Understanding and detecting root causes of inventory record inaccuracy in a Just-In-Time supply chain

Dekker, Roy H.G.

Award date:
2023

Master Thesis

Operations Management & Logistics

Understanding and detecting root causes of inventory record inaccuracy in a Just-In-Time supply chain

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Author: R.H.G. Dekker (Roy) - 1329839
1st Supervisor: K.H. van Donselaar
2nd Supervisor: L. Genga
Supervisor Picnic: G.L. Basso
Eindhoven, October 4, 2023

Abstract

This thesis explores the concept of inventory record inaccuracy (IRI) resulting in the inability to meet customer demand in the context of an online grocery retailer. By analysing the fulfilment process of its manual fulfilment centres, several potential operational root causes were identified. These operational root causes consist of errors made during the fulfilment subprocesses of receiving, replenishing/moving stock, counting, picking, and clearing/adjusting stock to errors that are not related to one single fulfilment subprocess. To confirm the identified potential root causes and gain a better understanding of their distribution, an automated root cause allocation was developed in the form of a rule-based expert system. This system applies expert knowledge captured in rules to historical inventory data such that the hypothesized potential root causes can be accepted or rejected for incomplete orders that occurred. Subsequently, two corrective solution approaches were explored that focus on detecting situations in which IRI is likely created instead of common counting procedures that focus on aggregate inventory data. First, some of the developed rules for the automatic root cause allocation could be converted into rules for creating real-time alerts that ask employees to check whether the root cause occurred and correct the IRI if it did. The maximum potential of this solution is expected to be a 14% reduction of incomplete orders due to IRI (belonging to incomplete cases due to IRI having at least 5 incomplete consumer units). Second, a logistic regression model was trained to predict the correctness of individual stock counting activities directly after they are performed with the aim of improving the current alerting system. Although the predictive capabilities of the current version of the logistic regression model were not perceived to match the performance that is needed for successful implementation, this method provided insights into drivers of incorrect counting activities.

Keywords: Inventory Record Inaccuracy (IRI), inventory management, automated root cause analysis (RCA), incomplete orders, order fulfilment

Executive summary

Introduction

In today's business world, many retailers employ a Just-in-Time supply chain strategy in combination with automated inventory management systems that are heavily dependent on the accuracy of inventory records. However, in practice, the recorded and physically available inventory are seldom aligned, resulting in holding unnecessary inventory or not being able to fulfil customer demand. According to the academic literature, many organizations struggle to control these discrepancies and therefore suffer from the resulting effects. So does Picnic, which is an online grocery retailer operating in the Netherlands, Germany, and France. Even though Picnic already has several inventory record inaccuracy (IRI) prevention and correction procedures in place, the percentage of incomplete orders that are caused by IRI still varies between 25% and 35%. To be able to get more in control of this inventory management problem, the manual fulfilment process of Picnic was researched to provide a better understanding of the operational root causes of IRI that resulted in incomplete orders. Subsequently, two corrective solution approaches were explored that can potentially enable Picnic to reduce this harmful phenomenon.

Theoretical background

Referring to the scientific literature, the concept of IRI, its causes, drivers, impacts, and solution directions based on prevention, correction, and integration have already been widely discussed (e.g. DeHoratius & Raman, 2008; Rekik et al., 2019). However, most literature focuses on a general description of the causes while literature providing methods for root cause allocation or a distribution of the root causes is lacking. Furthermore, most corrective solution approaches consist of counting policies that focus on aggregate inventory data (e.g. Amaya Silva, 2022; Rossetti et al., 2001; Wijffels et al., 2016) instead of trying to detect individual situations that result in the creation of IRI. Therefore, it was found that there is a clear foundation of academic research on this concept and its potential solutions, while there are still several gaps to be explored.

Research questions

Combining the defined problem of Picnic with the gaps in the academic literature, the main research question to be answered in this research is defined as: "*How can inaccurate inventory records potentially resulting in incomplete customer orders in a manual fulfilment centre of Picnic be reduced?".* Inspired by the 10-step problem-solving framework of Okes (2019), three sub-research questions are defined that help to answer this research question in a structured way:

RQ1: "What operational errors can be considered as potential root causes of inventory record inaccuracy that may subsequently result in incomplete customer orders in the context of a manual fulfilment centre of Picnic?"

RQ2: "How can incomplete cases due to inventory record inaccuracy in a manual fulfilment centre of Picnic be allocated to their root causes?"

RQ3: "How can one assess Picnic's WMS data for suspicious situations that indicate that a root cause of inventory record inaccuracy has probably occurred?"

Methods

The first research question was solved by evaluating the academic literature, participating in the fulfilment process, and having discussions with relevant employees to come up with hypotheses on potential root causes of IRI. These hypotheses were thereafter compiled into a logic tree. To be able to accept or reject these potential root causes for allocating root causes to incomplete orders, the

possibility of developing a rule-based expert system that automatically performs this task by analysing historical inventory data was evaluated. Thereafter, two corrective solution approaches were explored. First, it was evaluated whether it is possible to convert the captured logic for historical root cause analysis into a logic that can create real-time alerts on the moment it is expected that the root causes occurred. Secondly, a logistic regression analysis was performed on data gathered by an alerting system that is currently in place for verifying suspicious stock-counting activities to explore whether this system can be enhanced by predicting the correctness of stock-counting activities.

Results

By creating the logic tree, various potential operational root causes for creating IRI that can subsequently result in incomplete orders in the context of a manual fulfilment centre of Picnic were identified. These root causes range from specific errors during the fulfilment subprocesses of receiving, replenishing/moving stock, counting, picking, and clearing/adjusting stock to errors that are not related to one single fulfilment subprocess such as freshness date management or registered stock on load carriers and roll cages that is physically not available.

To be able to confirm and therefore allocate these root causes to incomplete orders, it was found that this process could be automated for incomplete orders that belong to incomplete cases due to IRI having at least 5 incomplete consumer units (CUs) by developing a rule-based expert system. Next to proving this methodology, this solution made it possible to get insights into a root cause distribution for the analysed incomplete orders. Besides, these results supported most of the hypothesized potential root causes in the logic tree (including some new to the academic literature) and various root causes already discussed in the academic literature.

Exploring the two identified corrective solution approaches showed that it was possible to transform some of the root cause allocation rules into rules for creating real-time alerts in situations that are suspicious of the occurrence of a root cause. Based on the found root cause distribution and assuming that these alerts are solved in a timely manner, the maximum potential of this solution method was estimated to be able to reduce incomplete orders due to IRI (belonging to incomplete cases due to IRI having at least 5 CUs) by roughly 14%. For the logistic regression analysis solution approach, although several significant predictor variables were found and the model was able to correctly predict the correctness of most counting activities, the current version of the model showed predictive capabilities that were perceived to be too low for implementation. While the number of unnecessary alert checks is expected to be reduced by 60%, 37% of the incorrect counting activities (of which 47% can potentially cause incomplete orders) would be skipped if this model were implemented, which can be harmful to Picnic.

Conclusion

This study provides insights into the root causes of IRI that subsequently lead to incomplete orders and found that these root causes can be automatically allocated to incomplete orders (belonging to incomplete cases due to IRI with at least 5 CUs) by employing a rule-based expert system. The latter also offers insights into the distribution of these root causes. Besides, the solution approach of creating more specific or additional alerts by converting root cause allocation rules into real-time alerts can potentially enable Picnic to get more control over the accuracy of its inventory records and potentially reduce the number of incomplete orders due to IRI (belonging to incomplete cases due to IRI with at least 5 CUs) by roughly 14%. Finally, the findings on the application of the logistic regression approach show that it is not yet advised to implement the trained logistic regression model. However, these findings provided insights into drivers of incorrect counting activities and might be used for further exploration of this solution approach.

Table of contents

List of figures

List of tables

List of abbreviations

API	Application Programming Interface
CCABS	Cycle Counting policy with Adjusted Base Stock levels
CU	Consumer Unit
DC	Distribution Centre
EPV	Electronic Picnic Vehicle
FC	Fulfilment Centre
FCA	Automated Fulfilment Centre
FEFO	First Expired First Out
FN	False Negative
FP	False Positive
IABS	Inspection Adjusted Base Stock policy
IQR	Interquartile Range
IRI	Inventory Record Inaccuracy
JIT	Just-In-Time
lot	Combination of a location in an FC, the article (SKU) ID, and its freshness date
OOS	Out Of Stock
POM	Purchase Order Management System
QoS	Quality of Service
RCA	Root Cause Analysis
RFID	Radio Frequency Identification
Runner	Delivery Driver
SCF	Stock Confidence Factor
Shopper	Employee of an FC
SKU	Stock Keeping Unit
SOP	Standard Operating Procedure
SSCC	License plate with stock information
TD	Transaction Dependent
TI	Transaction Independent
TN	True Negative
TP	True Positive
TU	Trade Unit
VIF	Variance Inflation Factor
WMS	Warehouse Management System

1. Introduction

In today's world, the retail industry is under pressure. Managing e-commerce supply chains becomes more and more expensive, competition is fierce, suppliers are increasingly trying to pass on raw-material cost inflation to retailers, and labour costs are steadily rising while customer expectations keep increasing (Begley et al., 2020). Besides, the retail industry has recently experienced many supply chain disruptions, making it challenging to control supply chains. Starting with the Covid-19 pandemic in the early 2020s followed by a post-pandemic consumer demand surge and the Russian invasion of Ukraine that also led to sanctions of Western countries against Russia. It has been shown that, even in turbulent environments, if properly executed, the use of a Just-In-Time (JIT) supply chain can be very helpful to deal with the mentioned circumstances (Choi et al., 2023). Since a JIT supply chain makes it possible to save costs and increase customer service, many retailers try to implement this supply chain strategy. However, next to its benefits, employing a JIT supply chain also has its challenges. One of them is that a JIT supply chain usually results in holding relatively low inventory levels and relying on automated inventory management systems that are heavily dependent on the accuracy of inventory records (Kang & Gershwin, 2005). If there is a discrepancy between the recorded inventory and the inventory that is physically available, it might result in inefficient ordering decisions by the automated inventory management system, potentially leading to stockouts or holding unnecessary inventory. This problem is faced by many retailers that are therefore not always meeting the expectations of their customers or have unnecessary inventory holding costs which are both very harmful to an organization (Rekik et al., 2019). This thesis explores the concept of inaccurate inventory records resulting in the inability to meet customer demand in a JIT supply chain in the context of an online grocery retailer.

1.1. Company description

This research is conducted at Picnic Technologies, hereafter called Picnic. Picnic is an online supermarket founded in the Netherlands in 2015 and expanded to Germany and France in 2018 and 2021 respectively. It currently operates 5 distribution centres, 1 automated fulfilment centre, and 16 manual fulfilment centres and delivers to customers through more than 100 hubs. Picnic started in Amersfoort, the Netherlands, after which it quickly expanded to other cities. The proposition that made Picnic successful is that it delivers groceries to the customer's doorstep for free while having the lowest prices and fresh products. This proposition made online grocery shopping with delivery available for everyone, which was not the case in the past since competitors were very expensive and only served a niche luxury market. Due to its successful proposition, the organization grew very fast resulting in being the Netherlands' fastest-growing company in 2019 (Picnic, 2023).

Picnic is able to significantly reduce the overall food supply chain footprint while maintaining affordable prices and the highest quality for the customer by applying best-in-class technology, smart planning, and a fleet of electric vehicles. Moreover, all key software that is needed to operate its supply chain is built in-house allowing full control of the operations. This ranges from a purchase order management system (POM), a warehouse management system (WMS), route planning, forecasting algorithms, and many more. Besides, Picnic does not have any physical brick-and-mortar stores since it is an online-only retailer. Skipping this usually expensive step in the supply chain in combination with employing a JIT supply chain that aims at ordering fresh products only if they are ordered by the customers makes it possible to deliver high-quality products at an affordable price. In addition, in contrast to its competitors, it makes use of a milkman model for scheduling delivery routes. These competitors usually employ a taxi model, meaning that the route is dependent on the orders that have to be delivered on timestamps desired by the customer which can be very inefficient due to crisscrossing through an entire city. Instead, Picnic makes use of a fixed 'Milkman' route on fixed time

windows such that all groceries of an entire area are accumulated and can be delivered by one single electric vehicle. This makes it also possible to provide a closer time window of expected delivery than its competitors, which is more convenient for the customer. (Picnic, 2023).

Due to its successful business model that is able to change the way people are doing groceries and reduce the supply chain footprint at an affordable price, Picnic can be considered a very successful startup that is still growing and expanding very fast. This success is not unnoticed by important investors. In 2021, Picnic raised an investment of €600 million from a group of international investors led by the Bill & Melinda Gates Foundation. This is the investment fund of Microsoft founder Bill Gates and his ex-wife Melinda French (NU.nl, 2021).

1.2. Problem statement

Picnic makes use of fulfilment centres (FCs) for receiving stock from suppliers which can thereafter be picked to fulfil customer orders. All stock that is managed and moved within a fulfilment centre is registered in Picnic's Warehouse Management System (WMS). If Picnic wants to deliver what has been promised to the customer, it is crucial to always keep the correct level of stock on all locations, especially since Picnic makes use of a JIT supply chain.

However, it occurs that the level of stock registered in WMS (also called inventory records) does not correspond to the stock that is physically available on a specific location. This happens through errors in processes/activities related to the internal management of stock. Where internal management of stock refers to all stock-related processes/activities that are executed in an FC such as receiving, stock moves, stock counts, replenishments, picks, and many more. If the registered stock of a stock keeping unit (SKU) in WMS is not zero while the physical stock of this SKU is fully depleted, it can occur that a customer order cannot be completely fulfilled since the ordered SKU is not available for picking. This results in a substitution or cancellation of the ordered SKU and is defined as order incompleteness.

Since incomplete orders are not solely caused by inaccurate inventory records, a dashboard is currently used that evaluates the total number of incomplete (substituted or cancelled) items per SKU during a specific picking shift at an FC and distributes this quantity over different root causes of which inaccurate inventory records is one of them. Next to the root cause of inaccurate inventory records, incomplete quantities can be assigned to inbound incompleteness, late receiving, unavailable supplier, not being able to order at the supplier, waste, and unknown. Even though Picnic already has several inventory record inaccuracy (IRI) prevention procedures in place, the percentage of incomplete consumer units (CUs) caused by inaccurate inventory records still varies between 25% and 35% according to the dashboard described above. This shows that inaccurate inventory records have quite some impact on the completeness of customer orders and there is room for improvement to prevent this from happening.

The goal of this thesis is to provide a better understanding of the operational root causes of inaccurate inventory records resulting incomplete orders. After understanding these causes, the aim is to come up with solutions that enable Picnic to reduce this harmful phenomenon.

1.3. Project scope

It is important to prevent the project from becoming too complex. Therefore, the project boundaries are carefully defined. As already mentioned, this project focuses on inaccurate inventory records that can subsequently result in incomplete orders, where inaccurate inventory records are the result of errors in the internal management of stock in an FC. Therefore, much of this research consists of analysing incomplete orders due to IRI for their root causes. However, since stock in Picnic's FCs is not linked to a specific order, it is impossible to track the stock/individual items belonging to one specific

order through all inventory management operations in the FC starting from the inbound of stock from suppliers and ending with picking. Hence, it is important to mention that the level of detail during this part of the research will not be on the level of individual orders. Instead, the level of detail will be on the total number of incomplete items due to IRI per SKU during a specific picking shift in an FC. In the rest of this thesis, this will be referred to as an incomplete case due to IRI.

Since there is already a logic in place that identifies these incomplete cases due to IRI to come up with the dashboard as described in section 1.2, only the incomplete order quantities that are allocated to the root cause of inaccurate inventory records by the current root cause allocation logic are considered during this project. The decision is made to use the results of this existing logic instead of reinventing the wheel such that more time can be spent on the analysis of the problem itself.

Furthermore, only manual fulfilment centres (FC) are considered since the operations within the automated centre (FCA) differ quite a lot from the manual ones while the operations in the manual FCs are standardized. Moreover, since Picnic operates in three different markets, namely the Netherlands, Germany, and France, it is important to include this in the scope of the project. Although all FCs work according to the same standardized procedures, there might be some small deviations per country. For example, different suppliers can result in different procedures for receiving stock. To keep it simple, only the FCs that serve the Dutch market are considered during this project.

Finally, shoppers (which is Picnic jargon for employees of an FC) perform various actions to make sure that orders can be fulfilled. Since assessing all possible causes is very time-consuming it is decided that the quality of the designed solution is prioritized over the number of root causes that are covered. It is therefore possible that the solution does not cover all possible root causes.

1.4. High level overview of the delivery structure and fulfilment process

This section contains a high-level overview of Picnic's delivery structure and its fulfilment process to get familiar with the context of this research. Starting with its delivery structure, customers use the Picnic app to fill their baskets with groceries and create an order for a specific delivery window. After the deadline of this delivery window passed by, the orders for this window are picked in one of Picnic's FCs. In these FCs, the stock is received from one of Picnic's own distribution centres (DCs) and external suppliers after which this stock is processed by shoppers to fulfil customer orders (the fulfilment process). After the fulfilment process, which makes sure that the ordered SKUs of a customer are gathered in customer totes, the customer totes filled with the customer's groceries are shipped to local hubs with a lorry. In these local hubs, the customer totes are divided over Picnic's own electric vehicles (EPV) and delivered to the customer's doorstep in the defined time window by a runner (which is Picnic jargon for a delivery driver). This delivery structure is visualized in Figure 1 below.

Figure 1 - Picnic's delivery structure

As described in the scope, this thesis focuses on IRI that can potentially result in incomplete orders resulting from errors in activities that are under the control of an FC. To get a better understanding of the fulfilment process in which these activities are performed, it is worthwhile to provide a concise description of the standard operating procedure (SOP) of the fulfilment process.

FCs have two different types of inbound stock, which depends on whether stock information of the delivered load carriers is provided by the supplier. The SKUs that are delivered by Picnic's own DC and some other suppliers contain stock information on the barcode of a load carrier that is scanned when it is received at the docks. This barcode is called an SSCC license plate and contains information regarding which SKUs are available on the load carrier along with their quantities and freshness dates according to the supplier. This type of inbound is directly received in WMS during the unloading process and immediately available for further processing. Stock on load carriers is registered as trade units (TUs) which contain a specific number of consumer units (CUs) represented by the CU per TU quantitiy.

After the load carriers are received, a replenishment round can be started by a shopper, which is a guided flow that sends the shopper to the correct pick location in the replenishment aisle after scanning the barcode of a TU on the load carrier. The picking circuit, of which an example is pictured in Figure 2, consists of shelves with pick locations that have a picking aisle on one side and a replenishment aisle on the other side (respectively the right and left aisle in Figure 2). Picnic makes use of this structure to isolate both processes such that the chance of congestion in the picking circuit is minimized and pick locations are replenished according to a FEFO policy (First Expired First Out). When the shopper arrives at the pick location, it has to enter the number of TUs it wants to replenish on the pick location and subsequently, the TU is unpacked and replenished on the pick location. From this moment, the items are registered as CUs and are ready to be picked.

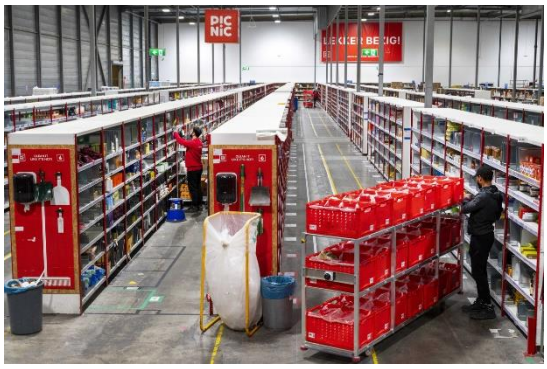

Figure 2 - Picking circuit in a Picnic fulfilment centre

In general, all TUs are replenished on the pick location if there is enough shelf space to do so. If not all TUs can be replenished due to a lack of space on the pick location, the leftover TUs are gathered on a roll cage such that they can be replenished on a buffer location for temporal storage. If the stock on a pick location is somewhat depleted again, this stock on a buffer location can be moved back to the pick location. This can be done by an urgent and non-urgent buffer move. Urgent buffer moves are performed when a pick location is (almost) fully depleted while there is still stock available at the buffer. This must happen as fast as possible since an empty pick location results in orders that cannot be fulfilled. Therefore, the stock is moved directly from the buffer to the pick location, which is considered an urgent buffer move. However, since performing individual stock moves from the buffer to a pick location can be a quite costly activity, the stock that can be moved to a pick location with less urgency is gathered on a roll cage such that multiple stock moves can be combined to save time, this is a non-urgent buffer move. Stock on a buffer and roll cage is still registered as a TU, which is transformed into a CU when the stock is successfully moved to the pick location.

Although the provided stock information by suppliers is usually correct, it sometimes occurs that the provided stock information mentions that there is more stock on the load carrier than expected by POM. In this case, only the expected stock is received on the load carrier in WMS and the excess stock is not yet registered. If the shopper tries to replenish stock that is not registered on the load carrier (the unexpected stock), it is sent to a quarantine cart. All excess stock on the quarantine cart is subsequently checked by an experienced team lead (captain) or receipt operator and manually over-received on either the pick location or a buffer location if it is indeed found that the FC received more than expected by POM. On the other hand, the stock information delivered by the supplier may state that there is stock on a load carrier while this is not the case. Therefore, if there is still stock on a load

carrier according to WMS while the load carrier is physically empty, shoppers can go to their captain (team lead) to call their load carrier empty. This automatically counts down all stock on the load carrier.

Once the stock is on the pick location, it is ready to be picked by the shoppers to fill the customer totes with the ordered products. After a picking round, the customer totes are gathered in a dispatching frame and dispatched to a local hub for delivery. The SOP of the fulfilment process for SKUs that are delivered by suppliers that provide stock information of the load carriers is visualized in Figure 3 below.

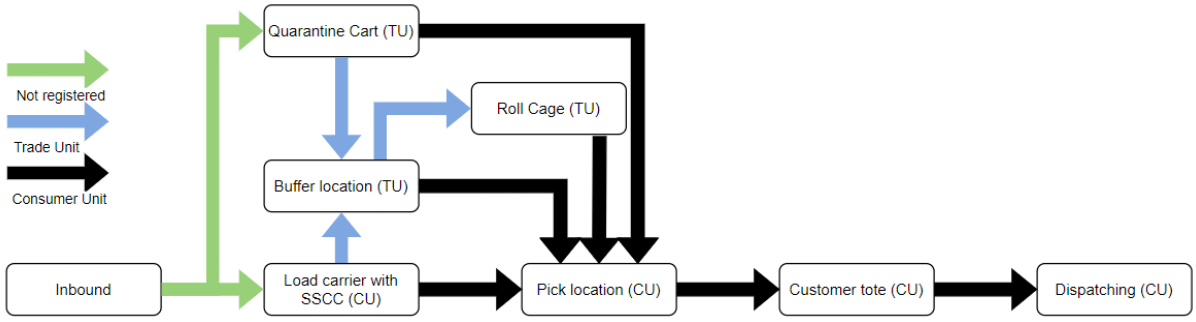

Figure 3 - SOP of fulfilment process for SKUs supplied with stock information service

In contrast to suppliers that provide stock information of the load carriers, which are usually Picnic's own DCs and suppliers of non-perishable SKUs, some suppliers of fresh groceries like fruit and vegetables do not support this service. For these SKUs, there are minor differences in the SOP of the fulfilment process. Instead of directly receiving stock at the docks by scanning the SSCC labels with stock information, this stock is not recorded in WMS and thus not available until it is manually received at either the pick or buffer location. Besides, it also occurs that one wants to receive more or another CU per TU size than expected by POM, if this happens, the stock is also sent to the quarantine cart after which an experienced captain or receipt operator can over-receive this stock. If the received stock is less than expected by POM, a receipt check is performed by checking the stock on all registered locations to verify whether the received quantity was indeed less than the ordered quantity. The rest of the SOP looks like that of the other SKUs. A visualization of this SOP is available in Figure 4.

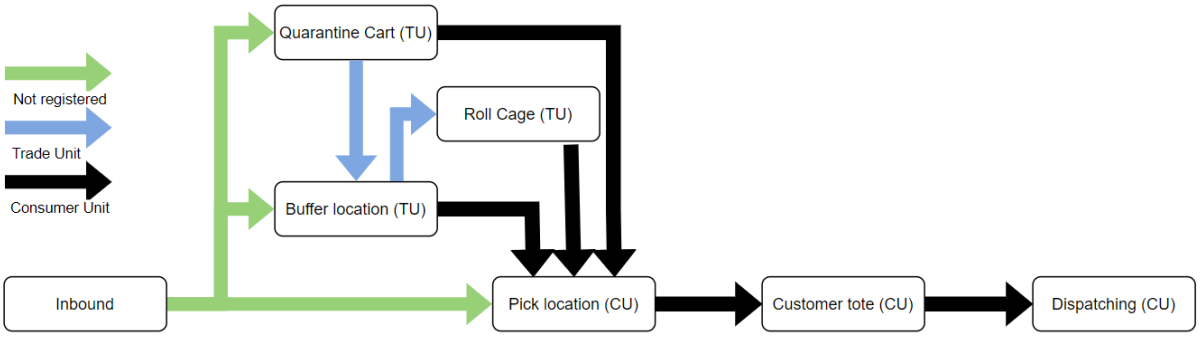

Figure 4 - SOP of fulfilment process for SKUs supplied without stock information service

It has to be noted that the SOPs as described above reflect the perfect execution of the fulfilment process. In reality, it is found that the process sometimes deviates somewhat from this SOP. For example, it also occurs that stock is replenished/moved via a regular stock move or put away action in WMS, which are quite similar to a regular replenishment but initiated via another WMS flow. Moreover, a location/flow that is not included to keep the SOP as simple as possible is consolidation. Although it is not frequently used, this flow makes it possible to move (a part of) new inbound stock directly to the dispatching area to top up already picked customer totes with products that were not available during the picking round to prevent orders from going incomplete.

Next to processing SKUs from inbound to picked customer totes that are ready for dispatching as described in the SOPs, some other interactions with stock are needed to successfully run the fulfilment process. Starting with stock counts, since discrepancies may occur between the physical and recorded stock in WMS, stock counts are performed to correct these discrepancies. There are four types of stock counts, namely a regular stock count, a location count, a load carrier empty action, and the creation of a freshness date. Next to the already explained load carrier empty action, a regular stock count is an unguided count, a location count is a count resulting from a guided counting round, and a create freshness date is a count from zero for a product for which there is currently no lot in WMS available yet. Here, a lot is referred to as the combination of an FC, the location in the FC, the article (SKU) identifier, and its freshness date, which is used by WMS to register the inventory data.

Besides, since Picnic offers a freshness guarantee to its customers for perishable items, the stock that is too old to be sold to the customer has to be removed from the fulfilment process such that it cannot be picked anymore. Removing these products is done via a guided WMS flow called stock clearing. Next to products that are too old to be sold, it is also possible that products must be thrown away because they are damaged, or their quality is too low to be sold. This is performed by a stock adjustment or quality check, which are respectively an unguided and guided WMS flow for registering these articles in WMS.

Finally, two actions are performed by shoppers/captains to deal with an empty pick location. If a shopper has to pick a product from a pick location that is fully depleted, it can call a shortage to let WMS and captains know that there is physically no stock available to be picked. Calling this shortage also allows the shopper to skip the pick and continue with the picking round. In some FCs, a count to zero threshold is activated (which is usually set to 5), this means that a location is counted down if a shortage is called by a shopper while the inventory on hand on this location according to WMS is equal to or less than the threshold. Shortages are also called by WMS itself if the stock according to WMS is equal to 0 while there are still picks that have to be performed on this location. These picks are then automatically skipped such that the shopper does not need to visit this pick location anymore. If shortages are called, the pick location should be checked and an urgent buffer move is created if any leftover stock is registered at a buffer location. If there is no leftover stock available in the FC, a captain can call a shortage solution request which results in substituting or cancelling the items that had and have to be picked from this pick location. Calling this shortage solution request thus results in incomplete orders.

To provide an overview of all relevant stock interactions during the fulfilment process, these interactions are listed in Table 1 below along with a concise explanation.

Table 1 - Stock interactions and their explanation

Stock interactions	Explanation
Receive	Inbound stock is registered in an FC for the first time
Solve issue	Excess unexpected inventory is manually over-received
Replenish stock	Stock is replenished by moving stock that is already available in an FC on a certain location to another location
Move stock	Stock is moved from one location to another location
Put away	Legacy flow to replenish stock on a load carrier to a pick location
Stock count	Stock is counted with unguided counting flow
Location count	Stock is counted with guided counting flow
Load carrier empty	Shopper mentions that a load carrier (LC) is empty while the LC still contains stock according to WMS, this stock is counted down
Create freshness date	A new lot is created for a specific freshness date of an article

Shortage	WMS or shopper reports that there is no stock on a pick location (either in WMS or physically)
Pick	Article(s) is(are) picked to fulfil an order
Stock clearing	Stock that does not meet the freshness guarantee anymore is cleared
Stock adjustment	Stock on a location is adjusted due to various reasons (damage, quality, etc.)
Quality check	Stock adjustment resulting from a guided quality check flow
Request shortage solution	Request a solution (substitution or cancellation) for order lines that cannot be fulfilled

1.5. The current IRI reduction initiatives

In terms of reducing IRI, currently, a guided counting policy is in place that uses a metric called the stock confidence factor (SCF). This metric aims at quantifying the confidence that the inventory record at a specific location is equal to the actual stock by making use of a weights effect model. This model subtracts weights when an event occurs that influences the stock. These events include picking, adjusting stock, reporting a shortage, counting mutations, and a count to zero. If this SCF drops below a certain threshold, the location is considered for a guided counting round. This policy aims to only count the locations of which one is not confident that the recorded inventory matches the inventory physically available at the location. Next to this guided counting procedure, there is also a possibility to perform unguided counts to correct any identified discrepancy. Besides, physical audits are performed to ensure financial conformance.

In addition, since correcting inventory records via stock counting activities can also create a discrepancy resulting from errors in this process, an alerting system is designed to provide alerts for SKUs that experienced an absolute counting delta larger than a specific threshold. The counted locations are included in a list that must be recounted to verify that there indeed was a delta. This list is currently ordered based on the highest absolute delta, meaning that SKUs with the highest absolute delta are being prioritized during the verification process. The alerting tool, as explained above, seems to be a very effective first step in practice. However, alerts are also created for counts that are actually correct, potentially resulting in multiple unnecessary counting activities. Besides, the alerting system currently only includes mutations from inventory counting, meaning that other sources of IRI are not yet covered by the current implementation. Therefore, this tool can be improved by creating a model that can predict whether a stock mutation due to counting is correct or not such that this can be used to prioritize or filter alerts in combination with/instead of the absolute delta. Besides, additional types of stock mutations can be added such that other activities resulting in IRI can also be verified to ensure that the inventory records correctly represent the physically available inventory.

1.6. Thesis outline

This thesis is structured as follows. After this introduction, in section 2, the academic literature on JIT supply chains, the concept of IRI along with its causes, drivers, impact and solutions are discussed after which gaps in the literature are identified to create an academic background for this thesis. Thereafter, the research questions are defined followed by the methods that are employed to answer these research questions in sections 3 and 0 respectively. The results of the applied methods are considered in section 5 which are used to draw conclusions in section 6. Finally, the implications, limitations and suggestions for future research are discussed in section 0.

2. Theoretical background

This section contains a literature review to create an academic background for this research and the relevant concepts. First, a concise overview is provided of a JIT supply chain, its principles, and the importance of accurate information when using JIT. Thereafter, the concept of inventory record inaccuracy (IRI) is explained along with its magnitude in practice, its behaviour over time, the causes of IRI, the drivers that increase the occurrence of IRI, and the monetary impact of IRI on an organization. Subsequently, three potential solution types for reducing IRI, which are preventing, correcting, and integrating are discussed. Finally, the gaps in the literature are described to get insight into the relevance of this research.

2.1. JIT supply chain

2.1.1. Introduction to a JIT supply chain

A JIT supply chain is defined as having the right items at each supply chain partner when needed in the right quantity, at the right place, and with the right quality (Yang et al., 2021). The principle of JIT is introduced to the business world in the 1970s by Toyota Motor Corporation's JIT production system. The goal of this production system was to reduce waste by getting the right quantity of (good quality) parts to Toyota's assembly line at the exact moment it was needed for production (Steele, 2001). Hereby, the precise demand of customers could be met with a minimum delay while costs can be minimized by restricting expenditure in any form (Sohal et al., 1989). This concept of JIT was proven to be effective and widely adopted in many different contexts (Fawcett & Birou, 1993). The reason for this is that a JIT supply chain enables organizations to cope with the accomplishment of increasing pressures and maintain their competitive position and performance for surviving in today's dynamic and rapidly evolving markets (Singh & Singh Ahuja, 2013).

2.1.2. Key principles of JIT

When looking at the advantages that can be attained by implementing the JIT philosophy into a supply chain, it is important to mention that it can be a challenge to make it work. According to Hussein & Zayed (2021), six fundamental principles can help in getting a successful JIT supply chain. The first factor is a pull system, meaning that materials or products are only sent after receiving orders from the demand side, resulting in the fact that the need for high inventory levels can be eliminated. Secondly, waste elimination is a JIT principle that focuses on eliminating activities that do not add value to the final product. The pull system is a great example of waste elimination since holding inventory can be considered a waste activity. Since JIT targets holding zero inventory, one has to maintain an interrupted workflow through different activities in the supply chain, this results in the third principle of having a smooth workflow. Besides, the principle of total quality management, perceived as receiving the required materials/products on time, with the required quantity and quality, is needed if one wants to implement a smooth workflow. This can be attained by having reliable, long-term customer-supplier relationships. Finally, JIT is only possible if the entire company shows commitment to the JIT philosophy, both top management and employees in the process (Hussein & Zayed, 2021).

2.1.3. Importance of information accuracy in JIT

Because of the reduced inventory levels associated with JIT supply chains, there exists an urgent need for accurate inventory records (Sheppard & Brown, 1993). To provide high availability of products at a minimal cost in industries where fierce competition exists and profit margins are thin, many companies automated their inventory management processes, for example, by using automated replenishment systems. These systems track the number of products that are available in stock and automatically place an order with the supplier if needed (Kang & Gershwin, 2005), or recommend order quantities to a store manager for every order cycle (van Donselaar et al., 2010). On average 1% of annual sales is

spent on acquiring and operating these systems when looking at companies in the retail sector (Rekik et al., 2019). However, automating inventory management processes results in an inventory system that depends on the accuracy of the computerized information system (Kang & Gershwin, 2005) since these automated replenishment systems assume that inventory records are accurate (Corsten & Gruen, 2003). This means that, if one wants to effectively coordinate the movement of goods using these automated systems, it is critical that information in the inventory management system regarding what product is in which location and in what quantity is accurate (Kang & Gershwin, 2005). If this is not the case, this inaccuracy can result in lost sales or unnecessary high inventory levels (Cannella et al., 2015; Chuang & Oliva, 2015; DeHoratius & Raman, 2008; Kang & Gershwin, 2005; Rekik et al., 2019; Shabani et al., 2021), and item obsolescence (Wijffels et al., 2016). Corsten & Gruen (2003) mentioned inaccurate inventory records to be a major obstacle to on-shelf availability.

2.2. Inventory record inaccuracy

2.2.1. What is inventory record inaccuracy?

Generally, an inventory record consists of at least an SKU identifier, location, and quantity on hand. An inventory record is inaccurate if an error exists in any of these fields (Rossetti et al., 2001). According to Chuang & Oliva (2015), inventory record inaccuracy (IRI) is defined as the discrepancy between physical and recorded inventory levels. Some retailers use a discrepancy zone, meaning that no error is assumed to exist if the discrepancy is within a specific range (Rekik et al., 2019). However, in this thesis, IRI is perceived as any discrepancy between physical and recorded inventory levels. The discrepancy can be positive and negative resulting in overstock and understock respectively. An overstock situation occurs when the recorded stock level is less than the actual stock level while an understock situation occurs when the recorded stock level exceeds the actual stock level (Ernst et al., 1993). According to Shabani et al. (2021), it is important to make this distinction since both versions of discrepancy have different consequences. It is worth noting that the situation of positive and negative discrepancies can occur interchangeably over time for the same SKU (Rekik et al., 2019).

Starting with a positive discrepancy, if the available stock of an SKU is higher than the stock recorded in the inventory management system, inefficient ordering decisions are made, such as ordering too early (Rekik et al., 2019) or ordering more than required (Kök & Shang, 2007). These inefficient ordering decisions result in carrying unnecessary inventory leading to higher procurement and holding costs (Rinehart, 1960; Shabani et al., 2021). Moreover, especially for perishable products, excessive inventories can result in waste if the sell-by-date has passed. This is detrimental to retailers, as waste is a major cost factor for perishable products (van Donselaar et al., 2006). The effect of ordering too early due to a positive discrepancy is visualized in Figure 5.

On the other hand, a negative discrepancy can delay a necessary replenishment or ordering a lower quantity than needed resulting in disappointing customers by promising a product that cannot be delivered due to an out-of-stock (OOS) situation. These effects lead to a decreased service level followed by eroded customer goodwill (Barratt et al., 2018; Ernst et al., 1993; Rekik et al., 2019; Shabani et al., 2021). When talking about out-of-stock situations, one can differentiate between store-out-of-stock and shelf-out-of-stock. Where store-out-of-stock means that the SKU is not available at the premises at all, while shelf-out-of-stock considers an available SKU that is at the premises but cannot be picked (Rekik et al., 2019). The effect of a delayed necessary replenishment resulting in an OOS is visualized in Figure 6. A concept caused by negative discrepancy that is worth mentioning is inventory freezing. Inventory gets frozen if the stock record of an SKU is above the reorder point while the SKU is physically out of stock. Since no replenishments will be triggered in this situation (because the inventory record is above the reorder point) and no stock is subtracted from the stock records due

to sales (not possible since OOS), this SKU is not available until this frozen inventory is discovered (DeHoratius et al., 2008; Rekik et al., 2019).

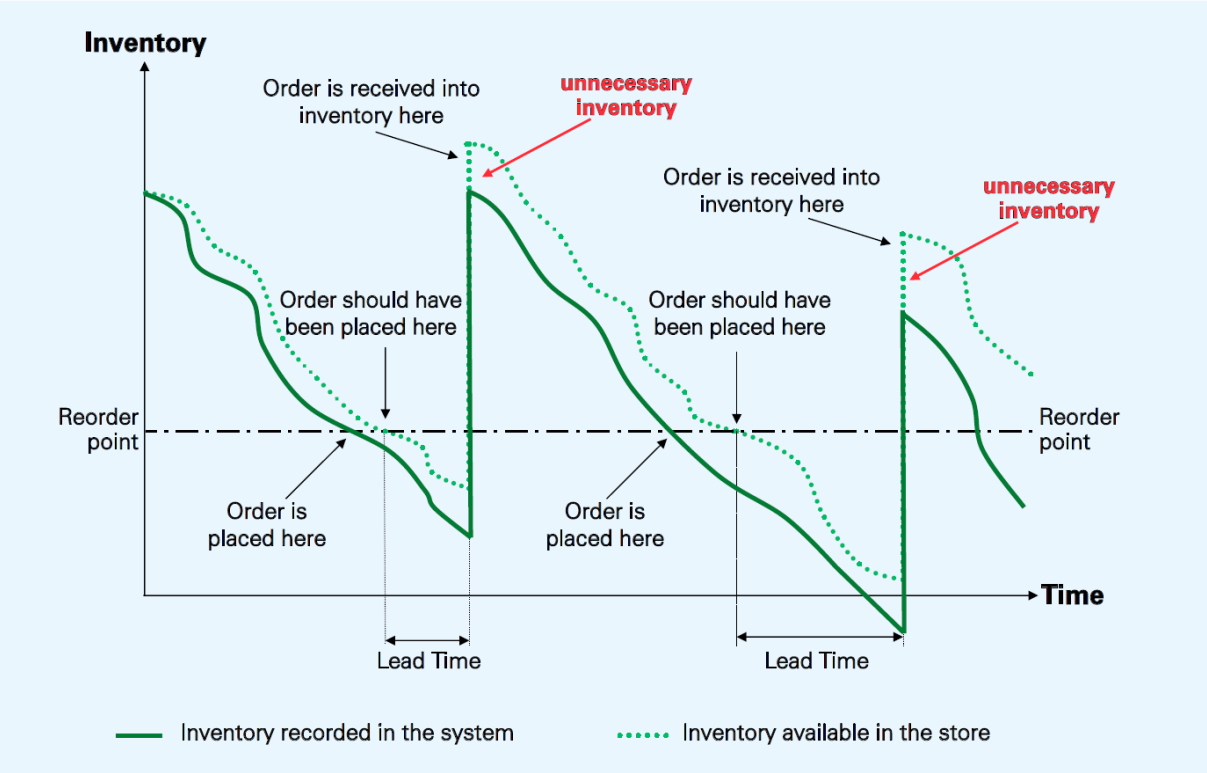

Figure 5 - Ordering too early due to positive discrepancy (Rekik et al., 2019)

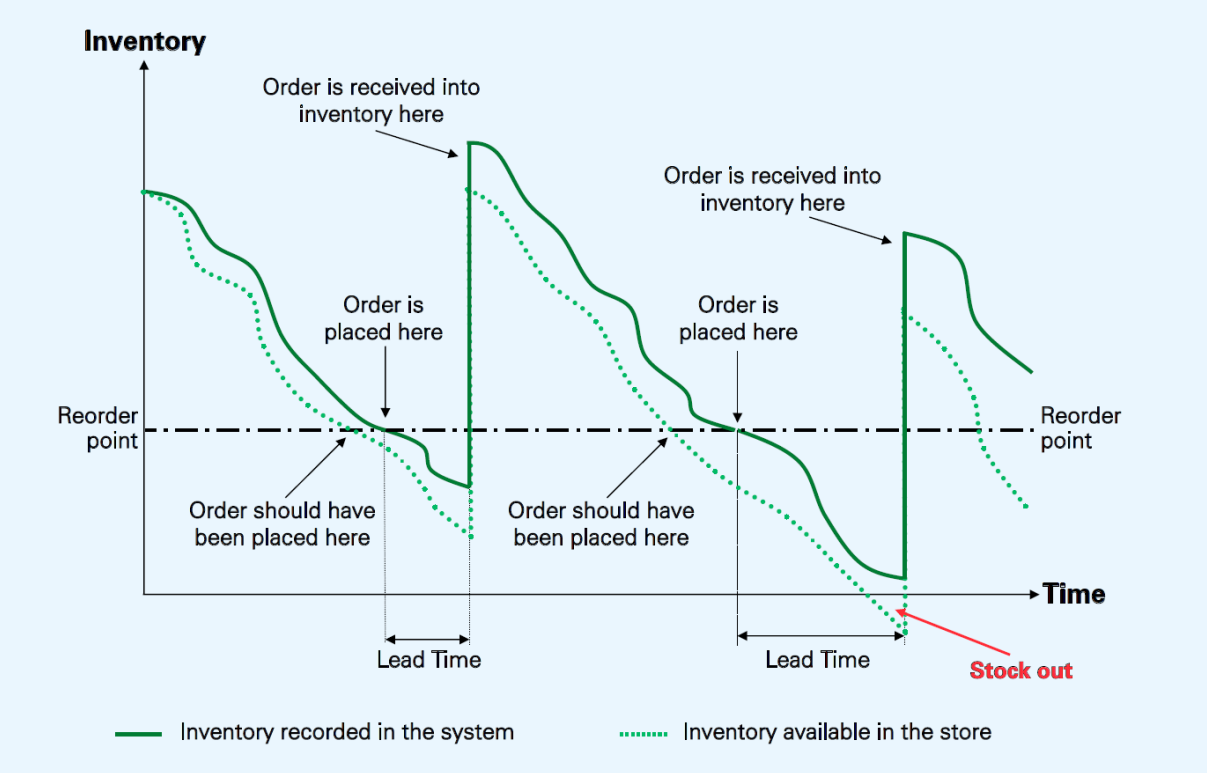

Figure 6 - Ordering too late due to negative discrepancy (Rekik et al., 2019)

2.2.2. Magnitude of IRI

Contrary to popular belief, inventory records and physical inventory are seldom aligned (Kök & Shang, 2007). A lot of research has been performed to get insight into the extent of IRI in various organizations and industries. Raman et al. (2001) found that 65% of the 369,592 inventory records (approximately 10,000 SKUs of 37 retail stores) at a leading retailer did not match the physically available inventory. The absolute discrepancy between the system and physical stock was on average 35% of the target inventory level (Raman et al., 2001). Rekik et al. (2019) researched the occurrence of IRI across multiple retailers that could be classified into either grocery/general or fashion/apparel retailers. They found that the physical stock of 59.54% of the 233,318 audited SKUs did not match the reported quantity in the information system, which is in line with the results of Raman et al. (2001). When comparing the different retailer types, it was found that the grocery/general retailers had a higher share of inaccurate stock records (63.39%) than the fashion/apparel retailers (54.08%). Rekik et al. (2019) attribute this difference to the fact that shrinkages, like decay, spoilage, and damage occur more frequently (more than twice) in groceries than in fashion. Moreover, the 59.54% of inaccurate records can be split into 32.07% records with negative and 27.46% with positive discrepancy. The average positive and negative discrepancies were found to be +6.6 and -6.0 units respectively (Rekik et al., 2019). Kang & Gershwin (2005) found, on average, 51% of the inventory records of a global retailer to be accurate, ranging from 30% to 80% per retail store. However, if using a discrepancy zone of ±5 units, the average inventory record accuracy rose to 76%, ranging from 60% to 90% per retail store (Kang & Gershwin, 2005).

2.2.3. IRI behaviour over time

It is also useful to know how IRI behaves over time. Rekik et al. (2019) found that positive discrepancies remain relatively stable and negative discrepancies tend to accumulate. A reason for this could be that negative discrepancies are often caused by shrinkage resulting in a permanent loss while positive discrepancies caused by misplaced items are counted down during a gap scan and counted up if the item is found again. These temporal losses due to misplacement can balance each other out over time (Rekik et al., 2019). The fact that discrepancy between physical and recorded inventory accumulates over time is also mentioned by DeHoratius & Raman (2008).

2.2.4. Causes of IRI

As items physically move through the supply chain and inventory records are automatically updated in the information systems, numerous opportunities exist for the recorded stock quantity to diverge from the actual stock quantity (DeHoratius & Raman, 2008). According to Kull et al. (2013), this discrepancy between recorded and available stock can result from two different types of errors in the supply chain. Starting with errors that are triggered by transactions, such as replenishments, demand orders, or product returns. These errors are called transaction-dependent (TD) errors and can take the form of incorrect picking, incorrect deliveries, or misplaced items (Kull et al., 2013). On the other hand, IRI can be caused by transaction-independent (TI) errors. This type consists of errors that occur irrespective of transactions in the supply chain and are influenced by the amount of inventory on hand. Examples of this type of error are errors related to the internal movement of materials, and shrinkage from theft, damage, or spoilage (Kull et al., 2013).

DeHoratius & Raman (2008) mention selling and restocking errors, replenishment errors, counting errors, database errors and problems with data synchronization, and customer/employee theft as commonly cited causes of IRI. Selling and restocking errors can occur due to mis-scanning items during the check-out process or not processing a restock correctly. For example, if multiple distinct products share the same price or look familiar, a salesperson might scan one product multiple times and leave the other products unscanned. Instead of depleting the inventory records of every item with one unit, the record of the scanned item is depleted multiple times while the records of the other products

remain unchanged. This results in a mismatch between recorded and available inventory (DeHoratius & Raman, 2008; Kang & Gershwin, 2005).

A typical example of IRI being caused by replenishment errors is that most companies do not scan every item that is delivered by a distribution centre. Instead, it is checked whether the delivered quantity of pallets or boxes is equal to the expected quantity on receipt. If this is the case, the inventory records are automatically updated with the ordered quantity. However, this is based on the assumption that the order has been filled correctly. If this assumption is not met because an order picker picked the wrong SKU and/or quantity, the received quantity does not match the ordered quantity leading to inaccurate inventory records (DeHoratius & Raman, 2008; Kang & Gershwin, 2005).

Moreover, IRI can be caused by poor inventory counts. In practice, large errors often remain in the inventory records due to inaccuracies in the counting procedure (Iglehart & Morey, 1972; Kang & Gershwin, 2005). In fact, Rinehart (1960) found that 80% of the researched discrepancies were caused by trying to correct the discrepancy by inventory counting itself, this effect of inventory counting resulting in the inducement of IRI is also mentioned by Kull et al. (2013). For example, an employee may count single items when he/she should be counting multi-packs (DeHoratius & Raman, 2008). Besides, discrepancies can be created if personnel estimate the count instead of truly counting the inventory (Woolsey, 1977). It also happens that transactions occur during the count, which makes it very difficult to ensure fully accurate counts (Iglehart & Morey, 1972). Of course, next to the mentioned examples, it is also possible for the count to be inaccurate due to human counting errors during the counting process (DeHoratius & Ton, 2015; Kull et al., 2013).

Database errors and poor data synchronization can result in IRI having a master database that states the wrong case-pack sizes or time lags between the flow of material in information (DeHoratius & Raman, 2008; Iglehart & Morey, 1972). If the case-pack size that is active in the database differs from the case-pack size received, an incorrect quantity is added to the records at the moment of receiving a replenishment. Besides, if there is a difference between the flow of material and information, it is possible that already processed receipts are not yet updated in the records, resulting in an inaccurately recorded quantity (DeHoratius & Raman, 2008).

Employee and customer theft is also causing IRI due to a decrease in stock that is not registered. Theft can happen across the entire supply chain, including the DC, during transport, and the store (DeHoratius & Raman, 2008). Employee and customer theft belongs to the concept of shrinkage. Shrinkage considers all errors that cause loss of products ready for sale (Wijffels et al., 2016) and is, next to theft, caused by out-of-date, damaged, and spoiled items (Kang & Gershwin, 2005). Two forms of shrinkage exist, namely known and unknown shrinkage. Known shrinkage refers to all losses of stock that are identified by employees and reflected in the inventory record. In contrast, unknown shrinkage is lost stock that was not registered resulting in not updating the inventory records. Employee and customer theft belongs to the latter form of shrinkage together with damaged/spoiled/out-of-date items that are discarded without any proper registration. Therefore, unknown shrinkage is one of the causes of inaccurate records (Kang & Gershwin, 2005).

2.2.5. Drivers of IRI

Next to mentioning the potential causes for IRI, research is also performed on the existence of drivers that increase the occurrence of these causes and affect IRI in general. Rekik et al. (2019) mention the annual selling quantity, size of product portfolio, importance of shrinkage, and product category as drivers or factors that influence IRI.

Starting with the annual selling quantity, which Is also mentioned by DeHoratius & Raman (2008) as a driver of IRI. Rekik et al. (2019) found that 19.30% of the items in grocery/general merchandise retailers that contributed to 70.3% of the sales, which are classified as fast movers, were responsible for 49.23% of the inventory discrepancies (in value) (Rekik et al., 2019). The results of fashion/apparel retailers showed similar findings with slight differences in proportions, however, the finding of fast movers being responsible for most of the inventory discrepancies was also found here. In contrast, it has to be noted that Sheppard & Brown (1993) found the number of transactions to be not significant. This was not expected since probability theory would suggest that more transactions result in a bigger chance of error. Possible explanations for this counterintuitive result are the research design, the design of the firm's computer application programs, and/or high intercorrelations with other variables resulting in a loss of univariate power in a multivariate model (Sheppard & Brown, 1993).

Secondly, the size of the product portfolio, also referred to as product variety, is found to be a driver of IRI (DeHoratius & Raman, 2008; DeHoratius & Ton, 2015; Rekik et al., 2019). Product variety increases the complexity and confusion in the operating environment causing employee errors (DeHoratius & Ton, 2015). For example, difficulties in differentiating products increase since the chance of having more nearly identical products becomes higher if more products are added. This can result in errors during the fulfilment, audit, or checkout processes (DeHoratius & Raman, 2008). Increasing product variety can also result in IRI since adding more products in the same area increases the number of steps needed to replenish the store. Stores have limited shelf space which causes more movement of products to other storage areas if more products are added. Since every step in a replenishment process is prone to errors, the probability of IRI increases if the product variety is increased (DeHoratius & Ton, 2015).

The importance of shrinkage is also a relevant driver of IRI and differs among sectors. For example, spoilage, decay, and product damage occur more frequently in the grocery/general merchandise compared to the fashion/apparel sector (Rekik et al., 2019). Furthermore, some product categories are more susceptible to IRI than others, but this seems to be very retailer-specific (Rekik et al., 2019).

In addition to the aforementioned drivers, the unit value and total value of an SKU are found to have a significant effect on IRI (DeHoratius & Raman, 2008; Sheppard & Brown, 1993). SKUs that have a high value are susceptible to theft and other types of shrinkage have a direct effect on operating profits. To be able to prevent shrinkage, these SKUs are usually closely monitored by auditing at multiple stages of the supply chain, using secured packaging, and storing them in locked cages. This results in a lower level of IRI due to a less error-prone environment (DeHoratius & Raman, 2008; Sheppard & Brown, 1993). The total dollar value of an SKU refers to the monetary value of the stock of an SKU, the impact of this factor is two-sided. It can either result in a lower IRI since the stock location with a high dollar value is better monitored because it contains valuable items (DeHoratius & Raman, 2008). On the other hand, a high dollar value of the stock can be the result of a large quantity of stock on hand, resulting in an increase in IRI (Sheppard & Brown, 1993).

The effect of high inventory levels resulting in higher IRI levels was also found by DeHoratius & Ton (2015) and Sheppard & Brown (1993). High inventory levels result in fewer learning opportunities since fewer stockouts occur if the inventory levels are high. A stock out is an indicator that something is wrong, presenting a learning opportunity where the stock records can be adjusted to zero if the stock record is not accurate. If fewer of these situations occur, the stock records are corrected less resulting in a higher level of IRI. Next to the reduction of learning opportunities, high inventory also increases the complexity of the operating environment causing execution problems (DeHoratius & Ton, 2015). For example, counting errors are expected to increase because the counting process is more time-consuming and cumbersome to execute (Sheppard & Brown, 1993).

Moreover, it is found that audit frequency has a significant impact on IRI (DeHoratius & Raman, 2008). As already mentioned in section 2.2.3, the discrepancy between physical and recorded inventory increases over time, especially when looking at negative discrepancies (Rekik et al., 2019). These errors accumulate as long as the records are not updated by a physical inventory audit (DeHoratius & Raman, 2008).

However, an attempt to align physical and recorded inventory can also result in IRI itself. Rekik et al. (2019) investigated the development of IRI based on manual adjustments. It was found that there is a clear relationship between the size of the manual adjustments and the inventory discrepancy. If one manually removed stock from the inventory records, results showed that this led to positive discrepancies over time. On the other hand, manually adding stock led to negative discrepancies over time. In addition, the magnitude of the manual interventions resulted in a higher distortion of the inventory record accuracy (Rekik et al., 2019).

Another factor found by DeHoratius & Raman (2008) to have a significant impact on IRI is the used distribution structure. Retailers make use of two different methods to distribute inventory to their stores. They either receive inventory into a DC that is owned and operated by the retailer itself, which subsequently distributes the inventory to the stores, or they have vendors that directly ship to the stores. The differences between both methods (how the inventory is handled due to differences in governance) can result in different levels of IRI. One can think of both pros and cons of both distribution methods. On the one hand, one might expect lower IRI among items shipped through a retail-owned DC due to greater willingness to cooperate with other members of the same group, established patterns of communication, and effective and efficient coordination thanks to vertical integration (DeHoratius & Raman, 2008). However, retailer-owned DCs are also susceptible to internal procurement bias, persistence, communication distortion, and internal expansion bias. It is mentioned that upstream parties can refuse to take costly actions to improve quality control when the downstream party is not paying a bonus. Since retailer-owned DCs cannot be rewarded (or penalized) for accurate (or inaccurate) fulfilment of orders whereas vendors are motivated to increase quality performance since the retailer also has alternative sourcing options. Moreover, retailer-owned DCs usually don't have an outside option to sell their goods resulting in only expending minimum effort to ensure quality (DeHoratius & Raman, 2008). In the end, DeHoratius & Raman (2008) found higher IRI levels for items received from retailer-owned DCs compared to items fulfilled by vendors.

Besides, every interaction between an employee and stock and/or information system is susceptible to result in IRI. Therefore, it is important to also look at drivers related to employees. DeHoratius & Ton (2015) state that employee turnover and training, employee workload, and employee effort have an impact on the accuracy of inventory records. The effect of labour effects, distinguishing full-time and part-time employees was investigated by Chuang & Oliva (2015). Starting with employee turnover and training. In general, retailers face high levels of employee turnover. Studies reported an employee turnover in the retail industry of 124% and 112% for part-time, and 74% and 65% for full-time employees while it was only about 10-15% for US businesses in general (DeHoratius & Ton, 2015). High levels of employee turnover have an impact on the execution of a process in numerous ways. First, existing operations are disrupted by employee turnover. If an employee decides to quit, this job often must be fulfilled by a new employee who has to be found and trained. During this period of acquiring and training a new employee, the existing employees generally experience a higher workload. which may result in more errors. Moreover, existing employees often tend to be demoralized by departing employees which may also cause more errors. Second, employee turnover results in a loss of accumulated experience. If one performs a job for a longer period, the performance usually increases, and fewer errors are made. Third, since employees in the retail industry typically leave within a year,

it is often decided to not invest a lot of money and time in training. This results in employees who start performing their jobs with no full understanding of the processes and their impact on the operations. Due to these reasons, high employee turnover and minimal training negatively affect the accuracy of inventory records (DeHoratius & Ton, 2015). As already mentioned, a high employee workload is likely to result in IRI. This is the case since employees are more likely to not conform to the designed processes if their workload is increased and more errors tend to be made (DeHoratius & Ton, 2015). Therefore, a high workload results in lower accuracy of inventory records. Besides, employee effort is mentioned to affect IRI since more effort in monitoring inventory probably results in more accurate inventory records. However, this concept of employee effort is unobservable (DeHoratius & Ton, 2015). Finally, Chuang & Oliva (2015) researched labour effects and found strong evidence that full-time labour reduces IRI while part-time labour is not able to alleviate it since full-time employees tend to improve operational execution in a better way than part-time employees.

Not yet mentioned drivers of IRI stated by Sheppard & Brown (1993) are quantity per transaction, weight counting, and the number of places used. A high quantity per transaction results in a high physical count of items to fulfil the transaction, the difficulty of this physical count is likely to increase as the quantity per transaction increases. It is especially easy to lose count of large volumes of small items but larger items in large lot sizes can also result in counting difficulties due to their bulk and packaging (Sheppard & Brown, 1993). However, it has to be mentioned that Sheppard & Brown (1993) counterintuitively did not find a significant effect of quantity per transaction on predicting record errors. Like their insignificant findings on the number of transactions, this can be due to the fact that the researched company used a computer application program that showed messages to the employee if possible discrepancies such as issuing more items than called for or issuing items that were not on the bill of material occurred. Also, the fact of multiple errors cancelling each other out is discussed as one of the reasons for this finding. Finally, it is possible that there was an unexplainable artefact in the researched environment or high intercorrelations with other variables that resulted in a loss of univariate power in the multivariate model that was used (Sheppard & Brown, 1993).

Moreover, weigh-counted parts are prone to inventory record errors since it is not certain that every item of the same SKU weighs exactly the same (Sheppard & Brown, 1993). Lastly, if the number of locations an item is used increases, the number of transactions and complexity of record keeping probably increases. As a result, it is expected that more errors occur if items use multiple locations (Raman et al., 2001; Sheppard & Brown, 1993). Both weigh-counted parts and the number of locations used were found to have a significant effect on inaccurate inventory records (Sheppard & Brown, 1993). In addition, Raman et al. (2001) mention that store design, especially the design, number, and size of the backrooms or other storage areas can have an impact on IRI.

As a final point, when looking specifically at distribution centres (DCs), Barratt et al. (2018) found different levels of IRI at different types of DCs. Recent developments of Internet-enabled channels to the customer resulted in the fact that the role of some DCs changed from one that only ships to brick-and-mortar stores to one that also fulfils customer demand by shipping directly to the customer. This increases the operational complexity for managers and employees resulting in more errors that can cause IRI. In fact, it was found that DCs that also fulfil online orders directly to the customer, facing higher complexity, had IRI ranging between 48% and 70% of its records, which was only between 22% and 56% for DCs that only served brick and mortar stores (Barratt et al., 2018).

2.2.6. IRI impact

IRI is one of the greatest obstacles to successful inventory management (Corsten & Gruen, 2003; Kang & Gershwin, 2005). As already mentioned, inaccurate records result in inefficient inventory management leading to holding unnecessary inventory and/or having a lower service level due to

stockouts. These consequences result in a monetary impact on a company. Raman et al. (2001) performed an in-depth study of two retailers and found that both retailers experienced a profit reduction of more than 10% as a result of inaccurate inventory records and misplaced SKUs. DeHoratius & Raman (2008) found a revenue loss of more than 1% and a reduction of 3% in gross profit due to IRI in a North American retail chain. Rekik et al. (2019) reported an average increase in sales of 5.98%, ranging from 3.83% and 8.38% across different stores after a stock take was performed to reduce IRI. Especially fast movers seemed to benefit from an increased inventory record accuracy with an average increase in sales of 7.63%, followed by middle and slow movers with an average increase in sales of 5.10% and 2.19% respectively. Finally, the increase in sales was higher for SKUs that suffered higher discrepancies. SKUs suffering from high, medium, and low discrepancy experienced an average increase in sales of 14.51%, 7.01%, and 2.11% respectively (Rekik et al., 2019).

2.3. Solutions for reducing IRI

Looking at the monetary impacts of inaccurate inventory records mentioned in section 2.2.6, it is clear that having accurate inventory records can be beneficial, if not necessary, for companies to enhance performance and stay ahead of the competition. Over the years, academic research and practical experience resulted in the development of various methods that aim to solve, or at least reduce IRI and/or its consequences. These methods range from different procedures for inventory audits to inventory policies with logics that automatically correct for discrepancies during order moments and many more. DeHoratius et al. (2008) mention prevention, correction, and integration as possible ways to respond to IRI. Here, prevention refers to reducing or eliminating the root causes of IRI by implementing and executing process improvements, correction consists of identifying and correcting discrepancies through auditing policies, and integration means using inventory planning and decision tools that are robust by accounting for IRI. Next to correcting IRI by making use of counting and inspection methods, and preventing IRI by benchmarking performance across all facilities of the company such that drivers of poor execution are understood and poorer-performing facilities can learn from their better-performing counterparts, Raman et al. (2001) mention creating awareness of the problem and its impact as an additional way for preventing IRI.

2.3.1. Preventing IRI

2.3.1.1. Create awareness of the problem and its impact

It frequently occurs that employees, even in functions like replenishment and merchandising that heavily rely on data, are unaware of the extent and magnitude of errors in the execution of operational processes. A lack of awareness is often one of the causes for inaccurately scanning merchandise since employees do not realize what the impact of their actions is on the accuracy of inventory records. Most types of training in retail operations focus on increasing productivity. This results in an environment where point-of-sale scanners are viewed as labour-saving devices whereas they are also critical data-gathering tools. To create this awareness, the importance of the data collected should be taught to the employees along with the effects that inventory record inaccuracies have throughout the organization (Raman et al., 2001). Next to the employees that perform the operational processes, senior management also needs to be aware of the impacts of inaccurate inventory records since this is a crucial step in gaining senior management commitment for enhancing the accuracy of inventory records (Raman et al., 2001).

2.3.1.2. Benchmarking within the retail chain

The execution of processes affecting IRI can be improved by learning from other parts of the retail chain. In a retail chain, it often occurs that some stores consistently execute their processes much better than other stores. These better-performing stores show evidence that it is possible to manage and improve operational process execution to reduce IRI. This evidence can be used to get insight into

the drivers of poor execution, in addition, it can motivate stores that are lagging behind. Since most stores within the same retail chain have the same monetary incentives and use the same information technology, it can be assured that the differences in performance across stores are not driven by technology and incentives (Raman et al., 2001).

2.3.2. Correcting IRI through inventory audits

Inventory audits can be used to update the recorded inventory aiming at removing the discrepancy between recorded and physical inventory. There are different alternatives for performing an inventory audit. The commonly used alternatives are physical audits/full inventory counts and cycle counting (DeHoratius & Raman, 2008; Raman et al., 2001; Wijffels et al., 2016).

2.3.2.1. Physical audit

A physical audit is usually performed on a periodic (typically annual) basis to ensure financial conformance (Raman et al., 2001). During a physical audit, every item in a store is counted while the store is closed to customers. The counted inventory is used to assess whether the aggregate dollar value of the inventory matches the expected value and the inventory records are updated afterwards to reflect the physical counts (DeHoratius et al., 2008). For the latter, the audit and count must be conducted on the item level instead of the dollar level alone (Raman et al., 2001). However, this method comes along with some disadvantages. It is very costly for the organization since counting the entire inventory requires a shutdown of the operations and/or closing the facility to customers, leading to a loss of revenue (Kang & Gershwin, 2005). In addition, a manual inventory count of all items at a premise is very labour-intensive (DeHoratius & Raman, 2008; Wijffels et al., 2016), resulting in high labour costs. Since most retailers already spend approximately 10% of their sales on labour expenses, labour costs have a substantial impact on profits/losses made (Rekik et al., 2019). Therefore, it is important to keep this factor in mind when looking for solutions. Finally, there are possibilities of miscounts resulting in no guarantee that the manual counts accurately reflect the physically available inventory (Kang & Gershwin, 2005).

Developments in recent technologies make it possible to fully automate the full count by introducing radio-frequency identification (RFID) tags. Although this technology can significantly improve inventory record accuracy, implementing such technology is very costly. The costs for this technology implementation are one of the main adoption barriers, especially for organizations that handle large volumes of low-value items (Wijffels et al., 2016). However, systems using this technology can accurately provide inventory information (Kök & Shang, 2007).

2.3.2.2. Cycle counting

To reduce the burden of counting the entire inventory during physical audits, it is possible to only check a sample of items by using cycle counting (Wijffels et al., 2016). Cycle counting is defined as any process for verifying the correctness of inventory records by counting proportions of the physical inventory on an ongoing basis (Kök & Shang, 2007). Next to aligning the physical and recorded inventory, cycle counting can also be used to identify process problems or other causes that result in inventory record errors (Kök & Shang, 2007; Rossetti et al., 2001; Sheppard & Brown, 1993; Wijffels et al., 2016). The challenge of defining a cycle counting policy is to determine which records are most likely to be inaccurate such that the maximum number of inaccuracies can be found while minimizing the number of checks (Wijffels et al., 2016). According to Rossetti et al. (2001), six primary cycle-counting methodologies can be applied for defining a cycle-counting policy. These methodologies consist of a random sample, ABC, process control, opportunity-based, transaction-based, and location-based cycle counting. Many large organizations tend to use a hybrid version with combinations of these primary methods when defining their cycle counting policy (Rossetti et al., 2001).

Starting with a random sample cycle counting policy. This policy generates a random sample from the population of inventory records to be counted during a cycle count. Various sampling techniques can be applied. Examples of these techniques are constant population, diminishing population, and diminishing population with timing such that every SKU has the same probability of being included in the cycle count (Rossetti et al., 2001).

The ABC method is the most commonly used cycle count method used in practice (Kök & Shang, 2007; Sheppard & Brown, 1993). It is a variant of the random sample method in which the population is based on a Pareto analysis into three categories (A, B, and C) where every category has a specific counting frequency (Rossetti et al., 2001). An ABC classification based on the Pareto principle assumes that a small share of the population account for a large share of the output, followed by a moderate share of the population that is responsible for a moderate share of the output while a large share of the population represents only a small share of the output (Liiv, 2006). The categories can be determined based on various factors related to SKU measures such as dollar volume of inventory, frequency of transactions, units sold (Kök & Shang, 2007), or other criteria that are relevant to the organization. In addition, the classification does not have to rely on one single criterion, it is also possible to include multiple criteria to extend the ABC analysis (Rossetti et al., 2001). The idea behind this cycle counting method is that items that are more important given the classification scheme are counted more frequently compared to less important items. Counting frequencies per category are typically based on subjective priories of the organization. Like the random sample cycle counting method, different sampling techniques can be used to sample within each category (Rossetti et al., 2001).

Process control cycle counting involves only counting items that are easy to count. This method is perceived to be controversial in theory but effective in practice. It applies to inventory records that have an item count on multiple locations and an inventory listing of these locations that is available during the counting procedure. The employee that performs the inventory count is free to determine if it counts the location or not based on whether the item is easy to count, whether the item on location is misplaced or misidentified, or whether there is an obvious discrepancy. After the counting procedure is performed, the accuracy of the inventory records is based on the SKUs that were actually counted and included in the sample while the skipped items are not included (Rossetti et al., 2001). This method can be beneficial since it is fast and requires fewer employees since fewer items may be counted. However, it can result in items that are never counted since they are always skipped. These items should be included in other counting procedures. An additional disadvantage is that employees are not performing a 'blind' count since there is already a record available on the inventory listing, this might result in a statistical bias (Rossetti et al., 2001).

Furthermore, opportunity-based cycle counting consists of a policy where counting is performed if particular key events in the process occur. These key events can take the form of but are not limited to, recording an item, restocking an item, picking an item, or the inventory record dropping below a specific threshold. A counting event might be scheduled if one of these events occurs, depending on the counting logic of the company regarding the frequency of counting and deciding which items to count (Rossetti et al., 2001).

Transaction-based cycle counting makes use of the number of transactions experienced by an SKU when scheduling counting events. For example, if an SKU experienced 5 transactions, it is considered for counting and a counting event is scheduled. Like with opportunity-based counting, the business rules regarding the frequency of counting and deciding which items to count are important parameters for this cycle counting policy (Rossetti et al., 2001).

The final primary cycle counting policy according to Rossetti et al. (2001) is location-based cycle counting. This policy is quite similar to the process control cycle counting, however, the employee is not provided with the recorded count and is not allowed to skip a location if it is perceived to be hard to count. The fact that the sample for the cycle count is formed by location instead of the characteristics of the items is considered to be a disadvantage of this method (Rossetti et al., 2001).

Next to the six primary cycle counting policies explained above, here are some additional policies that are worth noticing. Raman et al. (2001) mention the "Zero-Balance Walk" method, which is a variation of cycle counting where employees identify items that are out of stock on a daily basis. If an item is out of stock, the recorded inventory is verified and adjusted if it is found to be inaccurate (Raman et al., 2001). This is somewhat similar to the solution of manually resetting the inventory record to zero if one experiences the unexpected pattern of having no sales for a product over a period of time. If the inventory manager sets the inventory record to zero, it overcomes the effect of freezing and the automated replenishment system can place orders again (Kang & Gershwin, 2005). However, a limitation of the latter solution can also result in false positives for low-demand products for which it is not sure if the inventory is zero if one does not experience sales. Overstocking occurs if one incorrectly sets the inventory record to zero, resulting in higher inventory costs (Kang & Gershwin, 2005).

Furthermore, in extension to the previously mentioned traditional policies, Wijffels et al. (2016) introduced an enhanced inventory cycle counting approach that uses both historical and current inventory data to identify items that are most likely to be inaccurate. Most cycle counting methods make static assumptions about the cause of inaccuracy which might not always be valid. An example of such a static assumption is that SKUs that are frequently moved are most likely to be inaccurate. Instead, the method of Wijffels et al. (2016) determines the causes of inaccuracy based on historical inventory data to avoid this problem. This is done by using data mining to classify items as being accurate or inaccurate based on classification models, logistic regression and a neural network in this case, that are developed on historical inventory checking data. Both models were found to outperform the random counting policy but failed to outperform other basic cycle counting policies such as the transaction or inventory-based policy. However, this might be because the number of transactions or size of inventory are dominant features that significantly predict IRI, therefore the classification models may practically do the same as the policies that only order by this feature. An advantage of these models is that no prior knowledge or expertise is needed since these models identify influential dimensions themselves. Instead, traditional policies need expertise from managers that identify the important features. Moreover, a logistic regression produces a weight vector that shows the influence of every feature on inaccurate records, this can help find and resolve root causes. Finally, these models can be frequently retrained with new counting data, so if there are any changes in the factors causing IRI, the model includes these factors, which is not the case if one uses a static assumption (Wijffels et al., 2016).

Combining deep learning in terms of a neural network in combination with a cycle counting routine to remedy IRI is also proposed by Amaya Silva (2022). Since more and more data becomes available nowadays, neural networks can be really helpful by introducing algorithmic-enabled decision-making since data does not need to be transformed by people because neural networks can automatically discover how to learn a simpler representation of a complex problem. The solution of Amaya Silva (2022) consists of training a deep learning algorithm in the form of a neural network to predict the probability of an inventory record being inaccurate using historical data. This model is thereafter used to predict inaccurate records after a specific period and recommends a set of products to audit for which there is a high chance that a discrepancy exists according to the model. Subsequently, an

employee counts the SKUs and corrects the inventory record if a discrepancy exists. Afterwards, the gathered data is used to retrain the deep learning algorithm such that the quality of the predictions can be continuously improved by feedback on the quality of the predictions. In the end, the model accounted for 56 variables including product characteristics, audit frequency, store characteristics, the time between audits, seasonal information regarding holidays, and many more. The results of using this solution are very promising since it was found that stores using the deep learning decision model were able to detect IRI 20% faster than stores that used traditional heuristics and recommendations from analysts (Amaya Silva, 2022).

Apart from the fact that cycle counting is less of a burden compared to physical audits, cycle counting also has some drawbacks that are important to mention. First of all, it occurs frequently that items cannot be found at their designated locations during a cycle count due to misplaced items by customers or employees. Especially in mass merchandise retailing with a large number of individual items, it is a challenge to find the items of interest during a cycle count (Kang & Gershwin, 2005). Moreover, the trade-off between inspection and inventory-related costs is typically not included if the managers are choosing the inspection cycle for every product. Especially if items are counted more frequently, the inventory records are more accurate resulting in lower inventory-related costs. However, on the other side, counting more frequently increases inspection costs (Kök & Shang, 2007). Finally, as already mentioned with physical audits, there are possibilities of miscounts resulting in no guarantee that the manual counts accurately reflect the physically available inventory (Kang & Gershwin, 2005).

2.3.3. Integrating IRI by robust inventory planning and decision tools

It is also possible for organizations to accept the fact that IRI exists, without eliminating the need for prevention and correction of IRI, and use inventory planning and decision tools that are robust enough to cope with IRI. DeHoratius et al. (2008) assume that the uncertainty around the physical inventory exists. This uncertainty is represented through a probability distribution of the inventory discrepancy at the SKU level based on discrepancy data of SKUs that are closely related to the selected SKU. This probability distribution can then be used as the basis of inventory management decisions instead of the traditional point estimate of the inventory record. However, given this probability distribution, developing an effective replenishment and audit policy based on the inventory distribution remains a challenge. Since optimization based on this distribution needs computationally expensive dynamic programs, heuristics are defined to come up with a practical solution. Simulations of the defined policies showed that they are capable of recouping a large part of the costs related to IRI without the usage of expensive tracking technologies like RFID (DeHoratius et al., 2008).

Furthermore, Kök & Shang (2007) defined a joint replenishment and inspection policy that aims at minimizing the total costs of incorrect inventory records and inspection. This policy uses a threshold inventory level for the inspection decision, meaning that the manager should only perform an inspection if the inventory record is equal to or below a specific threshold. For the inventory replenishment policy, a base-stock level is optimal. Both the threshold and base-stock level are dependent on the level of IRI in the system and the policy that is the result of combining these levels is called the “inspection-adjusted base-stock policy”, also abbreviated as IABS (Kök & Shang, 2007). Since optimizing this policy is very expensive in terms of computational costs, a simple heuristic is proposed that is near optimal. This heuristic suggested performing inspections in fixed cycles, suggesting a cycle count policy with same the cycle length as that of the IABS heuristic. However, IRI is often not taken into account if replenishment decisions are made between cycle counts. Therefore, the IABS heuristic is used to propose a new cycle count heuristic that adjusts the base stock level according to the number of periods since the last inspection, this adjusted base stock level is called

CCABS. The cycle length is determined by using either simulation or the results of the IABS heuristic. To sum up, when aiming at minimizing total inventory and inspection costs, base stock levels need to be adjusted across periods. The order up to level should be increased if the number of periods since the last inspection increases to accommodate the added uncertainty due to IRI (Kök & Shang, 2007).

In addition, Iglehart & Morey (1972) developed a logic for selecting a proper frequency of inventory counts together with modifications to a predetermined stocking policy to minimize the total costs per unit time of IRI and inspections. Typically, buffer stocks are designed to protect against variations in demand and lead time, while it does not include errors in inventory records. First, numerical formulas are derived for determining the buffer stocks and timing of inventory counts based on either the number of periods that have passed since the last count or the cumulative demand since the last count. This is to protect against shortages due to errors in inventory records assuming that the inventory counts perfectly match the true inventory level, meaning imperfect counts are not yet included. However, as already mentioned, it occurs frequently that counting errors occur during the counting process (Iglehart & Morey, 1972; Kang & Gershwin, 2005). To overcome this, the formulas for calculating the buffer stock mentioned above are extended for specific types of counting. These formulas can be used to numerically define optimal buffer and counting policies to cope with IRI due to errors in operational processes and counting procedures (Iglehart & Morey, 1972). The solution of extending the buffer stock, also called safety stock, from only protecting against uncertainties in lead time and demand to also protecting against uncertainty in inventory records is also proposed by Kang & Gershwin (2005). This method seems to be effective in preventing stockouts if small stock losses occur. However, the extra inventory results in higher inventory carrying costs and if stock losses tend to be large and/or if the inaccuracy is caused by a nonzero-mean stock loss, the solution does not perform well and is thus not a desirable solution (Kang & Gershwin, 2005).

Finally, if one is aware of stock loss for a specific SKU and one knows its stochastic behaviour, it is possible to decrement the recorded inventory with the average stock loss during a period. This will not eliminate the error since the actual value of the stock loss during a period is not exactly known. However, it is probably still better than keeping the original inventory record. Although it is a simple solution that is effective in keeping the stockouts low, there are also some potential disadvantages. With this solution, depending on the implementation, the inventory record can be negative and/or not integer, which is of course not possible in reality. Even more importantly, since an expected value is used to decrement the inventory level, deviations from this expected value can increase stockouts. If the actual stock losses are higher for a few consecutive periods while it is not physically checked, the concept of replenishment freezing may occur for a short specific period. On the other side, since the inventory records are decremented during every period, the inventory record will suddenly drop below the reorder point. This solves the replenishment freeze which eliminates extreme out-of-stock situations. If the actual stock loss is lower than the expected amount which is decremented during every period, inventory levels become larger and larger resulting in higher inventory carrying costs (Kang & Gershwin, 2005).

2.4. Academic relevance

The reviewed literature led to various insights into the importance of accurate records in a JIT supply chain and the concept of IRI, its causes, drivers, and potential solutions for reducing IRI. However, some concepts are not thoroughly covered by the academic literature yet. Most academic literature tends to describe the causes and drivers of IRI in general along with the fact that this concept can result in stockouts or holding excess inventory. However, no literature is found that proposes logics or methods that can be used to classify these causes based on historical inventory data. In addition, research regarding the distribution of the defined operational causes of IRI that subsequently result in stockouts

is also an uncovered topic in the literature. When looking at the proposed solution methods that are mentioned in the reviewed literature, most articles propose a variation of a cycle counting approach to correct IRI. These variations range from basic rules to more advanced models. However, no counting approach is found that tries to evaluate individual interactions with the stock and suspicious situations for potential errors. Instead, the rules or models evaluate criteria regarding the time since the last count or accumulated stock data to determine whether the record itself is expected to be correct or not. This is interesting since errors during individual interactions with the stock are actually causing the IRI. When looking at the defined gaps in the literature, it can be concluded that this thesis can contribute to the academic literature by gaining insights into these concepts that are not yet frequently addressed.

3. Research questions

3.1. Main research question

As described in section 1.2 containing the problem statement, this thesis aims to reduce Picnic's problem of having inaccurate inventory records in its manual FCs that can subsequently result in incomplete customer orders. Therefore, the main research question that is considered in this thesis is defined as follows:

RQ: How can inaccurate inventory records potentially resulting in incomplete customer orders in a manual fulfilment centre of Picnic be reduced?

3.2. Sub research questions

Since the main research question is quite general and is too broad to answer in one go, sub-research questions are defined based on the frequently cited 10-step problem-solving framework of Okes (2019) that aims at finding and correcting causes of an identified problem in a structured way. This framework, which is available in Figure 7, consists of a diagnostic phase containing a root cause analysis followed by a solution phase to be able to solve the problem. First, one has to define the problem, which was already done in section 1.2. Thereafter, it is important to define the boundaries of the problem and to make sure that the process is well understood before directly jumping to possible causes. Creating a high-level flowchart is suggested as a tool to complete this step (Okes, 2019). The boundary definition and the process description are already available in sections 1.3 and 1.4 respectively.

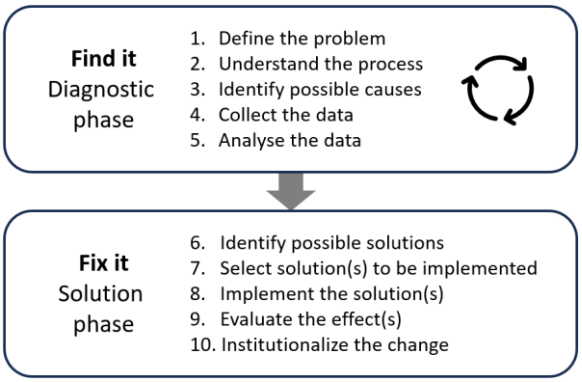

Figure 7 - Problem-solving framework of Okes (2019)

If the problem is defined and the process is well understood, the next step in the framework is to identify the possible causes of the problem. As found in the literature, IRI can result from various root causes. Therefore, it is useful to get an overview of all possible root causes of IRI that can subsequently result in incomplete orders in the context of a manual fulfilment centre of Picnic. Therefore, this step is represented by the first sub-research question which is defined as follows:

RQ1: "What operational errors can be considered as potential root causes of inventory record inaccuracy that may subsequently result in incomplete customer orders in the context of a manual fulfilment centre of Picnic?"

After answering *RQ1*, one should have a clear overview of the potential root causes of IRI that can result in incomplete orders. To be able to perform a diagnosis such that the true root causes of the problem are uncovered, data needs to be gathered (step 4) and analysed (step 5). Given the incomplete cases due to IRI that are provided by the current incompleteness dashboard as explained in section 1.2, gathering relevant data and analysing this data to allocate the true root causes to these incomplete cases provides insights into the distribution of the root causes that actually caused the inaccurate inventory records and therefore the incomplete customer orders. Hence, these steps are translated into the second sub-research question:

RQ2: "How can incomplete cases due to inventory record inaccuracy in a manual fulfilment centre of Picnic be allocated to their root causes?"

The results of *RQ2* should provide an overview and distribution of the root causes of the problem, which concludes the diagnostic phase of the framework and makes it possible to continue with the solution phase. This phase aims at the development of solutions to solve the found root cause(s) and therefore reduce the chance that the problem occurs. This is done by identifying the possible solutions (step 6) and selecting the solution(s) to be implemented (step 7). Referring to the literature regarding IRI solutions in section 2.3, it was found that there are generally three approaches to solving IRI, namely prevention, correction, and integration (DeHoratius et al., 2008). Prevention in the form of fully eliminating the found root causes by adjusting the WMS workflows can be quite difficult and might result in shoppers neglecting an overload of error messages or solutions that are blocking the operations, which is not desired to ensure a smooth and efficient fulfilment process. Besides if one can reduce the occurrence of some root causes, they might still occur in the future. Alternatively, the integration approach often results in holding additional inventory while it typically does not perform well when coping with large discrepancies (Kang & Gershwin, 2005), which is the case in Picnic's FCs. Moreover, integrating IRI in the inventory planning and decision tools does not eliminate the need for IRI prevention and correction methods (DeHoratius et al., 2008). Due to these reasons, Picnic prefers to have a solution that focuses on the correction approach. As mentioned in section 1.5, one of Picnic's initiatives to correct IRI and therefore reduce incompletes is using an alerting system. This system sends experienced shoppers to locations where suspicious stock counts have taken place to verify if the stock count was correct or whether IRI was created. However, this alerting system solely looks at stock counts that caused an absolute stock mutation quantity larger than a specific threshold and prioritizes the alerts according to this absolute stock mutation quantity in descending order. This can result in the fact that some IRI root causes are not yet covered by this alerting system and/or it requires a lot of checks that are found to be correct. Therefore, it would be useful to be able to assess, based on WMS data, whether a root cause of IRI occurred that creates a discrepancy which might subsequently lead to incomplete orders. This is represented by the third research question:

RQ3: "How can one assess Picnic's WMS data for suspicious situations that indicate that a root cause of inventory record inaccuracy has probably occurred?"

The results of *RQ3* can help with filtering/flagging suspicious situations that need to be checked to reduce the occurrence of IRI. A reduction of IRI means that there is a lower chance of incomplete orders that are caused by this concept. Since implementing this solution (step 8), evaluating the effect(s) (step 9) and institutionalization of the change (step 10) are out of the scope of this research due to resource and time limitations, these steps of the problem-solving framework of Okes (2019) are not covered in this thesis.

4. Methods

This section contains the research design, containing both the methods and data that were employed to answer every defined sub-research question. As mentioned in the previous section, these sub-research questions, and therefore the methodology that is used, are based on the 10-step problem-solving framework of Okes (2019).

4.1. Identifying root causes of IRI that can result in incomplete orders

Okes (2019) described a root cause analysis (RCA) as a deductive thinking process that first involves the development of hypotheses about the potential causes of a problem after which empirical evidence is gathered to support or refute each hypothesis. Therefore, to be able to allocate root causes to incomplete orders quantities that are the result of IRI due to the internal management of stock, it is important to get an overview of the possible root causes. These root causes are the hypotheses, which can thereafter be evaluated based on the available data. Okes (2019) proposed three methods to do so, which are treating every step in the process flow chart as a possible cause, creating a logic tree that identifies all possible causes at each level of the system, and using a brainstorming session to come up with a cause-and-effect diagram. The first two alternatives are perceived to be more structured, scientific, and logical than the last one. When using the flow chart method, the process that is investigated must be standardized, which is not always the case for Picnic's fulfilment process. Besides, it is less powerful than a logic tree since causes might also occur in between the steps of the process. Instead, a logic tree, which is a simplified version of a fault-tree analysis that breaks down the system into logical, incremental cause-and-effect relationships, is perceived to be the most powerful since it can cover all causes and prevents making gigantic leaps of faith (Okes, 2019). These incremental cause-and-effect relationships are created and visualized in the form of a tree by reconstructing the timeline of events that caused the problem to occur. This process tends to uncover various system deficiencies (related to procedures, policies, poor practices, training, etc.) that cause the poor outcomes (Latino, 2005, 2015, 2022). The fact that hypotheses are validated using hard evidence instead of hearsay results in an analysis that is based on evidence instead of the ideas of the loudest expert in the room (Latino, 2022). A logic tree is therefore perceived as a proper RCA method since it is an evidence-based and disciplined approach that seeks to do RCA 'right' the first time (Latino, 2015).

Due to these reasons, it was decided to create a logic tree for identifying the potential root causes. This logic tree was constructed by evaluating the causes and drivers of IRI that were found during the literature review in section 2 for their possibility of occurrence in the context of Picnic's fulfilment process. In addition, to get a better view of the fulfilment process and potential errors, questions were asked to employees that are related to this process. Besides, the fulfilment process and its WMS flows were experienced in practice by performing tasks as a shopper in two different FCs.

4.2. Allocating incomplete cases due to IRI to their root causes

4.2.1. Method selection

Once the potential root causes of IRI that subsequently lead to incomplete orders were identified, data had to be collected and analysed to indicate whether a defined cause did or did not contribute to the incomplete order quantity due to IRI. There is no dominant or straightforward method to do so since this heavily depends on the problem that is investigated and its context. However, according to Okes (2019), there are four basic steps for the data analysis phase. First, one must be clear about the hypothesis that is tested. Second, one should predict what the data would look like if the hypothesis were true. Subsequently, the data must be analysed and interpreted to see whether the defined hypothesis can be supported. Finally, one should consider other conclusions that might result from the data. Although this approach can assign root causes to a problem, it is important to keep in mind that

the findings can never be proved to be 100 per cent true because the data is rarely complete (Okes, 2019).

When looking at the problem of Picnic, it is worthwhile to mention that it does not consist of finding the root cause(s) of one individual problem. Instead, every incomplete case due to IRI can be perceived as a single problem that should be individually analysed for its root causes. Analysing one incomplete case can already take quite some time because many possible root causes must be evaluated. However, Picnic experiences many incomplete cases due to IRI per month, meaning that manually analysing every incomplete case will be almost impossible since this process becomes too time-consuming. Therefore, a solution had to be developed that can automate this root cause analysis.

e Oliveira et al. (2022) analysed 35 academic articles regarding automatic root cause analysis methods to provide an overview of the frequently used methods for automating this usually time-consuming process. It was found that this classification problem is often solved by applying either classification/association rules or machine learning techniques. Although machine learning techniques can be very powerful, these techniques require data on the (independent) response variable that can be used for training a machine learning model. Since no data was available in which incomplete cases due to IRI were already allocated to specific root causes (i.e. labelled), it was hard if not impossible to apply this method. Besides, in practice the root causes for this problem are typically uncovered by comparing particular attributes (e.g. quantity, timestamp, location, freshness date, location type, event type) of specific WMS events with each other. It was perceived to be hard to capture this comparison that is very context specific in features that can be used for machine learning techniques. Instead, the decision was made to develop a so-called rule-based expert system based on classification rules that can automatically allocate root causes to a problem. These classification rules contain expert knowledge that is used to check whether the historical inventory data of an incomplete case matches a specific pattern that represents a specific root cause. An example of this approach is described by dos Santos Nicolau et al. (2017) who successfully combined a logic tree with IF/THEN rules to automate the root cause analysis in case of a shutdown of a nuclear power plant. This was done using backward inference, which refers to the process of starting with an assumption and trying to prove it with the developed rules (dos Santos Nicolau et al., 2017).

A rule-based expert system is a computer program that solves complex decision problems in a specific domain by simulating the thought process of a human expert. This allows the computer program to perform tasks that would normally require a human expert. (Abraham, 2005; Gupta & Singhal, 2013; O'Keefe et al., 1986; O'Keefe & O'Leary, 1993). Next to the advantage of producing faster solutions than human experts, originally irreplaceable human experience is captured and preserved, a system is developed that is more consistent than human experts, and the need for human expertise is minimized (Abraham, 2005). This expert system usually consists of a knowledge base, a rule engine, and a user interface, which is visualized in Figure 8. To create such a system, small fragments of human knowledge are collected and coded into rules to create a knowledge base that can be used to reason through a problem. These rules are conditional statements that link given conditions to an outcome. The rule engine applies these rules to the sample input provided by the non-expert user via the user interface after which it returns the results of the analysis to the user interface and non-expert user respectively (Abraham, 2005).

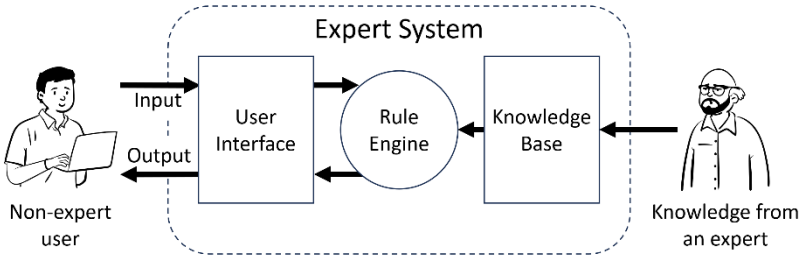

Figure 8 - Expert system structure

4.2.2. Data collection

Before constructing the rule-based expert system, a dataset was built by creating a SQL query that can be run on Picnic's data warehouse to gather all relevant historical inventory data that was needed during the creation, usage, and validation phase of this expert system. This dataset combined the WMS event log data with the WMS stock mutations data such that all relevant information regarding the performed actions, their stock mutations, timestamps, locations, quantities, freshness dates and employees was available. To be able to create this dataset, first, the complex dynamics between the WMS events and stock mutation table had to be thoroughly analysed and well understood to make sure to have a correct dataset as a result.

The majority of expert systems are built by a developer who constructs the knowledge base in a purely empirical manner in which the knowledge is iteratively discovered without reference to any underlying theory. The developer of an expert system must become a 'near-expert' of the applicable domain to be able to create such a system (O'Keefe & O'Leary, 1993). Therefore, this dataset was used by the developer to manually analyse incomplete cases due to IRI, both individually and with experts, that served as input for the development as well as the validation of the expert system. The selected time window for the historical inventory data belonging to an incomplete case was 7 days before and 3 days after an incomplete case occurred. This window was selected since, after asking several experts, it was found that the root cause of incompletes due to IRI can usually be found within this time frame. However, typically several hundred or a few thousand stock interactions are recorded for an SKU in this time interval while probably only one or a few of these interactions were wrong and subsequently led to the inventory records being incorrect. This could be compared to finding a needle in a haystack, especially when analysing the data by going through every row of the dataset in a tabular format. Remus (1987) found that, especially in cases with high environmental complexity, graphical aids tend to outperform aids in a tabular format. Therefore, a dashboard was developed in Tableau (data visualisation software) that visualised all historical inventory data in one view such that it was easier to spot related activities and potential mistakes. An example view of the developed dashboard is available in Figure 9 below.

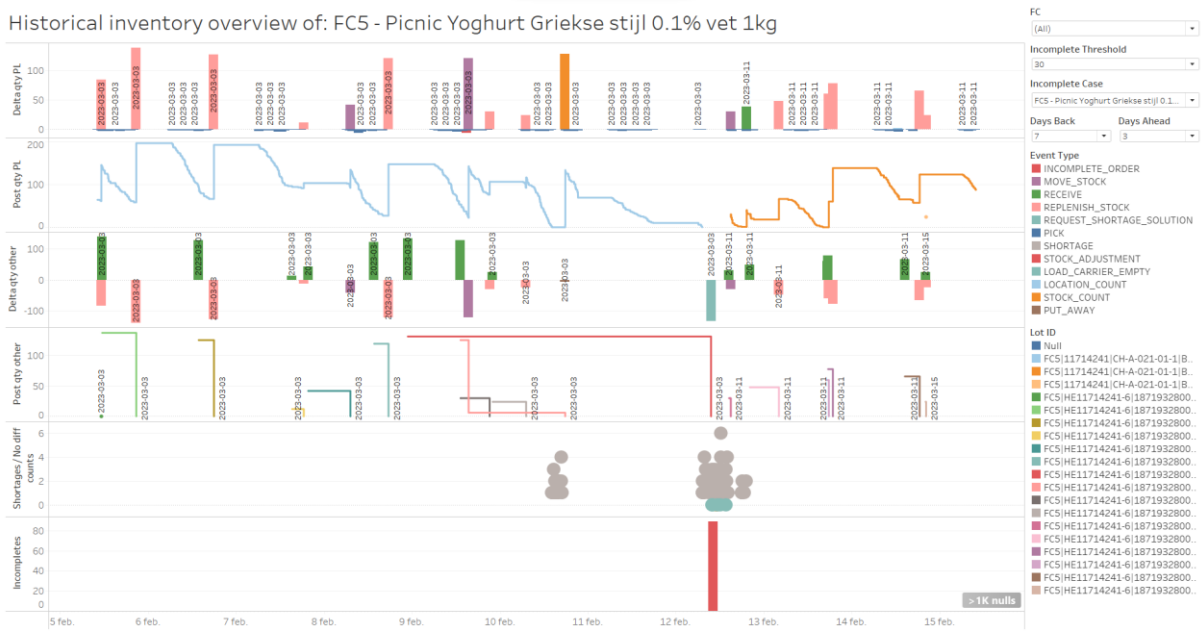

Figure 9 - Historical inventory dashboard

The dashboard was constructed such that one could select an incomplete case due to IRI in the drop-down menu in the upper right corner, which returned the inventory data of that incomplete case. Extra filter options were included to filter the list of incomplete cases in the dropdown menu based on the FC and the minimal incomplete quantity of the incomplete case. Besides, one could adjust the visualized time window, which was by default set to 7 days before and 3 days after the day of the incomplete case. Below the filter options, two legends were available that showed which activity and lot (which is the combination of an FC, article, location, and freshness date) were represented by which colour in the graphs. The view itself consisted of 6 subgraphs. The first subgraph represented the stock mutation quantities on the pick location, where every colour represented a specific activity. When hovering over these bars, additional information regarding the event, timestamp, location type (pick location, buffer location, load carrier, etc.), pre/delta/post CU quantity, freshness date, impacted locations, shopper and ID for tracing the activity back in the data warehouse became visible in a tooltip (this was also the case for all data points in the other subgraphs). The second subgraph contained a line chart that represented the (CU) stock levels after the event in subgraph 1 took place. This was on a lot level, meaning that if multiple freshness dates were available, every lot had its own colour. Subgraphs 3 and 4 contained the stock mutations and post CU stock levels of events that happened on all locations other than a pick location. Again, every lot had its own colour. Subgraph 5 showed the called shortages and shortage resolution requests.

4.2.3. Building the solution

After the development of the historical inventory dashboard, the rule-based expert system was developed by using this dashboard to manually analyse incomplete cases due to IRI for the hypothesized root causes that were defined in the logic tree resulting from *RQ1*. This manual analysis was performed both individually and during meetings with relevant employees (experts) and aimed to find a generic logic that could be used to accept the applicable hypothesis of the logic tree. This logic was subsequently translated into a rule and programmed using Python. After programming the initial rule, it was applied to a sample of incomplete cases due to IRI after which the cases that were assigned to this hypothesis (root cause) were evaluated with the dashboard to check its validity. From this moment, multiple iterations of finetuning the programmed rule (and its parameters) and validating its results were performed. This iterative process was used to be able to come up with a rule that was usually able to correctly accept the hypothesis if it should be accepted and reduce the occurrence of false positives.

The sample incompletes due to IRI that was used for this development contains incomplete cases due to IRI from January 2023 through April 2023 with an incomplete CU quantity larger or equal to 5. This decision was made together with employees of Picnic since very small incomplete quantities are hard if not impossible to assign to a specific root cause, which might lead to the fact that a lot of hypotheses would be invalidly accepted. Besides, it is possible that these small quantities cannot be allocated to a root cause by solely looking at the historical inventory data since these discrepancies might also happen by small unrecorded errors such as shoppers picking 7 instead of 6 items of a specific SKU. The decision to only look at incomplete cases due to IRI with an incomplete CU quantity higher or equal to 5 implied that approximately 65% of the total incomplete CU quantity due to IRI during these months was included in the scope of the expert system.

After programming the individual rules, which served as the expert system's knowledge base, the rule engine was programmed which made it possible to apply them in the appropriate sequence. To be able to provide the rule engine with the required input data, a simple user interface in the form of Python IDE was used in which SQL queries could be provided that load the incomplete cases due to IRI to be assessed and their historical inventory data directly from Picnic's data warehouse in the Python

environment. Besides, the parameters for every rule were also provided via this user interface and could be easily adjusted. After the required input data was loaded into the Python environment, every incomplete case in the provided table containing incomplete cases to be assessed was evaluated by filtering the historical inventory dataset on the data belonging to this specific incomplete case, whereafter the rules were applied. The resulting root cause allocation was thereafter returned in a tabular format with a column containing all provided incomplete cases followed by a column for every root cause that contained the incomplete quantity that could be assigned to that specific root cause.

It was argued that it is well possible that an incomplete order can be explained by multiple independent root causes. For example, it can occur that an SKU experienced 10 incompletes due to IRI on a day at which a negative discrepancy of 20 CUs and 30 CUs are found resulting from an error during the replenishment and receiving processes respectively. Since stock in Picnic's FCs is not linked to a specific order, both errors could have an equal chance of causing the 10 incomplete items. This concept was also mentioned by Picnic employees during the development of this tool. It was therefore possible for the expert system to accept multiple hypotheses for the same incomplete case. In addition, it was also possible that incomplete quantities could not be (fully) allocated to any of the root causes. The quantity that could not be explained by the developed rules was therefore assigned to an unknown root cause bucket. This quantity could be the result of a root cause that was not covered by the expert system or due to the fact that the rules were unable to detect the root cause for this incomplete case.

To get an overview of the root cause distribution over all evaluated incomplete cases, the quantities that were assigned to the root causes were distributed according to their impact on the incomplete case such that the sum of all root causes was equal to the incomplete quantity of the incomplete case. This distribution was calculated as follows: For every root cause (excluding the unknown root cause bucket), the assigned quantity was divided by the sum of all root cause quantities (excluding the unknown root cause bucket) after which it was multiplied by the incomplete quantity subtracted by the quantity that could not be assigned to any root cause. After applying this formula, every incomplete case was distributed over its applicable root causes based on their contribution to the incomplete case.

4.2.4. Validation

Like with any software, the developed expert system may provide unwanted results due to so-called 'bugs'. Besides, since an expert system is a model of human knowledge and reasoning, the system will never be perfect due to the fact that a model is a simplified representation of reality. However, because an expert system is built to replace a human expert, it is critical to show that the developed system is in some sense correct (O'Keefe & O'Leary, 1993). To get insight into the quality of the decisions that are made by the system, it was validated according to the expert system validation framework of O'Keefe & O'Leary (1993). This framework proposes methods to establish criteria, test criterion and construct validity, maintain objectivity, and ensure reliability.

To be able to validate the expert system, a measure had to be developed that reflects the expertise of the expert system. Typically, the accuracy measure is used, which compares the number of answers that are in line with those of an expert to the total number of provided answers. Based on this measure, a minimum level of competence called the acceptable level of performance, was defined (O'Keefe & O'Leary, 1993). When looking at the type of expert system that is created, e Oliveira et al. (2022) state that the accuracy measure is useful for the validation of automated root cause allocation in which no data set with root cause labels is available, which applies to the case of Picnic. Besides, the main focus of automating a root cause allocation process is not on the predictive capabilities, instead, it is important if the root causes that are extracted correspond to the real root causes that should have been detected (e Oliveira et al., 2022). Therefore, it was decided to use this accuracy measure for

validating the developed rule-based expert system. It was agreed that the system is considered to perform at an acceptable level if this accuracy measure was at least 80%.

There are two concepts of validation, the first one is criterion validation, which refers to the concept of testing the accuracy as described above. Secondly, construct validation is to validate the logic on which this system is built against theory. Criterion validity is dominant in expert system validation since the rules in expert systems are usually purely built by iteratively discovering knowledge in an empirical manner instead of grounding it with theory (O'Keefe & O'Leary, 1993). Since this was also the case for the developed expert system, the decision was made to focus on criterion validation. The use of test cases was found to be the dominant method for this type of validation (O'Keefe & O'Leary, 1993) and was therefore also applied for validating the expert system. This method includes presenting test cases to both experts and the system, after which both solutions are compared. It is very important to maintain objectivity during these tests, this can be done by making the test as independent from the validator (the developer) as possible. Finally, after the criterion validation is performed, the reliability of the expert system should be evaluated, this refers to checking whether the rules in the knowledge base are a reliable representation of the expert's knowledge (O'Keefe & O'Leary, 1993).

To validate the expert system according to the principles mentioned above, a 2-hour during testing session was organized with two Quality of Service (QoS) employees who have a detailed understanding of the fulfilment process and are usually analysing the inventory data for incomplete cases. During this session, the accuracy of the expert system was tested by analysing 50 incomplete cases due to IRI with the developed Tableau dashboard and evaluating whether the expert system provided the same results as the experts. Since analysing individual cases can be quite time-consuming and given that the available time for testing was 2 hours, a sample size of 50 was selected such that the test could be finalized within this testing session. If one wants to use test cases for the validation of an expert system, it is important to have an appropriate sample of test cases. First, any used test cases in the development should be discarded. Moreover, the system should be validated on a cross-sectional basis by randomly sampling the test cases using stratified sampling such that the test sample is defined independently from the validator and represents the distribution of the entire population (O'Keefe et al., 1986). This means that if the expert system found that 20 per cent of the incomplete orders are caused by a specific root cause, approximately 20 per cent of the sample should also consist of randomly selected cases to which this root cause is assigned. Therefore, the test sample consisted of 50 incomplete cases due to IRI that were sampled from all incomplete cases of May 2023 with an incomplete quantity larger or equal to 5 by using a random stratified sampling technique. Finally, during and after the development process, the programmed rules were regularly explained to the experts after which it was evaluated whether they were a reliable representation of the knowledge that was used for the root cause allocation process.

4.3. Solutions for assessing WMS data for suspicious situations that can potentially cause IRI

To be able to assess WMS data for suspicious situations that can potentially cause IRI, two approaches were explored. First, it was examined whether the knowledge gained during the development process of the rule-based expert system for assigning root causes to incomplete orders could be used and converted from historical analysis into creating more specific or additional real-time alerts. Second, it was explored whether it was possible to predict the correctness of individual stock count activities by training a logistic regression model on the data gathered by the current version of the alerting system. The aim of the second approach was to create a filter for the current alerting system to increase its effectiveness. Besides, it could potentially be used to enhance the developed real-time alerting rules that use stock counting activities resulting from the first solution approach.

4.3.1. Converting root cause allocation logic into logic for real-time alerts

The method of converting the rule-based expert system logic for automatic root cause allocation into a logic for real-time alert generation was inspired by dos Santos Nicolau et al. (2017). This research found that it is possible to create a rule-based expert system based on IF/THEN rules that can automate the root cause allocation in case of a nuclear power plant shutdown as well as the prediction of the emergency condition of this nuclear power plant to assess whether something is wrong or not. In contrast to the root cause allocation that used backward inference, the prediction used forward inference, which means that one starts with known facts which are used by the rules in the knowledge base to take actions or to create new facts and conclusions (dos Santos Nicolau et al., 2017). This problem is quite comparable to Picnic's problem that is researched in this thesis. Therefore, it was decided to explore this solution direction for its possibilities.

To be able to do this, the root causes that were covered by the developed rule-based expert system were evaluated for their possibility of transforming the logic from a historical analysis into a real-time assessment. Therefore, the rules had to be converted such that they could be applied to Picnic's WMS endpoints instead of historical data from Picnic's data warehouse. These WMS endpoints are APIs (Application Programming Interfaces) that extract real-time inventory information from WMS during the operations of the fulfilment process. It has to be mentioned that it was not possible to convert the logic of all identified root causes due to the nature of these root causes and the moment at which the root causes become apparent in terms of data. For example, many root causes only become visible at the moment the discrepancy is discovered and corrected, in these cases it would not make sense to create alerts since the discrepancy is already corrected.

To get insight into the performance of this solution method, it would have been best to implement the alerting rules such that they could be applied on WMS endpoints and tested in practice. However, due to limited time and resources, the implementation and testing in practice were out of the scope of this research. Besides, according to the found root cause distribution, a lot of the covered root causes by the real-time alerting rules were edge cases resulting in incomplete orders (less than 1% in the root cause distribution). Hence, it was expected that these root causes do not frequently occur, which made testing the rules in practice even harder. Therefore, it was not possible to assess whether alerts were created for cases in which the root cause occurred and IRI was present, also known as true positives, and whether alerts were created for cases in which the root cause did not occur and no IRI was present, also known as false positives. Instead, it was evaluated whether alerts were created for individual cases that are assigned to the root cause for which the rule was created. This was done by recreating the WMS endpoints based on historical data in Picnic's data warehouse and applying the logic to these recreated endpoints. Using this approach, it was possible to get an indication of whether the underlying logic of the developed alerting rule could catch these root causes. Besides, the results of the root cause distribution for incomplete orders due to IRI were used to get insight into the maximum potential of these alerts on reducing incomplete orders, however, this should not be perceived as a hard estimate but as a rough indication of the maximum potential.

4.3.2. Logistic regression model for predicting the correctness of stock counts

An alternative to the use of the rule-based expert system for real-time classification is the application of machine learning (Dreiseitl & Ohno-Machado, 2002). However, if one wants to train a machine learning model, data is needed on the response variable. Thanks to the alerting system that was already in place, data was gathered on the correctness of individual stock counts and stored in an alerting database. Therefore, it was possible to explore the application of machine learning techniques for predicting the correctness of individual stock counting activities. The goal of this solution approach was to create a filter for the current version of the alerting system to increase the effectiveness of the

checked alerts. Besides, if the model performs well, it could potentially be used to enhance the developed real-time alerting rules that use stock counting activities resulting from the solution approach described in section 4.3.1. There are various machine learning techniques available and each of them has its pros and cons. Given that the response variable of this classification problem was binomial (the count is correct or incorrect) and Picnic preferred models that can be well interpreted, a logistic regression model was explored for its possibilities.

A logistic regression can be compared to multiple linear regression, however, instead of having an unbounded continuous response (dependent) variable, the response variable of a logistic regression is binomial. The result of a logistic regression consists of the impact that each independent variable has on the odds ratio of the dependent variable. This method has the advantage of analysing the association of all variables together, which avoids confounding effects. One of the challenges is selecting the independent variables to include. Usually, many independent variables are selected whereafter one tries to find variables that have a significant effect. However, since samples are used, one has to be careful for spurious results. It is for example well possible that variables are statistically significant while having no theory to link it to the response variable. Moreover, the statistical power of a model tends to decrease if more independent variables are included. Because of this, it is possible to miss an association because the model is saturated resulting in the fact that it is not sensitive enough to detect the association. In contrast to many deep learning models, logistic regression has low complexity and has the advantage that its results are well interpretable (Sperandei, 2014). In addition, Wijffels et al. (2016) found that a logistic regression model is not significantly outperformed by a more complex neural network when it comes to predicting whether an inventory record is correct or not, which is somewhat similar to the context of this thesis.

To be able to train a logistic regression model for the correctness of stock count activities, first, data in the alerting database was cleaned and prepared. Resolving an alert consists of performing a new stock count. The alert database contains data about the stock count activity that triggered an alert, including a unique identifier that allows the activity to be traced back into Picnic's data warehouse and data about the performed recount to verify the first count, including the resulting delta. Since there is some time between the first count and the count for verification and to cover small unforeseen discrepancies, a count was considered correct if the difference between the first and second count deltas was less than 10% of the first count delta. This metric, representing a proportional deviation to the first counting delta, was already defined by Picnic and preferred over an absolute deviation based on the logic that it is expected to be more likely to encounter some unforeseen errors if the delta of the first count is larger compared to a small delta. Besides, due to the structure of the alerting database and Picnic's data warehouse, the decision was made to exclude guided counts from the analysis as well as alerts that were created due to a delta resulting from multiple stock counts in a short time window (5 minutes).

After preparing the response variable, independent variables were selected, defined, and extracted from the data warehouse using an SQL query. The selection of these independent variables was done by evaluating relevant literature and discussions with multiple employees who have a lot of experience and knowledge of the fulfilment process. These variables ranged from features related to the shopper that executed the first count, the location, the article, and inventory-related features.

Once potential variables were selected, dummy variables were created for the categorical variables by a one-hot encoding procedure that excluded the first value. Subsequently, the data was split into a training and a testing dataset based on a 75%-25% split after which outliers of numerical variables were removed from the training dataset. The outliers were not removed from the testing dataset such that it represented real cases that are encountered in practice to get a better view of the practical

performance. The outliers were detected according to the interquartile range (IQR) rule. According to this rule, a value is identified as an outlier if the value is smaller than the first quartile subtracted by 1.5 times the interquartile range or larger than the third quartile plus 1.5 times the interquartile range. This rule was selected because it can be used for various distributions since it does not assume the data to be normally distributed (Jeong et al., 2017). However, instead of multiplying the IQR by 1.5, it was multiplied by 3 to limit the number of outliers that were removed. After that, it was checked whether multicollinearity existed for the numerical variables in the training dataset by calculating the VIF (Variance Inflation Factor) for every variable to check if any VIF score exceeded the value of 10. If the VIF score exceeds a value of 10, this indicates that multicollinearity is a problem and action should be undertaken (Montgomery & Runger, 2014). However, there is no true consensus regarding the maximum value of the VIF score and some researchers argue that VIF values higher than 5 can already indicate minor problems regarding multicollinearity (Montgomery & Runger, 2014). Therefore, variables that had a VIF score larger than 5 were examined more closely by checking their correlations with other variables. In addition, the numerical variables were standardized, and the training data set was balanced by random under-sampling the majority group before fitting the logistic regression model. Balancing the training data set was performed because of an imbalance between the number of correct and incorrect counts, which would have made the model biased toward the majority group. Random under-sampling, which consists of decreasing the frequency of the bulk class, was selected as the method for balancing the training dataset since it was found to be a simple, widely used, and effective resampling method (Itoo et al., 2021). To determine whether the used sample size is sufficiently large for the analysis, the rule of thumb for determining the minimum sample size for a logistic regression analysis defined by Bujang et al. (2018) was applied. This rule states that the sample size should at least have $n = 100 + 50i$ observations where i refers to the number of independent variables of the final model. Finally, to avoid an overfit model, it was checked whether every categorical variable (also dummy variables) had no fewer than 10 outcomes for every value of the response variable (Stoltzfus, 2011). The independent variables were selected according to the forward selection method, meaning that, starting with a small group of relevant variables, regressors were added to the model one at a time until no candidate variables provided significant results (Montgomery & Runger, 2014).

The logistic regression was performed using the 'Logit' model of the statsmodels library in Python which employs the Maximum Likelihood Estimation (MLE) method to estimate the parameters. In addition, for the random under-sampling process, the 'RandomUnderSampler' from the imbalanced-learn library was used and the data was standardized by employing the 'StandardScaler' of the scikit-learn library.

To get insight into the performance of the trained model for predicting whether a count is correct or incorrect, several performance metrics on the predictive capabilities were evaluated based on the testing dataset. Starting with the accuracy, which is the number of counts that were correctly predicted to be either correct or incorrect divided by the total number of counts for which a prediction was made. Since the data in the testing dataset was imbalanced, only looking at the accuracy score was not very useful. To overcome this, one can use the true positives (TPs), false positives (FNs), true negatives (TNs), and false negatives (FNs) that are found by

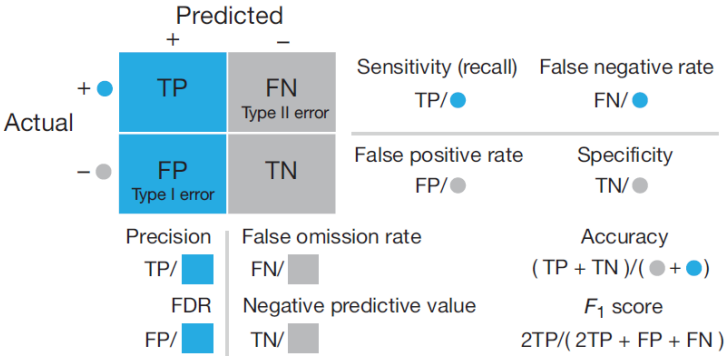

Figure 10 - Classification evaluation metrics overview (Lever et al., 2016)

comparing the predicted results are compared to the actual results using a confusion matrix. Based on this confusion matrix various evaluation metrics can be calculated (Lever et al., 2016), an overview of these metrics is visualized in Figure 10.

Ideally, the trained model should have very low numbers of both false negatives and false positives. This would mean that the model can correctly predict almost all correct counts being correct and incorrect counts being incorrect. Meaning that this would reduce time spent on alerts that are not necessary since the count was correct while only the incorrect counts are checked and corrected. The most popular and widely used metric is the F_1 score, this metric equally balances precision and recall which respectively focus on the proportion of predicted positives that are correct and the proportion of known positives that are correctly predicted (Lever et al., 2016). These metrics are calculated for every class after which a comparison is made to the confusion matrix and performance metrics of the alerting system that is currently in place. Finally, it is evaluated whether any of the findings can be used to strengthen the defined real-time alerting rules resulting from section 4.3.1.

5. Results

This section contains the results that were found by applying the methods as described in section 0 to be able to answer the sub-research questions that are defined in section 3.2. First, the potential root causes are identified after which they are allocated to incomplete orders. Finally, the findings on potential corrective solution approaches are presented.

5.1. Potential root causes of IRI

The identified logic tree with hypotheses for possible root causes of inaccurate inventory records that can result in incomplete customer orders is available in Figure 11 below. The root cause numbers in this figure are not relevant for this sub-research question but serve the purpose of being able to link this figure with the found root cause distribution in section 5.2.3.

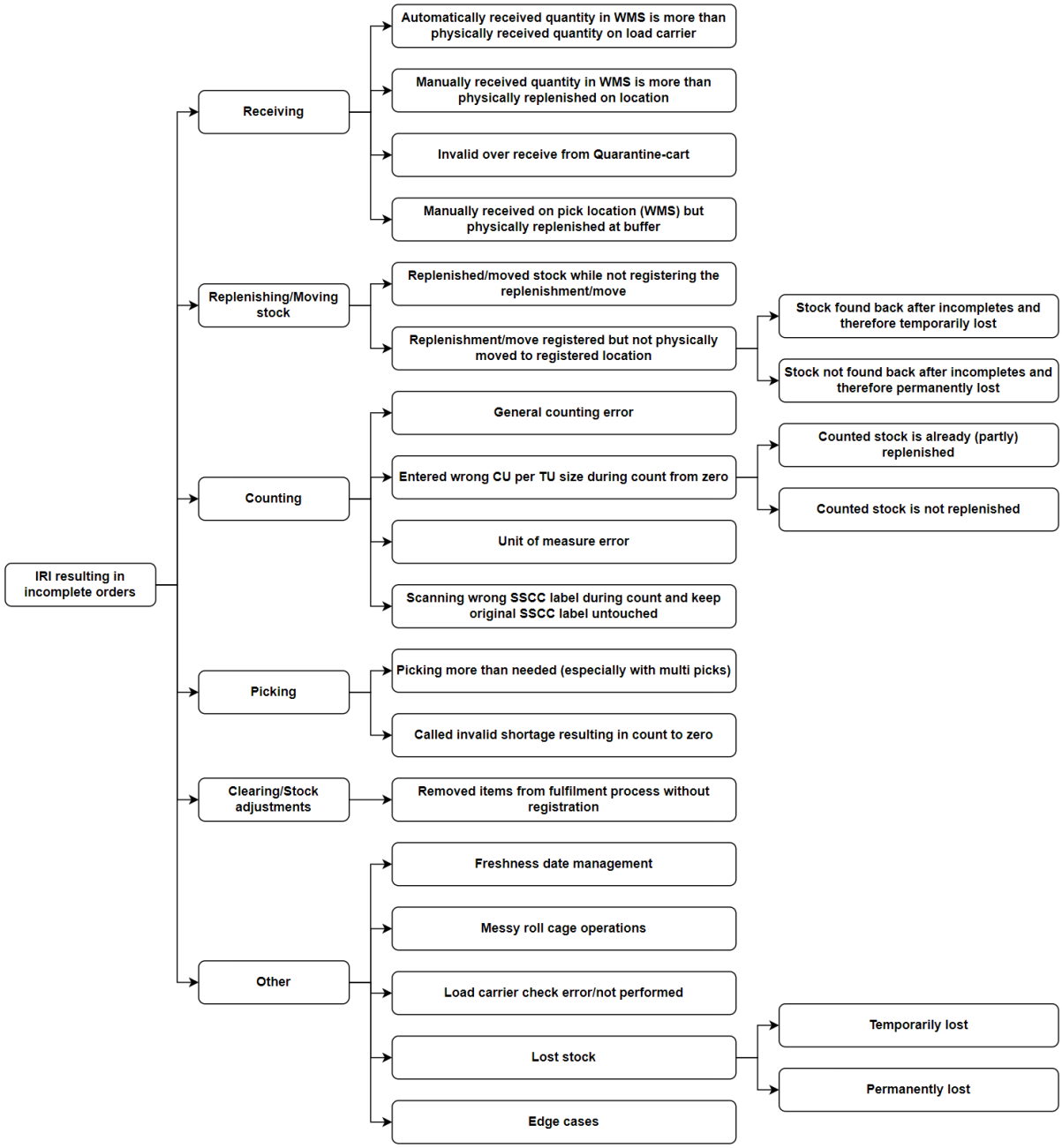

Figure 11 - Logic tree with root cause hypotheses

Since the initially developed logic tree is too granular to present clearly, this version is somewhat simplified by summarizing the hypotheses rather than constructing the entire timeline of events. The first branches consist of the key subprocesses of Picnic's fulfilment process and a branch of causes that are not related to a specific subprocess. This logic tree mainly focuses on errors that result in a negative discrepancy between physical and recorded inventory on at least one location. These errors were considered to be most important since incomplete orders due to IRI are typically caused by this type of discrepancy.

5.1.1. Receiving

Starting with receiving, four related hypotheses for potential root causes were identified. First of all, for products that are supplied by suppliers that provide stock information on the SSCC labels of the delivered load carriers, there may be a discrepancy between the provided stock information and the stock that is physically available on the load carrier. Since this type of inbound is automatically received on the load carrier during the unloading process, the stock is immediately recorded in WMS. If the automatically received quantity in WMS (recorded quantity) is more than the physically received quantity, a negative discrepancy is created that can cause incomplete orders. This case can be the result of picking errors on the supplier side, either one of Picnic's DCs or external suppliers, and is also described by DeHoratius & Raman (2008) and Kang & Gershwin (2005).

Second, it is also possible that errors are made during the receiving process of products that are supplied by suppliers that do not provide the stock information service. Therefore, this stock needs to be manually received on either the pick location or a buffer location. Although several checks are already included in the form of receipt checks and preventing the shoppers from receiving unexpected inventory, this process is still perceived to be error-prone. It is for example possible that the expected quantity is received in WMS (recorded quantity) while the stock that is physically replenished on the pick/buffer location is less, this again results in a negative discrepancy.

In addition, as already mentioned, if unexpected stock is trying to be received, either automatically or manually, it is sent to a quarantine location. Only experienced shoppers or captains can over-receive this stock. The procedure for over-receiving is that one checks and counts all locations of the item that needs to be over-received. If it was indeed found that the stock is unexpected inbound, the stock is received on a location via a solve issue activity. However, not all locations may be checked and properly counted, resulting in the duplication of stock that is already registered somewhere else.

Furthermore, since pick locations are checked for replenishment before the inbound stock is replenished on a buffer location, it is possible that shoppers manually receive stock that is supplied without stock information on the pick location after which it is physically replenished on a buffer location. A possible reason for this is that there was not sufficient shelf space on the pick location. However, if one does not register this as a stock move from the pick to the buffer location, the stock is lost for as long as it is not found back and registered on the correct location. This can lead to incomplete orders if the lost stock is needed for the fulfilment of customer orders.

5.1.2. Replenishing/moving stock

The second key subprocess in which errors can be made that result in IRI is the movement of stock from one location to another. For this subprocess, two scenarios are hypothesized. Starting with replenishments or stock moves that are performed without any registration. In this case, the stock is physically available on the location to which it is replenished/moved and recorded on the location from which it is replenished/moved, creating positive and negative discrepancies on these locations respectively. If this positive discrepancy is discovered and corrected by a counting event while the negative discrepancy on the original location is not, the stock is duplicated in WMS. This can result in

the fact that necessary orders are not generated by POM since it thinks that there is still sufficient stock available in the FC.

The other scenario consists of a replenishment/stock move being registered in WMS while the stock was not physically moved to the registered location. Since the stock is registered on this location but physically available on another location, this results in so-called 'ghost' stock. This is stock that is somewhere in the FC but it is not possible to access when using the information provided by WMS, which can result in incomplete orders at the moment this stock is needed. There are two variants of such 'ghost' stock, sometimes it occurs that the stock is found (after the incompletes occurred) and counted up on the location to which it was replenished/moved. This resulted in the stock being temporarily lost. On the other hand, it is also possible that this stock is not found back again, this can lead to the fact that stock is permanently lost.

5.1.3. Counting

Another key subprocess that is found to be very error prone that is therefore resulting in IRI and incomplete orders is the process that actually tries to correct discrepancies between recorded and physically available stock, namely the counting process (DeHoratius & Raman, 2008; DeHoratius & Ton, 2015; Iglehart & Morey, 1972; Kull et al., 2013; Rinehart, 1960; Woolsey, 1977). These can consist of regular counting errors, where the count resulted in stock being added to the inventory records in WMS while it was not physically available. This can happen on all different location types (e.g. pick locations, buffers, load carriers). However, some more specific counting errors can be made during an inventory count that are useful to discuss.

When performing a count from zero on a buffer location or a load carrier, one must manually enter the CU per TU quantity of the TUs that are counted. If a shopper enters a larger CU per TU quantity than the actual CU per TU quantity, the location is counted up with more CUs than physically available while the entered quantity TUs are correct. If this stock is not replenished/moved, it will only result in a negative discrepancy on the counted location. However, it is also possible that this stock is replenished/moved to another location. If this CU per TU error is made, all TUs that are replenished/moved from this counted location are automatically moved with this incorrect CU per TU, meaning that WMS moves more CUs than physically present. This also causes a negative discrepancy on the location to which the stock is replenished/moved and might therefore result in incomplete orders. Using wrong CU per TU sizes (case-pack sizes) is also stated by DeHoratius & Raman (2008) and Iglehart & Morey (1972) as a cause of IRI.

Next to CU per TU errors, it is also possible that shoppers use the wrong unit of measure during a count (DeHoratius & Raman, 2008). For example, if a shopper is asked to count the stock on a location that is filled with TUs containing multiple CUs. This stock is also registered as TUs, but instead of counting the number of TUs, the shopper counts the number of CUs. This means that the stock is multiplied by the CU per TU size which results in a negative discrepancy. If this error is not noticed in time, this negative discrepancy may prevent the purchase order management system from ordering new stock that is needed to fulfil the orders.

The last specific counting error that was identified in the logic tree is scanning the wrong SSCC label, which is mainly applicable to load carriers in the form of pallets. As already mentioned, all load carriers in Picnic's FCs are identified with an SSCC license plate label represented by a barcode. If one wants to perform an operation with this load carrier (e.g. move the load carrier, start a replenishment round, empty a load carrier, perform a put-away action, etc.) one has to scan this barcode. However, it sometimes occurs that old SSCC labels are not properly removed from the load carrier. If one wants to process this load carrier, it is possible that the shopper cannot find the correct label and thinks that

the load carrier does not contain any stock information according to WMS. In order to process this load carrier, it happens that the shopper performs a count from zero on the wrong (old) SSCC label after which it processes the load carrier. However, since the stock information on the original SSCC label still exists, it results in the fact that the stock is duplicated. If this original SSCC label is not counted down in time, it can cause incompletes since WMS still thinks that there is available stock while this is physically not the case.

5.1.4. Picking

The subprocess that makes sure that customer totes are filled with stock from the pick locations, the picking process, is also susceptible to errors that can cause IRI. One of these errors is that shoppers pick more items than requested by WMS. Especially during multi-picks, which is a pick request in which a multiple of the same SKU has to be picked, it is possible that shoppers unknowingly pick more items than needed to fulfil the order due to a counting error. If such an error is made, the requested quantity to be picked is subtracted from the inventory records while the physically picked quantity is larger. This subsequently results in a negative discrepancy. Although Sheppard & Brown (1993) did not find a significant effect, they still highlighted this concept as a potential cause of IRI, which was also confirmed by Picnic personnel.

Besides, if a shopper needs to pick an SKU from a pick location that is fully depleted, it calls a shortage in WMS to notify that the order was not picked due to a shortage. In some FCs, calling a shortage on a location that has an inventory level lower than the activated count to zero threshold results in counting the entire location down to zero. Although this saves time, a shopper may call a shortage while there is still stock available, resulting in an invalid countdown to zero. If this stock on the pick location was the only available stock in the FC (e.g. no buffer stock), this countdown to zero results in the request of a shortage solution that substitutes or cancels order lines that therefore go incomplete. Due to this reason, calling an invalid incomplete during the picking process can result in incomplete orders while these orders could be fulfilled by the stock that was still physically available on the pick location.

5.1.5. Stock clearing/adjustments

Next to the process of fulfilling orders by physically moving stock through an FC starting at inbound and ending in customer totes, it is also possible that errors occur during supporting processes. For example, if items are removed from the pick location due to any type of spoilage without proper registration (Kang & Gershwin, 2005). This spoilage can consist of products with a freshness date that cannot be sold anymore, damages, or quality issues. If this stock is discarded without proper registration, the stock is still available in WMS whereas it cannot be physically picked anymore. This negative discrepancy can therefore result in incomplete orders.

5.1.6. Other causes

After discussing possible hypotheses for root causes of IRI that can lead to incomplete orders based on the key subprocesses of the fulfilment process, some possible root causes were identified that are not directly related to one specific subprocess. Starting with freshness date management, since Picnic also sells perishable products, WMS assumes that products are processed according to the FEFO policy. However, this is not always the case, it is for example possible that items are not replenished on and/or picked from a pick location according to this FEFO policy, whereas WMS depletes the stock on a specific location according to this policy. Therefore, WMS thinks there is enough fresh stock available that can be used to fulfil customer orders while in reality this stock is too old to be sold and needs to be discarded. Since this stock was fresh enough to be sold according to WMS, orders are confirmed based on the stock that was physically too old.

Besides, roll cages can be used to move stock through an FC and consolidate multiple non-urgent buffer moves for efficiency. However, the registration of stock to and from this roll cage can be messy, leading to the fact that the stock stays on this roll cage according to WMS whereas it is physically not on this roll cage anymore. In practice, all roll cages should be empty at the end of a shift since they should be solely used as a mode of transportation in an FC instead of a storage location. However, this is not always the case. If these roll cage operations are not properly registered, these roll cages and/or the stock registered on these roll cages may get (temporarily or permanently) lost.

Typically, received load carriers from inbound are processed within one day by moving/replenishing them to a specific location. Though, due to various reasons, stock may be registered on a load carrier for a longer period. This can be correct since the stock was not yet processed, but on the other hand, it can be incorrect if the stock is only registered on this load carrier while it is physically not available on this load carrier. This can be the result of errors during the replenishment process or stock that is under-received and not calling load carriers to be empty at the end of a replenishment round. To make sure that these load carriers still exist and contain the stock that is registered in WMS, non-empty load carriers that are available in an FC for a longer period (according to WMS) should be checked to verify whether they still exist and whether the recorded stock is physically available. If this check is not (correctly) performed, meaning that this load carrier and/or its stock are not available anymore while they are not removed from WMS, it can result in incomplete orders since orders are confirmed based on this stock and/or new orders are not triggered by POM since WMS mentions that there is still enough stock available in the FC.

Finally, stock may get either temporarily or permanently lost due to not yet identified hypotheses (e.g. due to theft) and incomplete orders can be caused by rarely occurring edge cases that are not yet covered by this issue tree (e.g. wrong barcodes on products/deliveries, synchronization errors of WMS data).

5.2. Automatic root cause allocation tool

5.2.1. Covered root causes

Referring to the identified logic tree in Figure 11, not all identified hypotheses were covered by the developed expert system since they could not be analysed by solely looking at historical inventory data since these errors cannot be represented by the WMS data. This considered the root cause of picking more items than requested and removing items from the FC process for clearing/stock adjustment purposes without registration, these errors are therefore perceived as unforeseen discrepancies. To make the expert system more robust against unforeseen discrepancies, margins were included for some of the rules, however, it is also possible that these cases end up in the unknown bucket. Moreover, the edge cases that were not specifically covered can be either allocated to a less granular root cause or end up in the unknown root cause bucket. Besides, it was found that the root cause allocation logic of incomplete orders that is currently in place as described in section 1.2 is sometimes falsely assigning incomplete quantities to the IRI root cause. This is due to the nature of the current root cause allocation logic for assigning incomplete quantities to the IRI root cause which solely looks at negative stock counts close to the moment at which orders went incomplete. Therefore, an additional root cause was created that checks whether the incomplete was caused by IRI or not and updates the incomplete quantity to be explained by the expert system accordingly.

A simplified representation of the logic for supporting the identified root causes that are covered by the expert system is available in Table 5 of Appendix A. Next to this table, this appendix also contains some extra information regarding the knowledge base of the expert system.

5.2.2. Validation

During the validation session, it was found that the developed expert system incorrectly assigned root causes for 7 out of the 50 cases that were included in the test sample for validation. This implied that, based on the tested sample, the developed expert system accurately analysed 86% of the incomplete cases due to IRI. Besides, the cases that were perceived to be incorrect were not skewed to one or a few specific root causes and the output of the expert system was found to be quite stable when comparing the root cause distribution output of the development and testing dataset. Moreover, the main logic of the rules that were applied in the expert system was explained to the experts on an ongoing basis and was perceived to be a reliable representation of the knowledge that was used to allocate incomplete cases to their root causes. Since the accuracy measure exceeded the defined minimum level of competence of 80% and the logic was perceived to be reliable, the expert system was accepted to perform at an acceptable level.

5.2.3. Expert system output

When providing the developed expert system with all incomplete cases due to IRI from January 2023 through April 2023 having an incomplete quantity of at least 5 CUs (which can be considered a relevant number of cases) along with their historical inventory data, it produced the distribution of root causes as shown in Table 2 below.

Table 2 - Root cause distribution of incomplete cases having at least 5 incomplete CUs (2023/01/01 - 2023/04/30)

Root cause	%
RC0 - The incomplete is invalidly assigned to the IRI root cause	3.56%
RC1 - Automatically received quantity in WMS is more than physically received quantity on load carrier	7.54%
RC2 - Manually received quantity in WMS is more than physically replenished on location	15.36%
RC3 - Invalid over receive from Quarantine-cart	4.43%
RC4 - Manually received on pick location (WMS) but physically replenished at buffer	0.94%
RC5 - Replenished/moved stock while not registering the replenishment/move	4.23%
RC6 - Stock temporarily lost due to a registered replenishment/move that is not physically performed	3.52%
RC7 - Stock permanently lost due to a registered replenishment/move that is not physically performed	5.62%
RC8 - Counting error on pick location	8.56%
RC9 - Counting error on buffer location	0.80%
RC10 - Counting error on load carrier/roll cage	0.24%
RC11 - Counting error on other	0.03%
RC12 - Entered wrong CU per TU size during count from zero and stock already (partly) replenished	0.70%
RC13 - Entered wrong CU per TU size during count from zero and stock not yet replenished	0.30%
RC14 - Unit of measure error	0.21%
RC15 - Scanning wrong SSCC label during count and keep original SSCC label untouched	0.24%
RC16 - Called invalid shortage resulting in count to zero	0.04%
RC17 - Freshness date management	0.30%
RC18 - Messy roll cage operations	2.23%
RC19 - Load carrier check error/not performed	6.33%
RC20 - Stock was temporarily lost due to unknown cause	19.31%
UNKN - Root cause could not be assigned by expert system	15.51%

Based on these results, it was found that the developed expert system was able to allocate 84.49% of the incomplete items that belonged to an incomplete case due to IRI of at least 5 CUs. Given that these cases represented 65% of the total number of incomplete items due to IRI in the evaluated period, this implies that 55% of the total number of incomplete items due to IRI in the period from January through April 2023 was allocated to a root cause by this system. Interesting findings are that about 24% (RC4, RC6, RC16, RC20) of the analysed incomplete items due to IRI was caused by stock that was physically available in the FC but could not be found at the time the stock was needed to fulfil the customer orders since it was temporarily lost. This means that these orders did not have to be cancelled or substituted if this stock was correctly recorded. On the other hand, approximately 33% (RC1, RC2, RC3, RC7) of the analysed incomplete items due to IRI was assigned to stock that was registered in an FC

and were not recovered during the days after the incompletes occurred, meaning that the stock did not arrive in the FC or was permanently lost. To make this abstract concept a bit more specific, roughly 27% originated from errors during receiving activities (RC1, RC2, RC3), this is quite a high percentage given that there are already quite a few initiatives in place to prevent IRI that originates from inbound processes (e.g. receipt checks and quarantine locations). Another key process that seemed to be error-prone was the replenishment/stock move process, roughly 13% of the analysed incomplete items due to IRI was allocated to errors in this key process (RC5, RC6, RC7). Furthermore, not properly checking untouched load carriers that still contain stock according to WMS can be very damaging, as about 6% of the analysed incomplete items due to IRI was allocated to this root cause (RC19). Besides, the results implied that approximately 11% of the analysed incomplete items due to IRI could be explained by the activity that aims to correct discrepancies, namely activities related to counting (RC8, RC9, RC10, RC11, RC12, RC13, RC14, RC15). Finally, some root causes of IRI did not seem to have a high impact on incomplete orders compared to the other root causes (root causes with an impact below 1%). However, when analysing individual cases that are the result of these root causes, most of these causes seemed to cause a very large negative discrepancy (e.g. RC12, RC13, RC14, RC15). Therefore, it is important that, although they do not happen frequently, these root causes are not neglected since these large discrepancies can have a very high impact if these root causes occur.

5.3. Assessing WMS data for suspicious situations that can potentially cause IRI

5.3.1. Converting RCA logic

After evaluating all identified root causes for their possibility to transform the rule that is developed to allocate incomplete cases due to IRI to these root causes into specific real-time alerts, it was found that this transformation was possible for only a limited number of the root causes. This is the case since most root causes become apparent at the moment the stock is needed, just before incomplete orders are created, and at this moment the IRI is also immediately corrected. However, for the root causes of a replenishment without registration (RC5), entering the wrong CU per TU size during a count with or without replenishing the stock (RC12 and RC13), making a unit of measure error during the count (RC14), scanning the wrong SSCC label during a count (RC15), messy roll cage operations (RC18), and load carrier check error/not performed (RC19) it was possible to come up with a logic to create real-time alerts during the operations such that the suspicious situations that can potentially cause IRI can be verified and (if necessary) corrected at the moment they occur. A high-level description of the defined real-time alerting rules is explained in this section. For more details, the example Python code with comments for every rule is available in Appendix B.

Starting with the root cause of a replenishment that was made without registration in WMS and scanning the wrong (old) SSCC label during a count, these root causes were combined into one rule since the latter is a more specific root cause that would otherwise also be identified by the rule for catching a replenishment without registration. If the wrong (old) SSCC label is used, a shopper duplicates the stock by performing a count from zero on another SSCC license plate with the same quantity and freshness date in a period shortly after the SSCC was received while leaving the original load carrier untouched. The quantity, freshness date, and time of this count from zero can be used to check whether this stock was automatically received somewhat earlier during the process. On the other hand, if one replenishes stock without registration in WMS, a shopper may discover a positive discrepancy in the location to which the stock was moved. If one corrects this discrepancy by performing a stock count on this location, the stock delta resulting from the count in terms of a stock mutation can be used to evaluate whether this stock was potentially moved from another location without registration and therefore duplicated in WMS if the original location is not counted down. Since the root cause of using the wrong (old) SSCC label is more specific and uses more information,

this root cause is evaluated first and if it is not found, it continues to check whether any replenishment was made without registration in WMS. To be able to catch these root causes, for every count with a positive delta, it is first checked whether it was performed on an SSCC and whether there exists another active SSCC in WMS that was automatically received at most 24 hours before the count up having the same SKU, quantity and freshness date. If this is the case, an alert can be created that asks a shopper to check whether this original load carrier still exists or whether the stock was indeed duplicated by the occurrence of this root cause and should be corrected by a count down to zero. Otherwise, it is checked whether the delta originating from the count up was the result of discovering stock that was replenished without registration. If this delta is approximately equal (margin of 20%) to the stock that is available on any other location according to WMS (e.g. buffer, load carrier, roll cage) at the moment of the stock count, an alert can be created that sends a shopper to these locations to verify whether the stock is still there. If this is not the case, this stock can be counted down to zero to confirm that the stock on this location has already been replenished whereby the IRI that is created by this root cause is corrected.

Subsequently, for CU per TU errors during a count from zero, it was possible to create a rule for cases in which a CU per TU error was made after which this stock was not replenished afterwards. This is done by saving the CU per TU sizes that were used during the receiving events during the last week(s). Since the receive events represent the first time that the stock is registered in an FC, the CU per TU sizes that are used in these receive events are the only CU per TU sizes that are expected to be present in the FC. If any count from zero is performed with a CU per TU size different from the received CU per TU sizes or one (which represents a CU), an alert can be created to send a shopper to the location at which this suspicious count up was performed to verify whether the CU per TU size and count was correct. However, since alerts cannot always be immediately solved, this rule is extended for CU per TU size errors on stock that is subsequently replenished or moved to other locations before the original alert was checked. Since replenishments, stock moves, and put-away actions are automatically registered with this potentially wrong CU per TU size, this can also result in discrepancies being created further down the fulfilment process. Therefore, alerts should be created for all locations that can be linked (via a replenishment, stock move, or put away action) to the location at which the potential CU per TU size error was made. If the shopper finds this root cause, it can correct the IRI by counting down the stock to the physically available quantity.

It was also possible to use the logic captured in the root cause allocation rule for allocating incomplete orders to the unit of measure during a count error to create a real-time alert for this root cause. If a count is performed in which the inventory record after the count is equal to the inventory before the count multiplied by the CU per TU quantity of the counted lot, an alert can be created to verify whether a unit of measure error was made and correct the stock if this was the case. Unlike the CU per TU error, the interactions with this location do not have to be checked since the CU per TU size is untouched.

Since roll cages are meant to be processed during the shift in which it is created, a rule could be defined that checks whether the stock is on a roll cage for longer than 8 hours. If one saves the time at which the stock was moved to/counted up on a roll cage, one could create an alert for checking this roll cage as soon as the difference between the current time and the saved time of the move/count exceeds 8 hours. A somewhat similar logic can be used for load carrier checks that experienced errors or were not performed. Load carriers from inbound are typically processed within one day. Therefore, it would be useful to check stock on automatically received load carriers if the stock is on that load carrier for longer than 24 hours. Like the alerting rule for roll cages that exist for longer than 8 hours, it can be checked whether the difference between the time of receiving and the current time exceeds 24 hours.

If this is the case, a suspicious situation occurs and an alert can be sent to check whether the load carrier and its stock are still present in the FC. If not, the IRI can be corrected.

Finally, as soon as the stock on the locations for which any alert is created is depleted due to any interaction with this location, the alert for this specific location should be removed since that indicates that IRI this stock was not duplicated. Besides, if the stock is fully depleted, there is no stock to check on this location.

These converted root cause allocation rules could help reduce IRI by sending more specific alerts or additional alerts. If these alerts are solved, the created IRI can be corrected such that the physically available stock is aligned with the inventory records in WMS. Apart from checking whether an alert was created for specific incomplete cases that were assigned to this root cause by simulating the WMS endpoints, which was the case, a more detailed indication of classification performance was not attained. However, referring to the results of the *RQ2* that provides a root cause distribution of incomplete orders belonging to incomplete cases due to IRI having at least 5 incomplete CUs, it was found that the sum of the root causes for which an alert could be created is 14.24%. Based on these root cause analysis results, the maximum potential of this solution method was estimated to be able to reduce incomplete orders due to IRI (that belong to incomplete cases due to IRI with at least 5 CUs) by roughly 14% if the alerts are solved in a timely manner.

5.3.2. Logistic regression

5.3.2.1. Selected predictor features

To be able to train a logistic regression model to predict the correctness of individual stock counting actions, features were selected related to the shopper, the article, the location, and inventory data. Starting with features related to the shopper that executed the count, the number of counts performed by the shopper during his/her career and his/her job role are included as predictor variables for the model. First, based on (DeHoratius & Ton, 2015), it was expected that more experienced shoppers would be less error-prone than shoppers who only performed a limited number of counts. This is also one of the reasons why alerts are currently checked by more experienced shoppers. This feature was transformed by taking the natural logarithm since the data was heavily skewed to the left and had a very large range of values. Besides, any specialization and training of shoppers was expected to impact the error proneness of the shopper (DeHoratius & Ton, 2015), which is represented by their job role. Since Captains or Receipt and Flow Operators are better trained to perform actions correctly and are perceived to be more aware of the impact of IRI in an FC compared to regular shoppers, it was expected that they would be less likely to make an error during the counting process.

The article-related features were limited to the sales volume of an SKU in a specific FC to distinguish between slower and faster-moving products. This feature was calculated by summing the total number of picks in the FC during the 7 days before the count took place. This feature was perceived to be relevant since it was expected that fast-moving products are more prone to errors because they experience more interactions, which usually results in somewhat messier locations. In addition, fast movers usually result in FCs keeping more inventory on larger locations with other characteristics (e.g. twin shelves which are two shelves on top of each other having the same SKU). These reasons can increase the environmental complexity and it was therefore expected that counting fast movers have a higher chance of creating IRI compared to slow movers (Rekik et al., 2019; Sheppard & Brown, 1993).

The selected features related to the location of the count were the FC, temperature zone, and location type of the count. These features represent the environment in which the count was performed, which could have an impact on the accuracy of the counting process. First, the distinction between FCs was made since some FCs are typically messier than others, making the counting process more error-prone.

Besides, the leadership and training can differ slightly among FCs. Furthermore, the temperature zone was selected since the different zones have different characteristics. For example, counting at a temperature of -20°C can be less convenient than counting in an ambient temperature, making the counting process more susceptible to errors. Additionally, the picking circuits vary among temperature zones, such as shelves in the chilled temperature zone that are usually more organized and spacious than shelves in the ambient temperature zone. Moreover, the location type at which the stock count took place was selected since every type has its own characteristics in terms of environmental complexity. For example, pick locations only contain one specific SKU while a buffer location consists of a shelf that can be filled with various SKUs. This makes the environment of the stock count much more complex and therefore probably more prone to errors (DeHoratius & Ton, 2015).

The final feature category, which is related to the inventory, was represented by the quantity that is counted by the shopper, the stock confidence factor before the count, the sign of the stock mutation resulting from the count, and the number of freshness dates that were on the counted location (according to WMS). Starting with the counted quantity, which is the quantity that the shopper entered in WMS, it was expected that counting larger quantities increases the likelihood of resulting in IRI by being more prone to errors like losing the count, estimating the quantity instead of counting it, or other counting inaccuracies (DeHoratius & Ton, 2015; Kull et al., 2013; Woolsey, 1977). Second, the metric that is currently used to represent the confidence in the inventory record being correct based on the interactions with the location, the stock confidence factor, was selected. If this value is high, the number of interactions with the stock on this location is quite low, meaning there is less opportunity for IRI to be created and therefore resulting in a stock delta on the counted location. Third, the sign of the delta resulting from the stock count was included since, in general, it was expected that counts with positive deltas are less prone to errors than counts with negative deltas. This is due to the fact that counts with a negative delta can consist of stock that is maybe temporarily lost or overlooked during the count while stock should usually be there to result in a positive delta. Finally, referring back to the concept of environmental complexity (DeHoratius & Ton, 2015), it was expected to be more difficult if there were multiple freshness dates on the location at the moment of the count. Perishable products are registered and counted on the lot level, where every lot represents a specific freshness date. If one had to count an SKU on a location that contains multiple freshness dates, the process was more complex since one also had to correctly select the items belonging to the specific lot that was counted. Besides, it is also possible shoppers did not count all lots on the location, this could cause problems if one performed a stock count to correct errors in freshness date management while keeping the other lots untouched.

5.3.2.2. Evaluated features that were not included in the final model

Next to the features mentioned in the previous section, several additional features were explored but not included in the model due to insignificant results or a negative impact on the predictive performance of the logistic regression model. Shopper-related features that were excluded from the final model were shopper tenure defined as the number of days the shopper worked for Picnic, the number of counts performed during the hour before the count, and the number of elapsed hours since the shopper started his/her shift. Article-related variables such as the volume, weight, packaging type (e.g. bag, jar, not packed), multipack (yes/no), article category (e.g. fruit, vegetables, dairy, meat, bakery), brand tier (e.g. A-brand, price entry, private label) and the unit of measure that is counted (CUs or TUs) were also evaluated but excluded as features for predicting the correctness of a stock count. When looking at location-related features, the shelf height of the counted location was also included in the analysis as a categorical variable that represented the level of a shelf, however, it was excluded from the final model due to insignificant results. For features related to inventory, many additional variables were explored for their possibilities. These variables consisted of the quantity on

the location before the count (according to WMS), the absolute delta resulting from the count, the total stock level of the entire FC, the expected inbound stock, whether it was a count from zero, and whether it was a count to zero. Finally, variables considering the time of the day and the day of the week were also evaluated but not included in the final model.

5.3.2.3. Trained logistic regression model

For this analysis, alerting data was available for 16,703 individual unguided stock counting activities during a period from 2023-03-31 until 2023-06-15. During the outlier removal process, 1,036 rows were removed which was equal to 8.27% of the original training dataset (12,527 rows). This was caused by 66, 687, and 496 outliers for the count experience (log-transformed), counted quantity, and weekly SKU sales of an FC respectively. The resulting training dataset consisted of 65.23% of the counts being labelled while the other 34.77% was labelled as incorrect. After balancing the dataset by applying a random under-sampling procedure, the training dataset consisted of 7,990 counting activities that were used for training the logistic regression model. The VIF scores and resulting model estimates are available in Table 3 below.

Table 3 - Logistic regression results for predicting the correctness of a stock count activity (correct/incorrect = 0/1)

Independent variable	VIF	Coefficient	P value	Odds ratio	Lower CI	Upper CI
Intercept		1.53893	<0.001	4.65960	2.75120	7.89181
Count experience [Log]	1.19585	-0.16883	<0.001	0.84465	0.80374	0.88764
Counted quantity	1.97430	0.07234	0.029	1.07502	1.00732	1.14727
Stock confidence factor	1.62450	0.08212	0.006	1.08559	1.02353	1.15141
Weekly SKU sales of FC	1.25200	0.14229	<0.001	1.15291	1.09493	1.21395
FC3 [R]						
FC4	2.31709	-0.53607	<0.001	0.58504	0.50977	0.67144
FC5	1.71229	0.18897	0.030	1.20800	1.01827	1.43309
FC6	1.75224	-0.23376	0.004	0.79155	0.67637	0.92636
FC7	1.67366	0.08122	0.325	1.08461	0.92251	1.27520
FC8	1.69129	-0.33035	<0.001	0.71867	0.60875	0.84844
AGF [R]						
CS Agent	1.07917	-1.65847	0.001	0.19043	0.07103	0.51056
Canteen	1.02474	-1.64968	0.003	0.19211	0.06394	0.57725
Captain	3.03282	-0.93438	<0.001	0.39283	0.23436	0.65844
FC Operations Manager	1.02803	-2.13447	0.001	0.11831	0.03320	0.42159
FC Safety & Quality Officer	1.06334	-1.05315	0.003	0.34884	0.17229	0.70630
FC Supervisor	1.13086	-1.17300	<0.001	0.30944	0.16142	0.59317
FC Trainer	1.05745	-1.07926	0.004	0.33985	0.16427	0.70310
Receipt & Flow Operator	1.19035	-1.04768	0.001	0.35075	0.18755	0.65596
Shopper	6.67823	-0.85747	0.001	0.42424	0.25494	0.70595
Tech Lead	1.03894	-1.18180	0.058	0.30673	0.09044	1.04022
Ambient [R]						
Chilled	1.73045	-0.09700	0.097	0.90756	0.80936	1.01767
Frozen	1.11329	0.39703	<0.001	1.48741	1.21189	1.82555
Buffer [R]						
Consolidation	1.02065	-0.05465	0.889	0.94682	0.43889	2.04258
Flowrack Pick Location	1.29142	-0.26192	0.080	0.76957	0.57387	1.03200
Pallet Pick Location	1.48615	-0.36450	0.005	0.69454	0.53978	0.89367
Put Away Location	1.07346	0.57344	0.012	1.77436	1.13472	2.77457
Roll Cage	1.28976	0.26364	0.002	1.30166	1.10611	1.53178
Shelf Pick Location	3.29219	-0.33679	<0.001	0.71406	0.61599	0.82774
Negative delta [R]						
Positive delta	3.31537	-0.74685	<0.001	0.47386	0.42387	0.52974
One freshness date [R]						
Multiple freshness dates	1.11925	0.56764	<0.001	1.76410	1.34339	2.31657

R: Reference variable for dummies; Log: Transformed by taking the natural logarithm

It was found that all variables had a VIF score lower than 10, this indicated that the presence of multicollinearity was not very high and thus not expected to result in large problems. However, there was one variable, the shopper job role, that exceeded a VIF score of 5 by having a value of 6.68. After a more detailed investigation of the correlations with variables other than the other dummy variables for the job role variable, the maximum correlation was found to be -0.303118, which is still quite low. Given that the VIF score was still below 10 and the absence of high correlations with other variables, it was decided to not exclude this variable from the final model. The final selection of independent variables resulted in 29 variables that were used to predict the correctness of a stock counting action. Given the rule of thumb for the minimum sample size, the minimum sample size for this analysis was calculated to be $n = 100 + 50 * 29 = 1{,}550$, which was met since the analysed sample consisted of 7,990 counting activities. Finally, all categorical variables had at least 10 outcomes for every value of the response variable. This means that the assumption to prevent an overfit model was satisfied.

According to the results in Table 3, it was found that, except for some dummy variables, almost all included variables had significant results due to having a p-value lower than 0.05. However, the insignificant dummy variables that are part of a larger set of dummy variables representing a categorical variable were still included in the analysis since excluding these insignificant dummy variables would change the reference level of the categorical variable. Although these variables did not significantly differ from the reference group, there might be a non-significant difference which would therefore change the coefficients and significance levels of the other dummy variables if the insignificant dummy variable was removed.

Based on the logistic regression results in Table 3, a significant negative relationship was found between the natural logarithm of the count experience and the odds of performing an incorrect count. This suggests that, as expected, shoppers who performed more counts during their career reduce the likelihood of performing an incorrect counting activity. When looking at shopper specialization, it was observed that all shopper roles except for the Tech Lead have an impact that is significantly different from the AGF shopper role. It is interesting to highlight that, based on the odds ratios, receipt & flow operators (who are specialized in counting activities) are less likely to perform an incorrect count than captains and regular shoppers. Besides, captains have slightly lower odds of performing an incorrect count compared to regular shoppers. These results confirm the expected relationships. The other job roles that also show significant results are less dominant in participating in stock counting actions.

Furthermore, as expected, it was discovered that the number of weekly sales of an SKU in an FC positively and significantly relates to the odds of having an incorrect count. As already explained in Section 5.3.2.1, this is probably due to messier locations during the count and other characteristics of locations for fast-moving products.

When looking at location related features, it was observed that FC4, FC6, and FC8 have significantly lower odds of performing an incorrect count compared to FC3. Where FC4 has the lowest odds followed by FC8 and FC6. For the other FCs, FC5 and FC7, it was found that a count performed in FC5 has significantly higher odds compared to FC3 while it was observed that FC7 was positive but insignificantly different from FC3. For temperature zones, no significant difference was found between the effect on the likelihood to perform an incorrect count when comparing the chilled to the ambient temperature zone whereas counts performed in the frozen temperature are significantly more likely to occur compared to the ambient temperature zone. The latter was also expected before training the model. Finally, compared to a buffer location, it was indeed revealed that counting activities performed on pick locations (except for flowrack pick locations) have significantly lower odds of resulting in an incorrect count. To be more specific, pallet pick locations are less likely to result in an incorrect compared to shelf pick locations. However, this difference is quite limited. On the other hand,

it was observed that a count performed on a roll cage and put away location are more likely to result in an incorrect count compared to a buffer location. The results for the consolidation and flowrack pick location were negative but insignificant, where the first showed highly insignificant results.

For the inventory-related features, the logistic regression analysis revealed that the odds of performing an incorrect counting activity significantly increase as the quantity that is counted by the shopper increases, which was also expected. Additionally, as hypothesized, a positive and significant relationship was found between the stock confidence factor before the count and the odds of a counting activity being incorrect. This means that if a count that resulted in a delta, it is more likely to be wrong if Picnic's WMS is more confident that the original inventory record was correct on the location of the count. Moreover, if the counted quantity is larger than the registered quantity before the count, resulting in a positive stock mutation, the odds of the counting activity being incorrect are found to be significantly lower compared to counts that resulted in a negative stock mutation. Lastly, it was observed that locations that contain multiple freshness dates during the count are significantly more prone to an incorrect counting activity. This effect was expected due to the concept of increased environmental complexity.

5.3.2.4. Performance

When testing the trained logistic regression model on the unfiltered imbalanced testing dataset, it was found that the model was able to accurately predict the correctness of 60.99% of the evaluated stock count activities. Although this metric provides some insight into the performance of the trained model, it is not very relevant for understanding its predictive capabilities. Instead, the results were compiled in a confusion matrix to evaluate the number of true negatives (TN), false negatives (FN), true positives (TP), and false positives (FP). This confusion matrix is available in Figure 12 below along with the confusion matrix of the current alerting system.

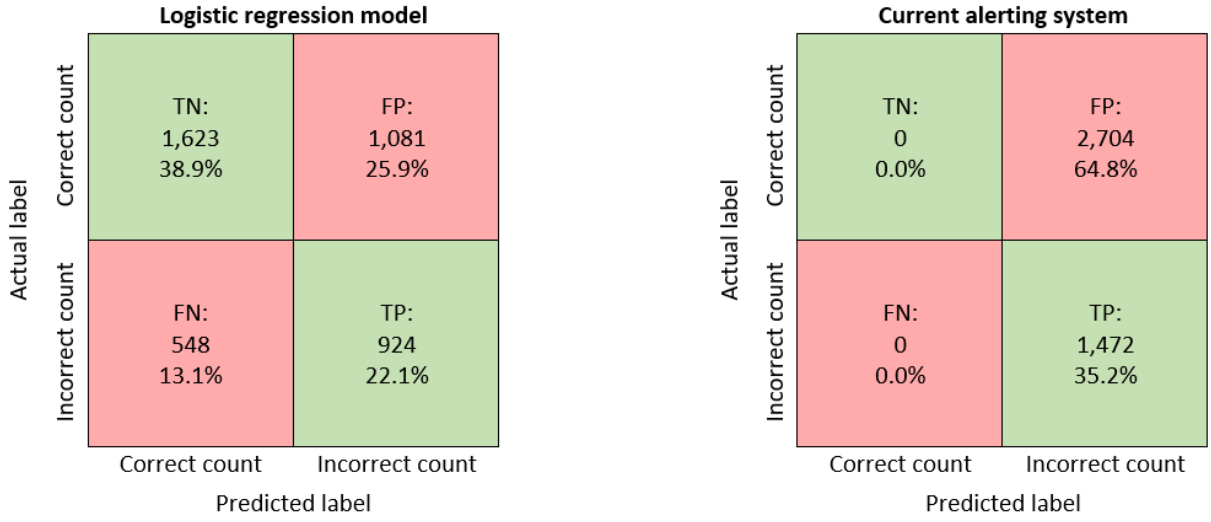

Figure 12 - Confusion matrices of logistic regression model and current alerting system

Based on the confusion matrix of the logistic regression model, it was observed that 60% of the correct counts were predicted to be correct and 63% of the incorrect counts were predicted to be incorrect, which represents the recall for each class. On the other side, out of the counts that were predicted to be correct, 75% was correctly predicted to be correct while only 46% of the counts that were predicted to be incorrect were incorrect, this represents the precision for each class. When balancing both metrics for every class by calculating the F_1-score, the model attained an F_1-score of 0.67 and 0.53 for counts being labelled correct and incorrect respectively. A summary of these metrics along with their support and averages are available in Table 4 below. Based on these statistics, it was found that the

trained logistic regression model was able to predict the majority of the correct counts to be correct and the majority of the incorrect counts to be incorrect. However, if one wants to successfully apply this model in practice to filter the alerts that need to be checked, it would be best if the number of false positives and false negatives were as close to zero as possible since this would save time by only verifying incorrect counts while not missing any counts that created IRI.

Table 4 – Classification report

Label/Aggregate	Precision	Recall	F_1-score	Support
Correct count	0.75	0.60	0.67	2,704
Incorrect count	0.46	0.63	0.53	1,472
Accuracy			0.61	4,176
Macro avg	0.60	0.61	0.60	4,176
Weighted avg	0.65	0.61	0.62	4,176

However, when referring to the confusion matrix of the logistic regression model, it was observed that 37% of the incorrect counts by which IRI was created would be missed and 40% of the correct counts would be checked while no IRI was created. Especially the first situation of missing counts that created IRI can be harmful to the organization because this IRI can result in holding unnecessary inventory or incomplete orders (Rekik et al., 2019). Since this thesis focuses on incomplete orders that are caused by negative discrepancies, the number of negative discrepancies in the false negatives was evaluated. There are two ways to check whether the incorrect count created a negative discrepancy. First, if the delta from the initial count was positive and the delta of solving the alert was negative, this means that a negative discrepancy was present due to counting too much stock during the first count. Additionally, if the delta from the initial count was negative and the delta of solving the alert was also negative, this means that the negative discrepancy was not fully corrected during the first count. Using this logic, it was observed that 258 out of the 548 false negatives (47%) consisted of a negative discrepancy.

To get insight into the performance of this solution approach, its confusion matrix and resulting performance metrics were compared to those of the current alerting system that sends alerts for all suspicious counting activities that pass a specific absolute delta threshold. Sending shoppers to all counting events that passed this threshold resulted in 1,472 true positives and 2,704 false positives while the number of true negatives and false negatives are both equal to zero since the testing dataset only consisted of counts that were included in the alerting system. Based on these numbers, it was found that the alerting system that is currently in place had an accuracy of 35.25% for the evaluated counting activities in this analysis. Due to the absence of counts that exceed the absolute delta threshold that are predicted to be correct by the current alerting system, it was only possible to compute and compare the precision, recall and F_1-score for the incorrect count class. The precision for the incorrect count class is represented by the accuracy measure (0.35) while the recall is found to be 1.00 since all suspicious counting activities were included. This led to F_1-score of the incorrect count class of 0.52. If one had applied the logistic regression model as a filter on the current alerting system, this would mean that the percentage of effective alert checks would increase from 35% to 46% (i.e. recall of the incorrect count class) while the percentage of incorrect counts that are caught by this tool would decrease from 100% to 63% (i.e. precision of the incorrect count class).

To provide an estimate of the potential impact of implementing this logistic regression model as a filter on the current alerting system, it was found that, based on this testing data, the number of unnecessary alert checks would be reduced by 60% at the cost of not checking 37% of the incorrect counting activities of which 47% can potentially cause incomplete orders. Finally, because of its relatively low predictive capabilities and the absence of highly dominant features that almost always resulted in a correct or incorrect count, it was decided to not use this model and/or its findings to improve the developed real time alerting rules that use stock counting activities in section 5.3.1.

6. Conclusion

This research aimed to provide a better understanding of the operational root causes of inaccurate inventory records in Picnic's Dutch manual fulfilment centres that can subsequently lead to incomplete orders and develop solutions that can potentially enable Picnic to reduce this harmful phenomenon. Therefore, the Picnic's manual fulfilment process was analysed to be able to answer the main research question: "*How can inaccurate inventory records potentially resulting in incomplete customer orders in a manual fulfilment centre of Picnic be reduced?"*. In order to do so, three sub-research questions were formulated which will be answered in this section and put in relation to the academic literature. After that, these findings are used to draw a conclusion on the main research question.

Starting with the first sub-research question: *"What operational errors can be considered as potential root causes of inventory record inaccuracy that may subsequently result in incomplete customer orders in the context of a manual fulfilment centre of Picnic?"*. Analysing the fulfilment process in combination with findings from the literature review and discussions with relevant employees led to the identification of various hypotheses for potential operational errors that can cause IRI and therefore incomplete customer orders in the researched context. These errors range from specific shopper mistakes during the fulfilment subprocesses of receiving, replenishing/moving stock, counting, picking, and clearing/adjusting stock to errors that are not related to one single fulfilment subprocess such as freshness date management or stock being registered on a load carrier or a roll cages that is physically not available anymore. The majority of the identified root causes are applicable in various contexts and also stated in academic literature (e.g. DeHoratius & Raman, 2008; DeHoratius & Ton, 2015; Iglehart & Morey, 1972; Kang & Gershwin, 2005; Rinehart, 1960; Sheppard & Brown, 1993; Woolsey, 1977) whereas others are more specific to the context of Picnic, such as manually (over) receiving stock or freshness date management. In addition, root causes related to the registration of moving stock within an FC (e.g. stock moves without registration, registered stock moves that are not moved to the registered location, and messy roll cage operations for buffer moves) were not covered in detail by the reviewed academic literature due to their researched context, which often consisted of regular retail stores.

To be able to get insight into the distribution of the root causes of IRI that subsequently resulted in incomplete orders the identified root causes had to be accepted or rejected for a bulk of individual incomplete cases. This is represented by the second sub-research question: *"How can incomplete cases due to inventory record inaccuracy in a manual fulfilment centre of Picnic be allocated to their root causes?".* Given that there was no historical data available in which root causes of incomplete cases due to IRI were already assigned and because manually analysing individual incomplete cases for their root causes was too time-consuming, repetitive, and needed a lot of expertise, it was found that it is possible to automate this process by developing a rule-based expert system. By capturing expert knowledge and programming it into rules to create a knowledge base, root causes could be automatically allocated to 84.49% of the incomplete orders that belong to incomplete cases due to IRI having a minimum CU quantity of 5 (representing 65% of all incomplete orders due to IRI) with an accuracy of 86%. This illustrates that developing a rule-based expert system can be used to automate the root cause analysis of incomplete orders due to IRI for an online retailer. This conclusion supports the academic literature on the ability of (rule-based) expert systems to automate a wide range of applications, of which diagnosis is one of them (e.g. Abraham, 2005; dos Santos Nicolau et al., 2017; Gupta & Singhal, 2013).

Next to that, the developed expert system provided a root cause distribution that provides insights into the impact of the potential root causes that were hypothesized by answering the first sub-research question. To the best of the writer's knowledge, this insight into the distribution and impact of the root

causes is not yet covered by academic literature on this topic since most papers only contain a general description. Interesting conclusions from this distribution are that 24% of the analysed incomplete items due to IRI were caused by stock that was present in the FC but not available for picking due to IRI on specific locations. Moreover, it was found that errors in the process of receiving stock are quite impactful in causing the analysed incomplete items due to IRI (27%), followed by errors in the internal movement of stock (13%), and errors made during the attempt to correct IRI via counting activities (11%). Besides, the output of this automated root cause analysis supports various stated root causes in the academic literature, such as errors in automatic receiving stock (DeHoratius & Raman, 2008; Kang & Gershwin, 2005), errors during the counting process (DeHoratius & Ton, 2015; Iglehart & Morey, 1972; Kull et al., 2013), of which using a wrong unit of measure is more specific (DeHoratius & Raman, 2008), and using wrong case-pack sizes (DeHoratius & Raman, 2008; Iglehart & Morey, 1972).

Finally, after understanding the root causes of IRI that subsequently resulted in incomplete orders, the third sub-research question considered the development of solutions based on a corrective solution approach that can potentially detect discrepancies resulting from the identified root causes. Instead of counting policies that use aggregate values of inventory data that are usually proposed by the academic literature (e.g. Amaya Silva, 2022; Rossetti et al., 2001; Wijffels et al., 2016), the goal was to catch individual situations that actually cause the IRI. This sub-research question was defined as follows: *"How can one assess Picnic's WMS data for suspicious situations that indicate that a root cause of inventory record inaccuracy has probably occurred?"*. First, it can be concluded that it is possible to convert some of the developed rules for the historical root cause analysis of incomplete cases due to IRI into more specific or additional real-time alerting rules for verifying/correcting the created IRI. This is in line with dos Santos Nicolau et al. (2017) who showed that the IF/THEN rules of an expert system for a root cause analysis can be transformed into real-time alerts using forward inference. Transforming these rules was possible for replenishments/moves that are performed without registration, using the wrong (old) barcode label on a load carrier, using the wrong CU per TU (case-pack) size, counting with the wrong unit of measure, messy roll cage operations, and load carrier checks that are not (correctly) performed. Based on the found root cause distribution, the maximum potential of this solution method was estimated to be able to reduce incomplete orders that belong to incomplete cases due to IRI with at least 5 CUs by roughly 14% if the alerts are solved in a timely manner.

Second, a logistic regression model was trained to be able to predict the correctness of stock counting activities such that alerts in the current alerting tool can be filtered based on whether it is expected that IRI is created or not. Although several significant predictor variables were found and the model was able to correctly predict the majority of the counting activities being correct/incorrect, the current version of the model was not perceived to have predictive capabilities that are high enough to employ this model in practice. While the number of unnecessary alert checks is expected to be reduced by 60%, 37% of the incorrect counting activities (of which 47% can potentially cause incomplete orders) would be skipped if this model were implemented, the latter can be harmful to the company. Although it is not advised to implement the current version of this model, academic literature on drivers of incorrect counts is supported, such as employees are expected to make fewer errors if they are more experienced or better trained (e.g. DeHoratius & Ton, 2015). In addition, faster movers are more prone to creating IRI which is mentioned in Rekik et al. (2019) and Sheppard & Brown (1993) where the latter was not able to find a significant relationship although it expected to do so. Also, counting larger quantities being more error-prone (e.g. DeHoratius & Ton, 2015; Kull et al., 2013; Woolsey, 1977) which was also expected by Sheppard & Brown (1993) who counterintuitively did not find significant results to support this hypothesis. In addition, the statement by DeHoratius & Ton (2015) on expecting more errors as the environmental complexity increases is also supported by various variables related

to this concept. However, the specific drivers of having multiple freshness dates on a location during the count and the difference in location types on which the count was performed, which represent a form of environmental complexity, were not yet addressed by the reviewed academic literature. This was also the case for the finding that stock counts resulting in a positive stock mutation are less likely to be incorrect compared to stock counts resulting in a negative stock mutation.

To sum up and draw a conclusion on the main research question based on the answers to the sub-research questions, potential root causes of IRI that can result in incomplete orders were identified after which it was possible to develop a rule-based expert system to be able to automatically allocate these root causes to incomplete orders that are the result of IRI. In addition to providing insight into the root cause distribution, some of the developed rules could be converted into real-time alerting rules for detecting suspicious situations in which a root cause of IRI that could result in incomplete orders has probably occurred. If such a situation is detected, an alert can be created to verify whether the root cause occurred and correct the created IRI if it did. When assuming that these alerts are solved in a timely manner, the maximum potential of this solution method is estimated to be able to reduce incomplete orders belonging to incomplete cases due to IRI having at least 5 CUs by roughly 14%. Besides, a logistic regression model was explored for its ability to predict the correctness of individual stock counting activities with the aim of improving the current alerting system and supporting the developed real-time alerting rules. Although the predictive capabilities of the current version of the logistic regression model were not perceived to match the performance that is needed for successful implementation, this method provided insights into drivers of incorrect counting activities.

7. Discussion

7.1. Academic and practical implications

As described in the previous section, this thesis supports the academic literature by confirming already identified root causes of IRI, drivers of incorrect counting activities, and effectiveness of methodologies. Moreover, several gaps in the literature are addressed by identifying and confirming additional root causes of IRI (which are sometimes very context-specific), automating the root cause analysis of incomplete orders due to IRI, getting insight into the root cause distribution of incomplete orders due to IRI, performing research on IRI correction approaches that are based on the occurrence of individual situations in which IRI is created instead of using aggregate values of inventory data, and finding additional drivers of incorrect counting activities.

Apart from academic implications, the research and findings in this thesis also resulted in practical implications that can be very helpful for Picnic specifically and organizations that are coping with problems similar to the problem that is addressed in this thesis. First of all, due to the complex dynamics between WMS events and their resulting stock mutations, Picnic did not yet have a method for combining these data warehouse tables into one historical dataset that contains all interactions with stock in a manual FC. For this research, a SQL query was constructed that makes it possible to construct this dataset which can also be used for analysing other problems related to WMS events and their stock mutations. Besides, the created dashboard that visualizes this dataset makes it possible to analyse historical inventory data and spot relationships between WMS events more easily instead of going through data in a tabular format, this can be helpful to uncover various operational problems in the fulfilment process of manual FCs.

Second, before this research was performed, Picnic did not have an idea of the specific root causes of incomplete orders due to IRI and their share in creating these incomplete orders. Thanks to this thesis, Picnic is provided with more detailed insights into these causes. Additionally, the developed rule-based expert system is flexible in a way that it can perform the root cause allocation on any dataset (e.g. different periods) as long as it is provided with the correct data (same data structure) and no significant changes are made to the fulfilment process. This allows Picnic to have a better understanding of the causes of incomplete orders due to IRI during a specific period such that it can explain why it happened and potentially intervene if necessary. Next to Picnic, other organizations that are coping with a similar problem can be inspired by the identified root causes, the method of developing a rule-based expert, and the main logic of the defined rules to uncover the root causes (and their distribution) within their own environment.

Finally, regarding solutions for reducing IRI and therefore incomplete orders, this thesis offers Picnic a solution direction that can help with correcting IRI that is based on real-time alerting rules that assess WMS data for suspicious situations in which a root cause likely occurred. Moreover, although it is not advised to implement the current version of the logistic regression approach for predicting the correctness of stock counting activities, the findings provide insights into drivers of these incorrect counting activities. These insights might be useful for Picnic and other organizations that want to understand this concept and its drivers or further develop a similar solution approach to improve its predictive capabilities. Referring to the identified drivers, if Picnic would like to increase the accuracy of its stock counting activities, it could for example be beneficial to provide additional training to its shoppers or assign counting tasks to shoppers that are more experienced in this task. Besides, Picnic could instruct or warn its shoppers to be extra careful during counting activities if the quantity to be counted becomes larger or if the location has multiple freshness dates. Finally, it is advised to keep the FC and its locations as organized as possible to limit the environmental complexity of the stock count.

7.2. Limitations

To acknowledge the limitations of this research, the findings in this thesis are based on data from Picnic's manual FCs that serve the Dutch market. Therefore, it is not known whether these findings also apply to the manual FCs that fulfil orders for the other markets (i.e. Germany and France) or other organizations having the same problem within a different context or fulfilment process. Besides, regarding the automated root cause allocation, the incomplete orders that are analysed are limited to incomplete orders belonging to an incomplete case due to IRI with at least 5 CUs. Meaning that the root cause allocation is performed for IRI that resulted in relatively large impacts. However, if one is also able to include incomplete orders belonging to incomplete cases below this threshold, one may observe shifts in the root cause distribution. Besides, the incomplete quantities that could not automatically be allocated to a root cause are still quite large, this might be the result of not including all root causes (e.g. picking more than necessary or removing stock from an FC without registration) or the fact that most developed rules focus on a countdown to zero instead of all countdown events. Furthermore, as already mentioned, a historical root cause analysis can never be proved to be 100 per cent true because relevant contextual information was not present (Okes, 2019), this in combination with the fact that a rule-based expert system is a simplified version of reality, will result in a root cause allocation will never be perfect (O'Keefe & O'Leary, 1993). Instead, the root causes that are most likely to explain the incomplete orders are allocated during the root cause analysis. Lastly, due to the limited time available for a validation session, the sample size for validating the developed expert system was limited to 50 cases, which is quite small. Therefore, slightly different accuracy measures may be observed when more cases are evaluated or if the random stratified sampling method provides a validation sample consisting of other cases.

For the development of solutions that aim at reducing IRI that can subsequently lead to incomplete orders, this research only focused on corrective actions since this solution direction was requested by Picnic. It is well possible that other solution directions such as additional training (DeHoratius & Ton, 2015), increasing awareness of IRI and its impact (Raman et al., 2001), or adjusting WMS flows can also be used to reduce the occurrence of errors that create IRI. Besides, when looking at the solution of converting some of the developed rules for historical root cause analysis into rules for creating real-time alerts, it was checked whether alerts were created for simulated WMS endpoints belonging to incomplete cases assigned to these root causes. However, with this approach it was not possible to check for false positives, meaning that there is no insight into the creation of alerts for situations in which no IRI was created.

Finally, the applicability of a logistic regression model for predicting the correctness of stock counting activities could not be confirmed by this research. A possible reason for this could be that the correctness of counting activities is, at least in part, influenced by randomness that cannot be explained by predictor variables. This would also explain the lack of academic literature on a model that predicts the correctness of counting activities at the moment they are performed. On the other hand, it is also possible that important predictor variables were not included in the model since they were not known during the research. In addition, it has to be mentioned that it is assumed that the second count that is performed to solve an alert is correct while this assumption might not always be met due to the possibility of making counting errors during the verification process. Moreover, it is worth noting that the model is only trained and evaluated on individual unguided counting activities that resulted in an absolute stock mutation that exceeded a specific threshold. Therefore, it is not yet known whether this model is generalizable to all counting activities and one does not know wat the actual true and false negatives of the current alerting system and the trained model are when looking at all counting activities rather than the evaluated subset. In the end, although the rule of thumb of

Bujang et al. (2018) regarding the minimum sample size was satisfied, it might be possible that the predictive capabilities are enhanced if the number of samples in the training dataset is increased.

7.3. Future research

This research not only offers academic and practical implications but also opens up some areas or concepts that can be further explored in future research. First, it would be interesting to see whether the existing methods and findings are also applicable to Picnic's manual FCs that serve the German and French markets, if this is the case, the results are expected to be somewhat more generalizable. Besides, to get an idea of the full root cause distribution of incomplete orders due to IRI, regardless of whether they belong to incomplete cases due to IRI having at least 5 incomplete CUs, an attempt can be made to further improving the automated root cause analysis such that a more complete root cause distribution is acquired. Elaborating on this, additional analysis can be performed on the incomplete quantities that currently cannot be assigned to a specific root cause such that the knowledge base can be further improved.

For the real-time alerting rules, it is suggested to implement them on the WMS endpoints such that they can be tested in practice for their ability to catch root causes of IRI and the creation of false positives. This would provide better insights into the effectiveness of this solution method which could not be attained during this thesis. If this test proves this solution method to be effective, it can be implemented in the current alerting tool after which its effect on incomplete orders can be analysed. Next to practical relevance, this finding would also add value to scientific literature regarding counting policies.

Lastly, since this thesis found that attempts to correct IRI by performing counting activities often tend to create additional discrepancies, which is also mentioned in the academic literature (e.g. DeHoratius & Raman, 2008; Iglehart & Morey, 1972; Kull et al., 2013; Rinehart, 1960; Woolsey, 1977), the concept of predicting the correctness of stock counting activities can be further explored. This can be done by gathering additional data, testing additional predictor variables, applying different machine-learning methods, or performing a similar analysis in other contexts. For example, it would be interesting to gather data on the accuracy of (a sample of) all counting activities. This can be used to test the performance of the trained model on counting activities that were not included in the evaluated subset. Besides, this data would help to get insights into the actual true and false negatives of the current alerting system and the trained logistic regression model. Finally, if one can devise another model that exhibits high predictive capabilities, it would certainly address a gap in the scientific literature and would be very helpful to any retailer struggling to control IRI.

8. References

Abraham, A. (2005). Rule-based Expert Systems. In P. H. Sydenham & R. Thorn (Eds.), *Handbook of Measuring System Design* (pp. 909–919). John Wiley & Sons.

Amaya Silva, J. R. (2022). Using Deep Learning to Improve Inventory Record Accuracy: Concept and Application. *Academy of Management Proceedings*, *2022*(1). https://doi.org/10.5465/ambpp.2022.305

Barratt, M., Kull, T. J., & Sodero, A. C. (2018). Inventory record inaccuracy dynamics and the role of employees within multi-channel distribution center inventory systems. *Journal of Operations Management*, *63*, 6–24. https://doi.org/10.1016/j.jom.2018.09.003

Begley, S., Hancock, B., Kilroy, T., & Kohli, S. (2020). Retail Practice Automation in retail: An executive overview for getting ready. *McKinsey & Company Retail Insights*.

Bujang, M. A., Sa'At, N., Tg Abu Bakar Sidik, T. M. I., & Lim, C. J. (2018). Sample size guidelines for logistic regression from observational studies with large population: Emphasis on the accuracy between statistics and parameters based on real life clinical data. *Malaysian Journal of Medical Sciences*, *25*(4), 122–130. https://doi.org/10.21315/mjms2018.25.4.12

Cannella, S., Framinan, J. M., Bruccoleri, M., Barbosa-Póvoa, A. P., & Relvas, S. (2015). The effect of inventory record inaccuracy in information exchange supply chains. *European Journal of Operational Research*, *243*(1), 120–129. https://doi.org/10.1016/j.ejor.2014.11.021

Choi, T. Y., Netland, T. H., Sanders, N., Sodhi, M. S., & Wagner, S. (2023). Just-in-time for supply chains in turbulent times. *Production and Operations Management*. https://doi.org/10.1111/poms.13979

Chuang, H. H. C., & Oliva, R. (2015). Inventory record inaccuracy: Causes and labor effects. *Journal of Operations Management*, *39–40*, 63–78. https://doi.org/10.1016/j.jom.2015.07.006

Corsten, D., & Gruen, T. (2003). Desperately seeking shelf availability: An examination of the extent, the causes, and the efforts to address retail out-of-stocks. *International Journal of Retail & Distribution Management*, *31*(12), 605–617. https://doi.org/10.1108/09590550310507731

DeHoratius, N., Mersereau, A. J., & Schrage, L. (2008). Retail inventory management when records are inaccurate. *Manufacturing and Service Operations Management*, *10*(2), 257–277. https://doi.org/10.1287/msom.1070.0203

DeHoratius, N., & Raman, A. (2008). Inventory record inaccuracy: An empirical analysis. *Management Science*, *54*(4), 627–641. https://doi.org/10.1287/mnsc.1070.0789

DeHoratius, N., & Ton, Z. (2015). The role of execution in managing product availability. In *Retail Supply Chain Management: Quantitative Models and Emperical Studies* (pp. 53–77). http://www.springer.com/series/6161

dos Santos Nicolau, A., da S.C Algusto, J. P., do N.A. Pereira, C. M., & Schirru, R. (2017). A Real Time Expert System for Decision Support in Nuclear Power Plants. *International Journal of Nuclear and Quantum Engineering*, *11*(9), 612–618. scholar.waset.org/1307-6892/10007810

Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: A methodology review. *Journal of Biomedical Informatics*, *35*(5–6), 352–359. https://doi.org/10.1016/S1532-0464(03)00034-0

e Oliveira, E., Miguéis, V. L., & Borges, J. L. (2022). Automatic root cause analysis in manufacturing: an overview & conceptualization. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-022-01914-3

Ernst, R., Guerrero, J.-L., & Roshwalb, A. (1993). A Quality Control Approach for Monitoring Inventory Stock Levels. *The Journal of the Operational Research Society*, *44*(11), 1115–1127. https://about.jstor.org/terms

Fawcett, S. E., & Birou, L. M. (1993). Just-In-Time sourcing techniques: Current state of adoption and performance benefits. *Production and Inventory Management Journal*, *34*(1), 18–24. https://www.proquest.com/scholarly-journals/just-time-sourcing-techniques-current-state/docview/199876975/se-2

Gupta, S., & Singhal, R. (2013). Fundamentals and Characteristics of an Expert System. *International Journal on Recent and Innovation Trends in Computing and Communication*, *1*(3), 110–113. http://www.ijritcc.org

Hussein, M., & Zayed, T. (2021). Critical factors for successful implementation of just-in-time concept in modular integrated construction: A systematic review and meta-analysis. *Journal of Cleaner Production*, *284*. https://doi.org/10.1016/j.jclepro.2020.124716

Iglehart, D. L., & Morey, R. C. (1972). Inventory Systems with Imperfect Asset Information. *Management Science*, *18*(8), 388–394. https://www.jstor.org/stable/2629009

Itoo, F., Meenakshi, & Singh, S. (2021). Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection. *International Journal of Information Technology (Singapore)*, *13*(4), 1503–1511. https://doi.org/10.1007/s41870-020-00430-y

Jeong, J., Park, E., Han, W. S., Kim, K., Choung, S., & Chung, I. M. (2017). Identifying outliers of non-Gaussian groundwater state data based on ensemble estimation for long-term trends. *Journal of Hydrology*, *548*, 135–144. https://doi.org/10.1016/j.jhydrol.2017.02.058

Kang, Y., & Gershwin, S. B. (2005). Information inaccuracy in inventory systems: Stock loss and stockout. *IIE Transactions (Institute of Industrial Engineers)*, *37*(9), 843–859. https://doi.org/10.1080/07408170590969861

Kök, A. G., & Shang, K. H. (2007). Inspection and replenishment policies for systems with inventory record inaccuracy. *Manufacturing and Service Operations Management*, *9*(2), 185–205. https://doi.org/10.1287/msom.1060.0136

Kull, T. J., Barratt, M., Sodero, A. C., & Rabinovich, E. (2013). Investigating the effects of daily inventory record inaccuracy in multichannel retailing. *Journal of Business Logistics*, *34*(3), 189–208. https://doi.org/10.1111/jbl.12019

Latino, R. J. (2005). The application of PROACT® RCA to terrorism/counter terrorism related events. *Intelligence and Security Informatics: IEEE International Conference on Intelligence and Security Informatics*, 579–589. https://doi.org/10.1007/11427995_62

Latino, R. J. (2015). How is the effectiveness of root cause analysis measured in healthcare? *Journal of Healthcare Risk Management : The Journal of the American Society for Healthcare Risk Management*, *35*(2), 21–30. https://doi.org/10.1002/jhrm.21198

Latino, R. J. (2022). The application of an effective root cause analysis to any STEM discipline. In *Empowering Women in STEM: Personal Stories and Career Journeys from Around the World* (pp. 3–27). CRC Press. https://doi.org/10.1201/9781003336495-2

Lever, J., Krzywinski, M., & Altman, N. (2016). Classification evaluation: It is important to understand both what a classification metric expresses and what it hides. *Nature Methods*, *13*(8), 603–604. https://link.gale.com/apps/doc/A459507798/HRCA?u=anon~60b72c21&sid=googleScholar&xid=9245e22c

Liiv, I. (2006). Inventory classification enhancement with demand associations. *2006 IEEE International Conference on Service Operations and Logistics, and Informatics*, 18–22. https://doi.org/10.1109/soli.2006.328975

Montgomery, D. C., & Runger, G. C. (2014). *Applied Statistics and Probability for Engineers* (6th ed.). John Wiley & Sons.

NU.nl. (2021). *Bill Gates pompt miljoenen in Nederlandse websuper Picnic*. https://www.nu.nl/economie/6157366/bill-gates-pompt-miljoenen-in-nederlandse-websuper-picnic.html

O'Keefe, R. M., Balci, O., & smith, E. P. (1986). *Validation of expert system performance*. http://hdl.handle.net/10919/19914

O'Keefe, R. M., & O'Leary, D. E. (1993). Expert system verification and validation: a survey and tutorial. *Artificial Intelligence Review*, *7*, 3–42. https://doi.org/10.1007/BF00849196

Okes, D. (2019). *Root cause analysis: The core of problem solving and corrective action* (2nd ed.). ASQ Quality press.

Picnic. (2023). *Over Picnic*. https://picnic.app/nl/over-picnic/

Raman, A., Dehoratius, N., & Ton, Z. (2001). Execution: The Missing Link in Retail Operations. *California Management Review*, *43*(3), 136–152. https://doi.org/10.2307/41166093

Rekik, Y., Syntetos, A. A., & Glock, C. H. (2019). *Inventory Inaccuracy In retaIllng: Does It Matter?* http://ecr-shrink-group.com

Remus, W. (1987). A Study of Graphical and Tabular Displays and Their Interaction with Environmental Complexity. *Source: Management Science*, *33*(9), 1200–1204. https://www.jstor.org/stable/2631885

Rinehart, R. F. (1960). Effects and Causes of Discrepancies in Supply Operations. *Operations Research*, *8*(4), 543–564. https://about.jstor.org/terms

Rossetti, M. D., Collins, T., & Kurgund, R. (2001). Inventory Cycle Counting - A Review. *The Proceedings of the 2001 Industrial Engineering Research Conference*, 457–463.

Shabani, A., Maroti, G., de Leeuw, S., & Dullaert, W. (2021). Inventory record inaccuracy and store-level performance. *International Journal of Production Economics*, *235*. https://doi.org/10.1016/j.ijpe.2021.108111

Sheppard, G. M., & Brown, K. A. (1993). Predicting inventory record-keeping errors with discriminant analysis: A field experiment. *International Journal of Production Economics*, *32*(1), 39–51. https://doi.org/10.1016/0925-5273(93)90006-7

Singh, G., & Singh Ahuja, I. (2013). Strategies and success factors for overcoming challenges in JIT implementation in Indian manufacturing industry. *International Journal of Technology, Policy and Management*, *13*(1), 15–33. https://doi.org/10.1504/IJTPM.2013.051006

Sohal, A. S., Keller, A. Z., & Fouad, R. H. (1989). A Review of Literature Relating to JIT. *International Journal of Operations & Production Management*, *9*(3), 15–25. https://doi.org/10.1108/EUM0000000001228

Sperandei, S. (2014). Understanding logistic regression analysis. *Biochemia Medica*, *24*(1), 12–18. https://doi.org/10.11613/BM.2014.003

Steele, A. L. (2001). Cost drivers and other management issues in the JIT supply chain environment. *Production and Inventory Management Journal*, *42*(2), 61–67. https://www.proquest.com/docview/199881944

Stoltzfus, J. C. (2011). Logistic regression: A brief primer. *Academic Emergency Medicine*, *18*(10), 1099–1104. https://doi.org/10.1111/j.1553-2712.2011.01185.x

van Donselaar, K. H., Gaur, V., van Woensel, T., Broekmeulen, R. A. C. M., & Fransoo, J. C. (2010). Ordering Behavior in Retail Stores and Implications for Automated Replenishment. *Management Science*, *56*(5), 766–784. https://doi.org/10.1287/mnsc.1090.1141

van Donselaar, K. H., van Woensel, T., Broekmeulen, R. A. C. M., & Fransoo, J. C. (2006). Inventory control of perishables in supermarkets. *International Journal of Production Economics*, *104*(2), 462–472. https://doi.org/10.1016/j.ijpe.2004.10.019

Wijffels, L., Giannikas, V., Woodall, P., McFarlane, D., & Lu, W. (2016). An enhanced cycle counting approach utilising historical inventory data. *IFAC-PapersOnLine*, *49*(12), 1347–1352. https://doi.org/10.1016/j.ifacol.2016.07.748

Woolsey, G. (1977). The Fifth Column. The Warehouse Model That Couldn't Be and the Inventory That Couldn't Be Zero. *Interfaces*, *7*(3), 14–17. https://www.jstor.org/stable/25059464

Yang, J., Xie, H., Yu, G., & Liu, M. (2021). Achieving a just–in–time supply chain: The role of supply chain intelligence. *International Journal of Production Economics*, *231*. https://doi.org/10.1016/j.ijpe.2020.107878

A. Appendix: Expert system knowledge base

This appendix contains a high-level overview of the logic that is captured in the rule-based expert system that was used to be able to support the root causes that were hypothesized by the identified logic tree. First of all, it was found that the root cause allocation logic of incomplete orders that is currently in place as described in section 1.2 was sometimes falsely assigning incomplete quantities to the IRI root cause. This is due to the nature of the current root cause allocation logic for assigning incomplete quantities to the IRI root cause which solely looks at negative stock counts close to the moment at which orders went incomplete. It sometimes occurs that a negative stock count is almost immediately followed up by a positive stock count to correct the quantities per freshness date or to register a stock move (which should not be performed via the stock count flow, but it happens) resulting in the absence of a negative delta quantity that can cause orders to go incomplete. Therefore, an additional root cause was added that checks whether the incomplete quantity was indeed caused by IRI and updates the incomplete quantity to be explained by the expert system accordingly.

Besides, not all defined hypotheses were covered by the developed expert system due to the fact that they could not be analysed by solely looking at historical inventory data since these actions are not recorded in the WMS data. This considers the root cause of picking more items than requested and removing items from the FC process for clearing/stock adjustment purposes without registration, these errors are therefore seen as unforeseen discrepancies. To make the expert system more robust against unforeseen discrepancies, margins were included for some of the rules. Moreover, the edge cases that were not specifically covered could be either allocated to a less granular root cause or ended up in the unknown bucket.

In general, the rules were based on a countdown evaluation window, meaning that all countdowns on the day of the incomplete before the shortage solution request was made and the previous day from 12:00 p.m. were evaluated to check whether they could have caused the incomplete orders and what operational error could have caused this incomplete order. This countdown window was selected since earlier countdowns could have been recovered by new inbound or solved by an emergency DC shipment or intra-shipment from another FC. An intra-shipment from another FC means that excess stock of other FCs is gathered and sent to the FC that expects incomplete orders such that these orders can still be fulfilled. However, if the negative discrepancy was found after 12:00 p.m. on the day before the incomplete, it was almost impossible for an FC to recover which subsequently resulted in the incomplete orders.

The hypotheses that were tested and a simplified representation of the logic for supporting a hypothesis is available in an overview represented by Table 5 below. The logic that was captured by the rules is presented as concisely as possible such that the overall way of reasoning can be well understood.

Table 5 - Overview of the expert system knowledge base

RC0	**The incomplete is invalidly assigned to the IRI cause**
	Every countdown during the countdown evaluation window is evaluated for its possibility of resulting in IRI by checking the total counting delta and total negative counting delta in a time window 15 minutes before and 15 minutes after the countdown. If the total delta was negative and larger than 10% of the total negative counting delta in absolute terms, the countdown is accepted to be correcting IRI that could have caused the incompletes. The sum of all accepted negative counts subtracted by the quantity assigned to RC17 (available in this table) and compared to the original incomplete quantity. The minimum of both is saved as the incomplete quantity that will be evaluated for the other root causes (except RC17). The outcome of the original incomplete quantity subtracted by the new incomplete quantity is allocated to the quantity that is invalidly assigned to the IRI cause.
RC1	**Automatically received less stock than registered**

	For all SSCC load carrier lots that are counted down to zero during the evaluation window, it is checked whether the stock is received on this load carrier. If this is the case, and the total stock count delta on this lot before the countdown is lower than 90% of the count to zero quantity, and the count delta on other lots after the receive event of the counted lot is lower than 90% of the count to zero quantity (to check possible unregistered replenishments/temporarily lost stock), the error quantity is increased with the counted down quantity. Finally, the minimum of the incomplete quantity and error quantity is assigned to this root cause.
RC2	**Manually received less stock than registered**
	For all non-SSCC load carrier lots that are counted down to zero during the evaluation window, check whether any manual receive was performed on this lot. If this was the case, check whether any stock count is performed on this lot that could have caused the count down to zero, and check whether any counting delta can explain the countdown (e.g. has a move been made). If the count delta on the current lot is smaller than 90% of the counted-down quantity, and the manually received quantity is larger than 90% of the counted-down quantity, and the count delta on other lots is smaller than 90 of the counted-down quantity, and the counted down quantity is at least larger than the CU per TU size during the replenishment, the counted down quantity is added as the potential error quantity of this root cause. To determine the quantity to be assigned to this root cause, the minimum of the incomplete quantity, potential error quantity, and sum of all received quantities on the lot that is evaluated is calculated and subtracted by RC5, RC6, RC8, RC9, RC10, RC16) and RC20 (since these causes are more likely to happen and might also be caught by this rule).
RC3	**Stock is over-received with solve issue**
	For all pick and buffer locations that are counted down to zero during the evaluation window, the total countdown quantity is summed. Besides, the total over-received quantity via a solve issue action during the 6 days before the evaluation window is summed. If the sum of the over-received quantity via a solve issue is positive, the potential error quantity for this root cause is set to the minimum of the incomplete quantity, the sum of the countdown quantity, and the sum of the over-received quantity via the solve issue event. Thereafter, it has to be checked whether (a part of) this quantity reappeared somewhere after incompletes occurred. Therefore, the maximum is selected of zero and the potential error quantity subtracted by the quantity that is temporarily lost for which the procedure is described in RC20 to determine the quantity that is assigned to this root cause.
RC4	**Stock was temporarily lost due to receiving on pick location instead of buffer**
	For all pick location lots that are counted down to zero during the evaluation window, it is checked whether stock was manually received on this lot. If the manually received quantity is at least equal to the counted down quantity, it is checked whether approximately the opposite (10% margin) was counted up on a buffer location during the 3 days after it was counted down. If this is the case, the counted-down quantity is added to the error quantity for this root cause. After that, the minimum of the incomplete quantity and the error quantity is assigned to this root cause.
RC5	**Replenished stock without registration**
	For all non-pick location lots that are counted down to zero during the evaluation window, it is checked whether approximately the same (at least 80% of the) quantity that is counted down to zero is counted up on other lots with the same freshness date in the period between the first event and the countdown of the lot that is being assessed. If this was the case the counted down quantity is added to the error quantity for this root cause. If multiple lots with the same freshness date experienced a countdown to zero, the error quantity cannot exceed the total quantity that is counted up on other locations with this freshness date. After evaluating all lots, the quantity for this root cause is calculated by taking the minimum of the incomplete quantity and the total error quantity. However, since cases belonging to RC15 are also caught by this rule due to their behaviour, the maximum of zero and the calculated root cause quantity subtracted by the result of RC15 is assigned to this root cause.
RC6	**Stock temporarily lost due to a replenishment error**
	For all lots that are counted down to zero during the evaluation window, it is checked whether this lot experienced a replenishment with an approximately similar quantity (margin of 20%) before the lot was counted down, if this was the case, the quantity that is counted down is added to the error quantity. Besides it is checked whether this stock was temporarily lost by using the logic described in RC20 to come up with its potential error quantity. If both the replenishment and temporarily lost rules are accepted, the counted-down quantity is added to the error quantity of this root cause. However, since this rule is quite abstract, it is first checked whether the quantity is not already explained by any of the other rules (except for RC16 and RC20), if this is the case the maximum of zero and the error quantity subtracted by the result of RC16 (which are is caught by this rule) is assigned to this root cause.
RC7	**Stock permanently lost due to replenishment error**
	For all lots that are counted down to zero during the evaluation window, it is checked whether this lot experienced a replenishment with an approximately similar quantity (margin of 20%) before the lot was counted down, if this was the case, the quantity that is counted down is added to the error quantity. Besides it is checked whether this stock was not temporarily lost by checking whether the logic described in RC20 to come up with the potential error quantity for temporarily lost stock. If both the replenishment rule is accepted and no temporarily lost stock is found, the counted down quantity is added to the error quantity of this root cause. However, since

	this rule is quite abstract, it is first checked whether the quantity is not already explained by any of the other rules (except for RC16 and RC20), if this is the case the maximum of zero and the error quantity subtracted by the result of RC16 (which are is caught by this rule) is assigned to this root cause.
RC8	**Counting error on the pick location**
	The pick location is perceived as one lot since it is a single location and freshness dates are juggled around because of errors in freshness date management. If the total count delta during the evaluation window on the pick location is negative and the total count delta on the pick location before the evaluation window is approximately the opposite (at least 70%), the minimum of the incomplete quantity and the negative counting delta during the evaluation window subtracted by the result of RC16 is allocated to this root cause.
RC9	**Counting error on a buffer location**
	For all buffer lots that experienced a countdown during the evaluation window, it is checked if the total counting delta during this evaluation window was negative for this lot. If this was the case, it is checked whether the total counting delta on this lot before the evaluation window is positive approximately the opposite (at least larger than 70%) of the total counting delta during the evaluation window. If this condition is satisfied, the total quantity of stock that is counted up on another lot after the evaluation window is calculated to check whether the stock was maybe temporarily lost (due to an error in another process). The counting error on the buffer lot that is evaluated is therefore set to the maximum of zero and the delta during the evaluation period (multiplied by -1 to change the sign) subtracted by the temporarily lost stock. The quantity that is assigned to this root cause is the minimum of the incomplete quantity and the maximum of zero and the sum of all counting errors per lot subtracted by RC13 and RC14 which can both be more granular causes of RC9.
RC10	**Counting error on a load carrier**
	For all load carrier (incl. roll cage) lots that experienced a countdown during the evaluation window, it is checked if the total counting delta during this evaluation window was negative. If this was the case, it is checked whether the total counting delta on this lot before the evaluation window is positive approximately the opposite (at least larger than 70%) of the total counting delta during the evaluation window. If this condition is satisfied, the total quantity of stock that is counted up on another lot after the evaluation window is calculated to check whether the stock was maybe temporarily lost (due to an error in another process). The counting error on the load carrier/roll cage lot that is evaluated is therefore set to the maximum of zero and the delta during the evaluation period (multiplied by -1 to change the sign) subtracted by the temporarily lost stock. The quantity that is assigned to this root cause is the minimum of the incomplete quantity and the maximum of zero and the sum of all counting errors per lot subtracted by RC13 and RC14 which can both be more granular causes of RC10.
RC11	**Counting error on other**
	For all lots other than a pick/buffer/load carrier/roll cage that experienced a countdown during the evaluation window, it is checked if the total counting delta during this evaluation window was negative. If this was the case, it is checked whether the total counting delta on this lot before the evaluation window is positive approximately the opposite (at least larger than 70%) of the total counting delta during the evaluation window. If this condition is satisfied, the total quantity of stock that is counted up on another lot after the evaluation window is calculated to check whether the stock was maybe temporarily lost (due to an error in another process). The counting error on the lot that is evaluated is therefore set to the maximum of zero and the delta during the evaluation period (multiplied by -1 to change the sign) subtracted by the temporarily lost stock. The quantity that is assigned to this root cause is the minimum of the incomplete quantity and the sum of all counting errors per lot.
RC12	**CU per TU error with replenishment**
	For all pick location lots that are counted down to zero during the evaluation window, all replenishments and stock moves to this lot are checked on whether it is replenished from a lot with a CU per TU size that is different from any of the received CU per TU sizes on the lots other than the lot it was replenished from. These replenishments/moves are flagged as suspicious. After that, the quantities of all suspicious replenishments/moves are summed for that lot, and it is checked whether this quantity is at least the quantity that is counted down on the pick location lot. If this is the case the countdown that is being assessed is added to the error quantity for this root cause. After that, the minimum of the incomplete quantity and the error quantity is assigned to this root cause.
RC13	**CU per TU error without replenishment**
	For all buffer and load carrier lots that are counted down to zero during the evaluation window, it is checked whether the used CU per TU size is different from any received CU per TU size on the lots other than the lot that is counted down. If this is the case, the counted-down quantity is added to the error quantity belonging to this root cause. Thereafter, the minimum of the incomplete quantity and the error quantity is assigned to this root cause.
RC14	**Unit of measure error during count**
	For all lots that consisted of a TU that are counted down to zero during the evaluation window, it is checked whether this lot experienced a count up where the registered CU quantity after the count was exactly the CU quantity before the count multiplied by the CU per TU size that is applicable for this lot. If this is the case, the count down to zero quantity is added to the error quantity for this root cause. After that, the minimum of the incomplete quantity and the error quantity is assigned to this root cause.

RC15	**Wrong SSCC label used**
	For all SSCC load carriers that are counted down to zero during the evaluation window, it is checked whether exactly the same stock (quantity and freshness date) is counted up on another (new) SSCC licence plate in between the automatic receive and countdown events of the load carrier and at most 24 hours after the stock was automatically received. If this is the case, the counted-down quantity is added to the error quantity belonging to this root cause. After that, the minimum of the incomplete quantity and error quantity is assigned to this root cause.
RC16	**Invalid shortage called that resulted in count down to zero**
	If a shortage that resulted in a stock mutation was called on the day of the incompletes before the RSS was activated, it is checked whether any count-up smaller or equal to the counted-down quantity (+2 CUs margin) was performed on the day of the incompletes after the RSS was activated. If this was the case, the quantity that is assigned to this root cause is the minimum of the incomplete quantity, the countdown quantity due to the shortage, and the maximum count-up quantity that fulfils the abovementioned criteria.
RC17	**Freshness date management**
	Clearing events on the pick location during the countdown evaluation window are checked on any correction of freshness dates via stock counts beforehand. If freshness dates were corrected by lots being counted down with a total negative delta that is approximately the same (20% margin) as the total cleared quantity and lots are counted up with a total positive delta that is approximately the same (20% margin) as the total cleared quantity, the total cleared quantity could have caused the incompletes. The minimum of the original incomplete quantity and incomplete quantity is assigned to this root cause
RC18	**Messy roll cage operations**
	All counted down to zero quantities on a roll cage during the evaluation window are summed. If this sum is positive, the minimum of the incomplete quantity and this sum is saved to this root cause.
RC19	**Load carrier check error/not performed**
	For all SSCC load carrier lots that are counted down to zero during the evaluation window. If the last stock on this load carrier was untouched for at least 24 hours, the counted down to zero quantity is added to the potential error quantity of this root cause. Thereafter, the quantity assigned to this root cause is determined by taking the minimum of the incomplete quantity and the total potential error quantity.
RC20	**Stock was temporarily lost due to an unknown root cause**
	For all stock that is counted down to zero during the evaluation window, it is checked if the stock reappeared somewhere during the 3 days after the RSS timestamp. This is done on two levels, by looking at whether the same freshness date was counted up with (at least 80% of) the counted down quantity on this freshness. The second option relaxes the freshness date and checks whether the total counting delta was (at least 80%) the opposite of the counted-down quantity. The maximum of both temporarily lost rules is added to the potential error quantity of this root cause. However, this potential error quantity can also consist of quantities that are already explained by other more granular causes that focus on temporarily lost stock (RC4, RC6, RC16), these quantities need to be subtracted from the potential error quantity to come up with this root cause.
UNKN	**The quantity that could not be assigned by the expert system**
	The incomplete quantity is subtracted by the minimum of the incomplete quantity and the sum of RC0 until RC20.

B. Appendix: Example of code for real-time alerting

This appendix contains examples of code with comments containing a concise explanation that can be used to create real-time alerts for situations in which it is expected that a root cause of IRI that can potentially result in incomplete orders is made.

- Stock was replenished without registration/an SSCC error was made:

```
### Initialize parameters and prepare required data structure ###
margin = 0.2            # Margin for replenishment without registration error
max_timedelta = 24      # Maximum timedelta for SSCC error after
alerts = []             # List with alert messages
alerts_BE = {}          # Alerting backend to save alert data
SSCC_dict = {}          # Dictionary that keeps track of all SSCC license plates

### Get the starting inventory of the lots (WMS endpoints) ###
# Data is a dataframe containing all events of an article on FC level sorted on event
timestamp (ASC)
starting_inventory = data.loc[data.groupby('MUTATION_LOT_ID')['EVENT_TS'].idxmin(),
                              ['MUTATION_LOT_ID','PRE_CU_QTY']]
inventory_overview = starting_inventory.set_index('MUTATION_LOT_ID')['PRE_CU_QTY'].to_dict()

### Simulation ###
for _, event in data.iterrows():
    # Update simulated WMS endpoints
    if event['MUTATION_LOT_ID'] in inventory_overview and not np.isnan(event['DIFF_CU_QTY']):
        mutation_lot_id = event['MUTATION_LOT_ID']
        diff_cu_qty = event['DIFF_CU_QTY']
        inventory_overview[mutation_lot_id] += diff_cu_qty
    # Add received SSCCs to SSCC_dict with relevant data
    if event['EVENT_ACTION'] in ['RECEIVE'] and event['LOCATION_TYPE'] == 'SSCC':
        SSCC_dict[event['MUTATION_LOT_ID']] = {'receive_ts':event['EVENT_TS'], 'time_delta':0,
                                               'active': True, 'freshness': event['MUTATION_BBD_OR_PD']}
    # Update the time since the SSCCs were received
    if len(SSCC_dict) > 0:
        for SSCC in SSCC_dict:
            SSCC_dict[SSCC]['time_delta'] = event['EVENT_TS'] - SSCC_dict[SSCC]['receive_ts']
    # If a countup is performed
    if event['EVENT_ACTION'] in ['STOCK_COUNT', 'LOCATION_COUNT', 'CREATE_FRESHNESS_DATE'] and
       event['DIFF CU QTY'] > 0:
        countup_qty = event['DIFF_CU_QTY']
        freshness_date = event['MUTATION_BBD_OR_PD']
        # Check for all other lots
        for lot in inventory_overview:
            if lot != event['MUTATION LOT ID']:
                inventory_level = inventory_overview[lot]
                # If an SSCC error was made
                if event['LOCATION_TYPE'] == 'SSCC' and lot in SSCC_dict and
                   event['DIFF_CU_QTY'] == inventory_level and event['MUTATION_BBD_OR_PD'] ==
                   SSCC_dict[lot]['freshness'] and SSCC_dict[lot]['time_delta'] <
                   pd.Timedelta(hours=max timedelta) and SSCC dict[lot]['active']:
                    timestamp = event['EVENT_TS']
                    # Create alert for verifying the old SSCC label
                    alerts_BE[lot] = {'ts': timestamp, 'qty': inventory_level, 'active': True,
                                      'type': 'Used wrong SSCC label'}
                    alert = f"ALERT!!!! - {timestamp} - Stock ({inventory_level} CUs) expected
                            to be duplicated by scanning wrong SSCC - CHECK location {lot} to
                            confirm"
                    alerts.append(alert)
                # Else check if the stock was replaced
                elif inventory_level >= countup_qty * (1 - 0.2) and inventory_level <=
                   countup_qty * (1 + margin):
                    timestamp = event['EVENT_TS']
                    # Create alert to verify whether stock on the location is indeed already
                    replenished
                    alerts_BE[lot] = {'ts': timestamp, 'qty': inventory_level, 'active': True,
                                      'type': 'Replenishment without registration'}
                    alert = f"ALERT!!!! - {timestamp} - Stock ({inventory_level} CUs) expected
                            to be replenished without registration - CHECK location {lot} to
                            confirm"
                    alerts.append(alert)
    # If the stock of a lot is fully depleted
    if event['POST_CU_QTY'] == 0:
        # If the lot was a SSCC, deactivate the SSCC such that it cannot cause new alerts
```

```
        if event['MUTATION_LOT_ID'] in SSCC_dict:
            SSCC_dict[event['MUTATION_LOT_ID']]['active'] = False
        # If an alert was created for this lot, automatically solve and deactivate the alert
        if event['MUTATION LOT ID'] in alerts BE:
            alert_type = alerts_BE[event['MUTATION_LOT_ID']]['type']
            alerts_BE[event['MUTATION_LOT_ID']]['active'] = False
            alert = f"SOLVED!!!! - {event['EVENT_TS']} - {alert_type} error on
                    {event['MUTATION_LOT_ID']}"
            alerts.append(alert)
    # Show incomplete orders in the alerting for simulation process (can be removed when
    implemented in practice)
    if event['EVENT_ACTION'] == 'INCOMPLETE_ORDER':
        incomplete_ts = event['EVENT_TS']
        incomplete_qty = event['ENTERED_CU_QTY']
        alerts.append(f"Incomplete order occurred on {incomplete_ts} with {incomplete_qty}
                      CUs")
    # Show WMS endpoints and created alerts during simulation (can be removed when implemented
    in practice)
    display(inventory_overview,alerts)
    clear_output(wait=True)
    time.sleep(0.01)
#Show final WMS endpoints and created/automatically solved alerts
display(inventory_overview)
for alert in alerts:
    print(alert)
```

- Unit of measure error

```
### Prepare required data structure ###
alerts = []             # List with alert messages
alerts_BE = {}          # Alerting backend to save alert data

### Get the starting inventory of the lots (WMS endpoints) ###
# Data is a dataframe containing all events of an article on FC level sorted on event
timestamp (ASC)
starting_inventory = data.loc[data.groupby('MUTATION_LOT_ID')['EVENT_TS'].idxmin(),
                              ['MUTATION_LOT_ID','PRE_CU_QTY']]
inventory_overview = starting_inventory.set_index('MUTATION_LOT_ID')['PRE_CU_QTY'].to_dict()

### Simulation ###
for _, event in data.iterrows():
    # Update simulated WMS endpoints
    if event['MUTATION_LOT_ID'] in inventory_overview and not np.isnan(event['DIFF_CU_QTY']):
        mutation_lot_id = event['MUTATION_LOT_ID']
        diff_cu_qty = event['DIFF_CU_QTY']
        inventory_overview[mutation_lot_id] += diff_cu_qty
    # If a countup is performed
    if event['EVENT_ACTION'] in ['STOCK_COUNT', 'LOCATION_COUNT'] and
       event['DIFF_CU_QTY'] > 0:
        # Check if unit of measure error is made
        if event['POST_CU_QTY'] == event['PRE_CU_QTY'] * event['CU_PER_TU']:
            timestamp = event['EVENT_TS']
            lot = event['MUTATION_LOT_ID']
            countup_qty = event['DIFF_CU_QTY']
            cu_per_tu_qty = event['CU_PER_TU']
            # Create alert for verifying whether unit of measure error was made
            alerts_BE[lot] = {'ts': timestamp, 'qty':countup_qty, 'cu_per_tu': cu_per_tu_qty,
                              'active': True}
            alert = f"ALERT!!!! - {timestamp} - Stock ({countup_qty} CUs) expected to be
                    counted up with unit of measure error ({cu per tu qty} CU/TU) - CHECK
                    location {lot} to confirm"
            alerts.append(alert)
    # If a countown is performed
    if event['EVENT_ACTION'] in ['STOCK_COUNT', 'LOCATION_COUNT', 'LOAD_CARRIER_EMPTY'] and
       event['DIFF_CU_QTY'] < 0:
        # If an alert is active for this lot
        if event['MUTATION_LOT_ID'] in alerts_BE:
            # If the unit of measure error is corrected before checking the alert,
            automatically solve and deactivate the alert
            if event['POST_CU_QTY'] == event['PRE_CU_QTY'] / event['CU_PER_TU']:
                alerts BE[event['MUTATION LOT ID']]['active'] = False
                alert = f"SOLVED!!!! - {event['EVENT_TS']} - UoM error on
                        {event['MUTATION_LOT_ID']}"
                alerts.append(alert)
```

```
            # If the stock on the lot is fully depleted, automatically solve and deactivate
            the alert
            elif event['POST_CU_QTY'] == 0:
                alerts_BE[event['MUTATION_LOT_ID']]['active'] = False
                alert = f"SOLVED!!!! - {event['EVENT_TS']} - UoM error on
                        {event['MUTATION_LOT_ID']}"
                alerts.append(alert)
    # Show incomplete orders in the alerting for simulation process (can be removed when
    implemented in practice)
    if event['EVENT_ACTION'] == 'INCOMPLETE_ORDER':
        incomplete_ts = event['EVENT_TS']
        incomplete_qty = event['ENTERED_CU_QTY']
        alerts.append(f"Incomplete order occurred on {incomplete_ts} with {incomplete_qty}
                      CUs")
    # Show WMS endpoints and created alerts during simulation (can be removed when implemented
    in practice)
    display(inventory_overview,alerts)
    clear_output(wait=True)
    time.sleep(0.01)
#Show final WMS endpoints and created/automatically solved alerts
display(inventory_overview)
for alert in alerts:
    print(alert)
```

- CU per TU error with/without registration

```
### Initialize parameters and prepare required data structure ###
ids = {}                # Dictionary for saving CU/TU size of trace IDs
CUpTU_sizes = [1.0]     # List for received TU/CU sizes (1.0 to initialize CU)
alerts = []             # List with alert messages
alerts_BE = {}          # Alerting backend to save alert data
trace_id = False        # Variable for saving trace ID of current replenishment/move/putaway

### Get the starting inventory of the lots (WMS endpoints) ###
# Data is a dataframe containing all events of an article on FC level sorted on event
timestamp (ASC)
starting_inventory = data.loc[data.groupby('MUTATION_LOT_ID')['EVENT_TS'].idxmin(),
                              ['MUTATION_LOT_ID','PRE_CU_QTY']]
inventory_overview = starting_inventory.set_index('MUTATION_LOT_ID')['PRE_CU_QTY'].to_dict()

### Simulation ###
for _, event in data.iterrows():
    # Update simulated WMS endpoints
    if event['MUTATION_LOT_ID'] in inventory_overview and not np.isnan(event['DIFF_CU_QTY']):
        mutation_lot_id = event['MUTATION_LOT_ID']
        diff cu qty = event['DIFF CU QTY']
        inventory_overview[mutation_lot_id] += diff_cu_qty
    # Add received CU/TU sizes to list
    if event['EVENT_ACTION'] == 'RECEIVE':
        CUpTU_sizes.append(event['CU_PER_TU'])
    # If a countup is performed
    if event['EVENT_ACTION'] in ['STOCK COUNT', 'LOCATION COUNT', 'CREATE FRESHNESS DATE'] and
       event['DIFF_CU_QTY'] > 0:
        #Check if the used CU/TU size was suspicious since it was not received
        if event['CU_PER_TU'] not in CUpTU_sizes:
            timestamp = event['EVENT_TS']
            lot = event['MUTATION_LOT_ID']
            countup_qty = event['DIFF_CU_QTY']
            cu_per_tu_qty = event['CU_PER_TU']
            # Create alert for checking whether CU/TU error was made on counted lot
            alerts_BE[lot] = {'ts': timestamp, 'qty':countup_qty, 'cu_per_tu': cu_per_tu_qty,
                              'active': True}
            alert = f"ALERT!!!! - {timestamp} - Stock ({countup_qty} CUs) expected to be
                    counted up with wrong CU/TU size ({cu_per_tu_qty} CU/TU) - CHECK location
                    {lot} to confirm"
            alerts.append(alert)
    # If a replenishment is made from a location for which a CU/TU error was expected to be
    made
    if event['EVENT_ACTION'] in ['REPLENISH_STOCK', 'MOVE_STOCK', 'PUT_AWAY'] and
       event['MUTATION_LOT_ID'] in alerts_BE and event['DIFF_CU_QTY'] < 0:
        # Save trace id and CU/TU size of negative stock mutation (which always happens before
        positive stock mutation on new location)
        trace_id = event['M_ID']
        ids[trace_id] = event['CU_PER_TU']
    # Create alert for checking whether stock was replenished with wrong CU/TU on new lot
    if event['M_ID'] == trace_id and event['DIFF_CU_QTY'] > 0:
```

```
            timestamp = event['EVENT_TS']
            lot = event['MUTATION_LOT_ID']
            moved_qty = event['DIFF_CU_QTY']
            cu per tu qty = ids[trace id]
            alerts_BE[lot] = {'ts': timestamp, 'qty':countup_qty, 'cu_per_tu': cu_per_tu_qty,
                              'active': True}
            alert = f"ALERT!!!! - {timestamp} - Stock ({moved_qty} CUs) expected to be moved with
                    wrong CU/TU size ({cu_per_tu_qty} CU/TU) - CHECK location {lot} to confirm"
            alerts.append(alert)
            trace id = False # Clear trace ID
        # If the stock on a lot with an CU/TU alert is fully depleted, automatically solve and
        deactivate the alert
        if event['EVENT_ACTION'] in ['STOCK_COUNT', 'LOCATION_COUNT', 'LOAD_CARRIER_EMPTY',
           'REPLENISH_STOCK', 'MOVE_STOCK', 'PUT_AWAY'] and event['DIFF_CU_QTY'] < 0:
            if event['MUTATION_LOT_ID'] in alerts_BE:
                if event['POST CU QTY'] == 0:
                    alerts_BE[event['MUTATION_LOT_ID']]['active'] = False
                    alert = f"SOLVED!!!! - {event['EVENT_TS']} - CU/TU error on
                            {event['MUTATION_LOT_ID']}"
                    alerts.append(alert)
        # Show incomplete orders in the alerting for simulation process (can be removed when
        implemented in practice)
        if event['EVENT_ACTION'] == 'INCOMPLETE_ORDER':
            incomplete_ts = event['EVENT_TS']
            incomplete_qty = event['ENTERED_CU_QTY']
            alerts.append(f"Incomplete order occurred on {incomplete_ts} with {incomplete_qty}
            CUs")
        # Show WMS endpoints and created alerts during simulation (can be removed when implemented
        in practice)
        display(inventory_overview,alerts)
        clear_output(wait=True)
        time.sleep(0.01)
    #Show final WMS endpoints and created/automatically solved alerts
    display(inventory_overview)
    for alert in alerts:
        print(alert)
```

- Messy roll cage operations

```
### Initialize parameters and prepare required data structure ###
time_trigger = 8        # Hours after which an alert is triggered
alerts = []             # List with alert messages
alerts_BE = {}          # Alerting backend to save alert data
RC_dict = {}            # Dictionary that keeps track of all roll cages

### Get the starting inventory of the lots (WMS endpoints) ###
# Data is a dataframe containing all events of an article on FC level sorted on event
timestamp (ASC)
starting_inventory = data.loc[data.groupby('MUTATION_LOT_ID')['EVENT_TS'].idxmin(),
                              ['MUTATION_LOT_ID','PRE_CU_QTY']]
inventory_overview = starting_inventory.set_index('MUTATION_LOT_ID')['PRE_CU_QTY'].to_dict()

### Simulation ###
for _, event in data.iterrows():
    # Update simulated WMS endpoints
    if event['MUTATION_LOT_ID'] in inventory_overview and not np.isnan(event['DIFF_CU_QTY']):
        mutation lot id = event['MUTATION LOT ID']
        diff_cu_qty = event['DIFF_CU_QTY']
        inventory_overview[mutation_lot_id] += diff_cu_qty
    # Add created roll cage lot to RC_dict with relevant data
    if event['LOCATION_TYPE'] == 'RC':
        if event['MUTATION_LOT_ID'] not in RC_dict:
            RC_dict[event['MUTATION_LOT_ID']] = {'receive_ts': event['EVENT_TS'],
                                                 'time_delta': 0, 'active': True,
                                                 'alert': False}
    # Update the time since RC lot was created + alert check
    if len(RC_dict) > 0:
        for RC in RC_dict:
            RC_dict[RC]['time_delta'] = event['EVENT_TS'] - RC_dict[RC]['receive_ts']
            # Check if an active RC lot exists for longer than the time trigger
            if RC_dict[RC]['time_delta'] > pd.Timedelta(hours=time_trigger) and
               RC_dict[RC]['active'] == True and RC_dict[RC]['alert'] == False:
                    timestamp = event['EVENT_TS']
                    qty = inventory_overview[RC]
                    # Create alert for verifying whether (stock on) roll cage still exists
                    alerts_BE[RC] = {'ts': timestamp, 'qty':qty,  'active': True}
```

```
                    alert = f"ALERT!!!! - {timestamp} - Stock ({qty} CUs) on roll cage for
                            more than {time_trigger} hours - CHECK location {RC} to confirm"
                    alerts.append(alert)
                    RC_dict[RC]['alert'] = True
    # If stock on roll cage is fully depleted
    if event['POST_CU_QTY'] == 0 and event['MUTATION_LOT_ID'] in RC_dict:
        # Deactivate roll cage
        RC_dict[event['MUTATION_LOT_ID']]['active'] = False
        # Automatically solve and deactivate any messy roll cage alert for this roll cage
        if event['MUTATION LOT ID'] in alerts_BE:
            alerts_BE[event['MUTATION_LOT_ID']]['active'] = False
            alert = f"SOLVED!!!! - {event['EVENT_TS']} - Stock on roll cage for more than
                    {time_trigger} hours - {event['MUTATION_LOT_ID']}"
            alerts.append(alert)
    # Show incomplete orders in the alerting for simulation process (can be removed when
    implemented in practice)
    if event['EVENT_ACTION'] == 'INCOMPLETE_ORDER':
        incomplete_ts = event['EVENT_TS']
        incomplete_qty = event['ENTERED_CU_QTY']
        alerts.append(f"Incomplete order occurred on {incomplete_ts} with {incomplete_qty}
                      CUs")
    # Show WMS endpoints and created alerts during simulation (can be removed when implemented
    in practice)
    display(inventory_overview,alerts)
    clear_output(wait=True)
    time.sleep(0.01)
#Show final WMS endpoints and created/automatically solved alerts
display(inventory_overview)
for alert in alerts:
    print(alert)
```

- Stock on load carrier for longer than 24 hours

```
### Initialize parameters and prepare required data structure ###
time_trigger = 24       # Hours after which an alert is triggered
alerts = []             # List with alert messages
alerts_BE = {}          # Alerting backend to save alert data
SSCC_dict = {}          # Dictionary that keeps track of all SSCC license plates

### Get the starting inventory of the lots (WMS endpoints) ###
# Data is a dataframe containing all events of an article on FC level sorted on event
timestamp (ASC)
starting_inventory = data.loc[data.groupby('MUTATION_LOT_ID')['EVENT_TS'].idxmin(),
                              ['MUTATION_LOT_ID','PRE_CU_QTY']]
inventory_overview = starting_inventory.set_index('MUTATION_LOT_ID')['PRE_CU_QTY'].to_dict()

### Simulation ###
for _, event in data.iterrows():
    # Update simulated WMS endpoints
    if event['MUTATION_LOT_ID'] in inventory_overview and not np.isnan(event['DIFF_CU_QTY']):
        mutation_lot_id = event['MUTATION_LOT_ID']
        diff_cu_qty = event['DIFF_CU_QTY']
        inventory_overview[mutation_lot_id] += diff_cu_qty
    # Add received SSCCs to SSCC_dict with relevant data
    if event['EVENT_ACTION'] in ['RECEIVE'] and event['LOCATION_TYPE'] == 'SSCC':
        SSCC_dict[event['MUTATION_LOT_ID']] = {'receive_ts':event['EVENT_TS'],
                                               'time delta': 0, 'active': True,
                                               'alert': False}
    # Update the time since SSCC was received + alert check
    if len(SSCC_dict) > 0:
        for SSCC in SSCC_dict:
            SSCC_dict[SSCC]['time_delta'] = event['EVENT_TS'] - SSCC_dict[SSCC]['receive_ts']
            # Check if an active SSCC exists for longer than the time trigger
            if SSCC_dict[SSCC]['time_delta'] > pd.Timedelta(hours=time_trigger) and
               SSCC_dict[SSCC]['active'] == True and SSCC_dict[SSCC]['alert'] == False:
                    timestamp = event['EVENT_TS']
                    qty = inventory_overview[SSCC]
                    # Create alert for verifying whether (stock on) load carrier still exists
                    alerts_BE[SSCC] = {'ts': timestamp, 'qty':qty,  'active': True}
                    alert = f"ALERT!!!! - {timestamp} - Stock ({qty} CUs) on Load Carrier for
                            more than {time_trigger} hours - CHECK location {SSCC} to confirm"
                    alerts.append(alert)
                    SSCC_dict[SSCC]['alert'] = True
    # If stock on load carrier is fully depleted
    if event['POST_CU_QTY'] == 0 and event['MUTATION_LOT_ID'] in SSCC_dict:
        # Deactivate load carrier
```

```
            SSCC_dict[event['MUTATION_LOT_ID']]['active'] = False
            # Automatically solve and deactivate any load carrier check alert for this load
            carrier
            if event['MUTATION LOT ID'] in alerts BE:
                alerts_BE[event['MUTATION_LOT_ID']]['active'] = False
                alert = f"SOLVED!!!! - {event['EVENT_TS']} - Stock on Load Carrier for more than
                        {time_trigger} hours - {event['MUTATION_LOT_ID']}"
                alerts.append(alert)
        # Show incomplete orders in the alerting for simulation process (can be removed when
        implemented in practice)
        if event['EVENT_ACTION'] == 'INCOMPLETE_ORDER':
            incomplete_ts = event['EVENT_TS']
            incomplete_qty = event['ENTERED_CU_QTY']
            alerts.append(f"Incomplete order occurred on {incomplete_ts} with {incomplete_qty}
                          CUs")
        # Show WMS endpoints and created alerts during simulation (can be removed when implemented
        in practice)
        display(inventory_overview,alerts)
        clear_output(wait=True)
        time.sleep(0.01)
    #Show final WMS endpoints and created/automatically solved alerts
    display(inventory_overview)
    for alert in alerts:
        print(alert)
```