly demand and variation. Negative arrival times are nonsensical, but by removing the negative values, the mean and CV are impacted.

The first question is what the truncation does specifically. What is the relation is between the input statistics and output statistics when the normal distribution becomes truncated? This is done for both truncating policies. A MATLAB code is used to investigate these relations, and is added as Appendix C.

Figure 41 below shows the tradeoff between the input mean and the resulting mean after a Set to Zero policy. Notice that it is linear, and that the slope is roughly 1.26. CV is kept constant at 1.0. What this plot implies is that in order to simulate an inter-arrival time of 384 minutes, an average of 305 minutes (384 divided by 1.26) should be put into the simulation. Truncating a normal distribution with a mean of 305 and a CV of 1.0 will then result in a mean of 384 minutes.

![Input mean vs. Actual mean after truncating, Set to Zero policy](image)

*Figure 41: Input mean vs. Output mean, using the Set to Zero truncating policy. The slope of the curve is 1.26.*
Figure 42 shows the tradeoff between the input CV and the resulting CV after truncating with the Set to Zero policy. The mean is held constant at 305 minutes. Notice that the curve reaches a limit well before an actual CV of 1.5. This is an issue, because for the high demand scenario the actual CV (CV\textsubscript{minute}) should be 5.2, in order to translate into a CV\textsubscript{month} of 1.

![Input CV vs. Actual CV after truncating. Set to Zero policy](image)

*Figure 42: Input CV vs. resulting actual CV. The red horizontal line indicates where the actual CV should ideally be for a low variability scenario (CV\textsubscript{minute} = 0.97), and the green vertical line shows what the CV should therefore be before it is fed into the truncating policy. Note that reaching the high demand scenario (Actual CV = 5.2) is impossible.*

This process can be repeated for the alternative truncating policy where instead of setting a negative number to zero, the random number is re-rolled. The results are shown in Figure 43 and Figure 44 below. Note that Figure 43 has a lower slope than Figure 41, and that Figure 44 hits its limit earlier than in Figure 42.

![Input Mean vs. Output mean. Re-throw policy](image)

*Figure 43: Input mean vs. Output mean, using the Re-throw policy. The slope of the curve is roughly 1.08.*
Figure 44: Input CV versus output CV, using the Re-throw policy. Note that a limit is reached somewhere around an effective CV of 0.8. This is insufficient to simulate a low variability scenario.

What these graphs tell us is that it is not possible to increase the effective CV after truncation to beyond 1.5. From Figure 37, this then tells us that the CV cannot go above roughly 0.3.

In other words, using a normally distributed inter-arrival time to simulate high and low variability in demand is not possible. This approach will not result in the desired variability. Our process has been documented here for others coming across similar problems.

A.6 Alternative Strategies

As discussed in the previous sections, a normally distributed inter-arrival rate is not suitable for the needs of the simulation.

Some other strategies are pursued, like taking a random number, X, from a normal distribution with mean of 25, and CV of CV. Then divide 9600 by this number to give the time between printers required to make X printers per month. However, there are limits to this strategy due to X needing to be positive, and not small. Ultimately it is not possible to pull any levers to change the CV, and as a result this method is discarded as well.

In the model an exponential distribution will be used, with the input variable being the average number of minutes per printer. This would then be run for multiple demand scenarios; like 10, 15, 20 and 25 printers per month. These distributions result in statistics as shown in Table 9 below. Table 9 also shows the effective CV, which is estimated using a simulation with a
sample size of 100,000. Note that the CV is roughly 0.2-0.3, which is a medium level for CV and will have to suffice for the purposes of the model.

<table>
<thead>
<tr>
<th>Monthly Demand</th>
<th>Min/printer</th>
<th>Estimated ( CV_{month} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>960</td>
<td>0.31</td>
</tr>
<tr>
<td>15</td>
<td>640</td>
<td>0.26</td>
</tr>
<tr>
<td>20</td>
<td>480</td>
<td>0.22</td>
</tr>
<tr>
<td>25</td>
<td>384</td>
<td>0.20</td>
</tr>
</tbody>
</table>

*Table 9: Monthly demand versus the input parameter and the resulting CV*
%This code allows us to translate an input (min) CV to an output CV (month) and then we can reverse it to find what to put into the simulation.

clc;
clear;

%This is the target CV for Printer/month. This can be adjusted to test what CV is required on the minute/printer level, to result in a 'MonthCV' as set here.
MonthCV = 0.2;

%Defines the range of CV to test. These should be
CVinputStart = 0.1;
CVinputStop = 5;
CVSteps = 0.1;
%Creates a matrix based on the previous inputs
Xaxis = CVinputStart:CVSteps:CVinputStop;

%This matrix will log the data that will be plotted
MasterData = [size(Xaxis),1];
MasterCounter = 1;

%for different input CVs, run the
for CV = CVinputStart:CVSteps:CVinputStop;
S = 100;
Xmean = 384;
Xstdev = CV*Xmean;

%Sets counters and matrices to 0 again
time = 0;
counter = 0;
j=0;
A = zeros(2000,1);

%Get a large sample size
for i = 1:2000

    %the main loop. Essentially it calculates how many printers (counter)
    %are sold in 1 month if the time between printers is
    %normally
    %distributed with a mean of Xmean, and a st.dev of Xstdev.

    while time < 9600; %9600 minutes in a month
        X = normrand(Xmean,Xstdev);
        time = time + X;
        counter = counter + 1;
    end

    A(i) = counter-1; %Rounds down the no. of printers sold in a month
    counter = 0;
    time = 0;
end

%Calculations and data collection
AverageA = mean(A);
AverageStdev = std(A);
CVoutput = AverageStdev/AverageA;

MasterData(Mastercounter) = CVoutput;
Mastercounter = Mastercounter + 1;
end

%Line of best fit for mean.
[t f] = polyfit(Xaxis,MasterData,1);
m = t(1);
c = t(2);
y = m*Xaxis + c;

%Linear regression.
fitlm(Xaxis,MasterData)

%plot the figure
figure
hold on
plot(Xaxis,MasterData)
plot(Xaxis,y)
xlabel('Input CV (CV_m_i_n_u_t_e)')
ylabel('Output CV (CV_m_o_n_t_h)')
title('CV of min/printer (input) versus CV of
printers/month (output)')
hold off

%This is what the CV should be on the minute level to get
an output of
%TargetCV on the month level
TargetCV = (MonthCV / m) - c
Appendix C – MATLAB code shifted mean and CV

% How does the normal distribution do when truncated?
% This model essentially outputs a graph relating the input mean to the output mean. This then gives us the ability to say that if we want the output to be some number (i.e. the TargetMean), what should the input be?
% First the CV is chosen. This is because as it turns out, there is a linear relationship between the output mean, and the input mean, for a given input

%Clears the workspace
clc
clear

%input parameters
TargetCV = 0.98;
TargetMean = 384;
TargetStDev = TargetCV * TargetMean;

InputCV = TargetCV;
InputMean = TargetMean;
InputStDev = TargetStDev;

%Define variables
A = []; %storage matrix for random numbers
Avg = 0;
stDev = 0;
CollectMat = [];
j=0;
k=0;
l=0;
Repeat what we did for the mean above, but now do it for CV.
InputCVMat = 0.1:0.05:10;
j=0;
InputMean = TargetMean;
B=[];

for InputCV = InputCVMat;
    j=j+1;  % row counter
    InputStDev = InputCV * InputMean;

    %Main loop taking a whole bunch of random numbers, truncating them
    %if needed, and then storing it in matrix A
    for i = 1:1:100000;
        X = normrnd(InputMean,InputStDev);
        if X <0
            X = 0;
        end
        B(i) = X;
    end

%Analyse data and store Data
stDev = std2(B);
Avg = mean(B);
OutputCV = stDev/Avg;
Matrix2(j) = Avg;
MatrixCV(j) = OutputCV;
end

%Find the corresponding CVinput
C = MatrixCV > TargetCV;
D = MatrixCV(C);
loc = find(MatrixCV==D(1));
CV2Use = InputCVMat(loc)

% Main loop for the average.
InputMeanMat = linspace(0,TargetMean,20);
InputCV = CV2Use;
Matrix=[];
j=0;
for InputMean = InputMeanMat;
    j=j+1; % row counter
    InputStDev = InputCV * InputMean;

    %Main loop taking a whole bunch of random numbers, truncating them
    %if needed, and then storing it in matrix A
    for i = 1:1:100000;
        X = normrnd(InputMean,InputStDev);
        if X <0
            X = 0;
        end
        A(i) = X;
    end

    %Analyse data and store Data
    stDev = std2(A);
    Avg = mean(A);
    OutputCV = stDev/Avg;
    Matrix(j) = Avg;
end

    %Line of best fit for mean
    [t f] = polyfit(InputMeanMat,Matrix,1);
    m = t(1);
    c = t(2);
    y = m*InputMeanMat + c;

    %Model's Mean
    TheMean2Use = (TargetMean - c)/m

    %plot figure for the average
    figure
    hold on
    plot(InputMeanMat,Matrix)
    plot(InputMeanMat,y)
    xlabel('InputMean')
    ylabel('Actual Mean')
    title('Input mean vs. Actual mean after truncating. Set to Zero policy')
    hold off
%plotting the CV
figure
hold on
plot(InputCVMat, MatrixCV)
plot([0 5], [TargetCV TargetCV])
plot([CV2Use CV2Use], [0 2])
xlabel('Input CV')
ylabel('Actual CV')
title('Input CV vs. Actual CV after truncating. Set to Zero policy')
hold off
Appendix D – Calculating the 99% Lead Time

% This model outputs the lead time associated with the Service level.
% In the flexsim model, go to the global table: Time_Tracking. Copy this data, and paste it in an Excel file called 'Data.xlsx'. Save this file in the same folder as this matlab code, and then run this code.
% Note that the code will clean up any non-numbers, and any zeros. For this reason we added a 0.1 min processing time at the very end of the manufacturing line to ensure that even orders which are filled instantly are considered in this matlab script.

clc;
clear;

% Define variables
ServiceLevel = 0.99;
plot = 0; % plotting on (1) or off (0) ?

% read data
A = xlsread('Data.xlsx','Sheet1');
A=A(:); % into 1 column
A(isnan(A))=[]; % removes NaN
A=A(A~=0); % removes zeros

% Sort the list from small to large
SortedA = sort(A);

% Find where the ServiceLevel corresponds to
Size = size(A);
index = ceil(Size(1)*ServiceLevel);

% get lead time
LeadTime = SortedA(index);
LeadTimeDays = LeadTime / 480;
%average lead time
AvgLeadTime = mean(A);
AvgLeadTimeDays = AvgLeadTime/480;

%Size of line
Length = size(A,1)/4;

if plot == 1

%Plot the data
figure
hold on
hist(SortedA,15);
plot([LeadTime LeadTime],[0 Length], 'k--','LineWidth',2)
xlim([0 max(A)])
xlabel('Wait time (min)'
ylabel('Number of occurrences')
title('Histogram showing the distribution of wait times')
ServiceLevel = ServiceLevel*100;
legend('Wait time distribution',['the ' num2str(ServiceLevel) ' Service level'])
hold off
end

Days = [AvgLeadTimeDays LeadTimeDays]
Minutes = [AvgLeadTime LeadTime]
Appendix E – Batch Fill-up Time

%The code calculates the batch fill-up time
clc; clear;

%Define demand interval
X = 5:1:25;
j=0;

%main calculation loop
for batch = [1 3 6 12] 
    j = j+1;
    k = 0;
    for demand = X
        k=k+1;
        Q=zeros(0,0);
        iat = 9600/demand;
        waittime = -iat;
        for i = 1:batch
            waittime = waittime + iat;
            Q(i) = waittime/480;
        end
        avgTime = mean(Q);
        A(k,j) = avgTime;
    end
end

%plotting the results
figure 
hold on
plot(transpose(X),A(:,4),'--','Linewidth',2)
plot(transpose(X),A(:,3),'-.','Linewidth',2)
plot(transpose(X),A(:,2),'Linewidth',2)
plot(transpose(X),A(:,1),':','Linewidth',2)
legend('Batch size = 12','Batch size = 6','Batch size = 3','Batch size = 1')
xlabel('Demand (printers/month)')
ylabel('Average Batch fill-up time (workdays)')
title('Average Batch Fill-Up Time vs. Demand in a BtO System')
ax = gca;
ax.YTick = linspace(0,24,13);
hold off

DataMat = [transpose(X) A];