Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 253 (2025) 3088–3097

Procedia Computer Science

www.elsevier.com/locate/procedia

6th International Conference on Industry 4.0 and Smart Manufacturing

Enhancing MRP Adoption in SMEs: A Python-Based Algorithmic Approach

Vitti M.[a, *], Coletta R.[a], Reis J.[b], Pinto F. S.[b], Facchini F.[a]

[a] *Polytechnic University of Bari, Via E. Orabona, 4, Bari – 70126, Italy*

[b] *Industrial Engineering and Management, Faculty of Engineering, Lusófona University and RCM2+, Campo Grande, 1749-024 Lisbon, Portugal*

Abstract

Implementing Enterprise Resource Planning (ERP) and Material Requirements Planning (MRP) systems enables companies to gain a competitive advantage in current dynamic and uncertain markets. Although implementing these systems offers several benefits, it encounters major barriers, especially in Small and Medium Enterprises (SMEs). They constitute a significant share of the industrial sector globally but often face challenges such as limited access to finance, technology, and markets, which can hinder their growth and sustainability. Addressing these barriers is vital for leveraging the full potential of SMEs and fostering a more dynamic and resilient global economy. In this context, adopting Python-based solutions could be a promising solution. Python's open-source nature indeed empowers users with unrestricted application development, distribution, and commercialization. This flexibility fosters platform-independent development and eliminates vendor lock-in. Furthermore, its clear syntax, powerful interpreter, and extensibility enable efficient and versatile programming across various paradigms, making it a very useful language for resource-limited contexts such as SMEs. To this concern, the objective of the present work was to investigate the benefits offered by implementing a Python-based algorithm to facilitate the adoption of an MRP system in SMEs. To this end, a Python algorithm has been developed based on a sequence of four logical steps: exploding, netting, lot sizing, and offsetting, and was numerically applied. The results obtained provided a clear perspective on the usefulness of Python and the benefits derived from its use due to its ease of use, flexibility, and reduced costs.

Keywords: Operations Management, Digitalization, Material Requirements Planning, Python algorithm, Production plan, Bill of Materials

* Corresponding author.
E-mail address: micaela.vitti@poliba.it

1877-0509
10.1016/j.procs.2025.02.033

1. Introduction

In the last decades of the 20th century, Material Requirements Planning (MRP) systems emerged as a significant approach for managing the procurement of raw materials and components within production plants [1]. MRP systems focus mainly on materials requirements planning, production management and stock control and, their primary objective is to assist companies in identifying the optimal quantity of raw materials, components and semi-finished goods needed for the production process [2]. These elements are essential for gaining a competitive advantage, as they lead to greater customer satisfaction and greater adaptability to market demands; therefore, the correct implementation and strategic use of these tools are essential to ensure the success and durability of companies in the current economic environment [3]. Adopting an MRP system has numerous benefits, including improved response to customer orders through improved schedules, rapid response to market changes, improved utilization of facilities and labor, and reduced inventory levels. By implementing this technique in its system, a company can better respond to customer demand, have greater efficiency in controlling production, avoid excess inventory, and have less waste [4].

In the late 1970s, although MRP as a production control system was widespread in many manufacturing industries, it did not achieve the same results as early adopters because it was only intended for materials planning in a deterministic environment [5], [6]. Much research has been conducted on the topic, and modifications to the MRP system have been proposed, giving rise to closed-loop MRP (or MRP II). In this system, production planning and capacity requirements are also considered during material requirement planning. However, the new system could not achieve the desired results due to the lack of computing power to handle the various factors affecting the MRP [7].

In the 1990s, the complexity of managing information systems favored the emergence of the so-called Enterprise Resource Planning (ERP) paradigm, leading to the development of integrated information systems to support various operational processes within a company. ERP systems are sets of integrated software modules (suites) that manage all relevant company information and the transactions that operate on it using a (logically) single, centralized database. Such systems enable the coordinated execution of operational processes within the company by supporting the required functions and support the planning and control of all company resources (human, systemic, financial, material) required for the execution of processes, integrating the operational and administrative cycles of companies. Each module characterizing the ERP controls the processes of specific business functions (such as Administration and Finance, Logistics, Production, Human Resources, etc.), while the core of the software takes the information from the central database and presents it to the user via its interface [8], [9]. ERP and MRP work together to optimize business operations. The MRP, as a subsystem of the ERP, provides material and resource data to the ERP, which integrates and utilizes it in other business areas. The main difference between ERP and MRP is that ERP systems help plan and automate a variety of back-office business functions, while MRP systems focus on materials management [10], [11]. The impact of ERP on the supply chain is highly significant: according to the American Production and Inventory Control Society, sales of MRP software and implementation support exceeded $1 billion in the United States in 1989. Moreover, worldwide sales of ERP packages, along with implementation support, have surpassed $15 billion and have experienced an annual growth rate of 30% [12]. In a study conducted by the Panorama Consulting Group, it was found that a growing percentage of organizations are implementing ERP across multiple offices, warehouses, and sites. From the same analysis, it emerged that the main players in the market are SAP, Oracle, Microsoft, Epicor, and Infor. These software vendors alone account for 67% of the market. SAP continues to be the vendor most frequently selected by organizations (27%), followed by Oracle (19%) and Microsoft Dynamics (13%) [13].

Despite the obtainable benefits, implementing ERP systems encounters major barriers, especially regarding Small and Medium Enterprises (SMEs), which form a key part of the industry sector globally. SMEs account for approximately 90% of businesses and more than 50% of employment worldwide, underscoring their pivotal role in economic development [14]. SMEs are also critical in fostering economic inclusivity, as they provide opportunities for diverse populations, including women and young entrepreneurs, thereby promoting social stability, and reducing inequality [15]. In many developing countries, SMEs contribute substantially to GDP and are essential for industrial diversification and resilience against economic shocks [16]. Moreover, they help integrate local economies into the global supply chain, enhancing trade and investment flows [17]. Despite their significance, SMEs often face challenges such as limited access to finance, technology, and markets, which can hinder their growth and sustainability [18]. These general barriers translate, in the operational context, into difficulty in implementing MRP and ERP systems in SMEs. One primary barrier is the high initial cost of implementation, which includes software acquisition,

customization, and employee training. These costs can be prohibitive for SMEs with limited financial resources [19]. Furthermore, the complex nature of ERP and MRP systems necessitates a significant investment of time and specialized knowledge. This requirement can burden the already constrained human resources of smaller companies. To address this complexity, companies often resort to hiring external consultants or reassigning key personnel to the implementation process, thus detracting them from their core functionalities [20]. An additional obstacle lies in the organizational resistance towards change. Employees habituated to established processes might exhibit reluctance to embrace novel systems, driven by anxieties concerning potential job displacement or the substantial learning curve necessitated by the implementation of new technologies [21]. Furthermore, SMEs might lack the technical infrastructure necessary to support advanced ERP and MRP systems, including robust IT support and reliable network capabilities [22]. Data security concerns also play a role, as smaller firms may not have the necessary safeguards in place to protect sensitive business information during and after the transition to these integrated systems [23]. Lastly, the potential disruption to business operations during the transition phase poses a significant risk. SMEs often operate with tight margins and cannot afford the downtime that might accompany the deployment of a new ERP or MRP system [24]. Addressing these barriers is vital for leveraging the full potential of SMEs and fostering a more dynamic and resilient global economy.

In this context, adopting Python-based solutions could be a promising solution. Python is a high-level object-oriented programming language suitable for distributed application development, scripting, numerical computation, and system testing, among other uses. It was designed by Guido van Rossum in the early nineties [25]. Python's permissive open-source license grants users' extensive freedom to utilize, distribute, or even commercialize Python-based applications without seeking additional authorization. This liberal licensing structure eliminates concerns associated with vendor lock-in and fosters broader application portability across diverse platforms. The language offers clean syntax and sophisticated constructs that allow one to write procedurally or fully object-oriented, depending on the specific needs. Its powerful interactive interpreter enables real-time code development and immediate experimentation, eliminating the need for a compilation phase that could reduce productivity. The ability to extend Python with custom-compiled code allows to train Python to perform any task as fast as the hardware supports. Python can be moreover embedded into existing applications, adding an intuitive, easy-to-use interface to older but reliable software. Finally, the flexibility to interact with a wide range of other software in the system allows one to exploit already acquired software skills to the best advantage [26].

As for the use of Python-based solutions for the implementation of MRP systems in SMEs, there emerges a research gap. In scientific literature, indeed, it is possible to find sources that deal with the integration of IT and web solutions for the implementation of MRP (e.g., [27]), but none has used Python for the implementation of MRP systems been addressed in depth. In this regard, this paper aims to investigate the benefits offered by implementing a Python-based algorithm to facilitate the adoption of an MRP system in SMEs. To this end, a Python algorithm has been developed based on a sequence of four logical steps and has been numerically applied.

The rest of the paper is structured as follows: Section 2 illustrates the logical sequence considered for the resolution of an MRP and, hence, for the development of the Python algorithm. Section 3 shows the results obtained from the numerical application of the developed algorithm and, subsequently, Section 4 highlights the main contributions of this work to both knowledge and practice. Finally, Section 5 highlights the conclusions of the present work together with insights for future studies.

2. Python algorithm development

This section illustrates the sequence of logical steps adopted for solving an MRP and its subsequent modeling in Python, according to the objective of the present work.

First, it is useful to frame the main inputs and outputs of the MRP and, consequently, of the developed algorithm. As can be seen from Fig 1, an MRP has three main inputs: the Master Production Schedule (MPS), the Bill of Materials (BOM) of components, and inventory records.

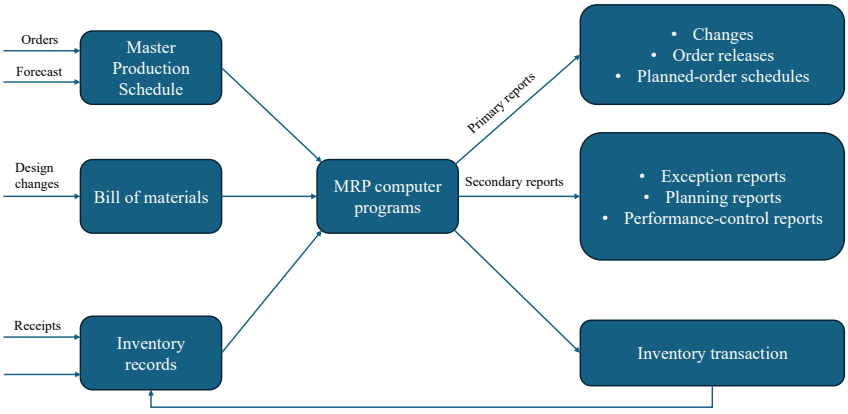

Fig. 1. Schematic representation of the main inputs and outputs of an MRP. Adapted from [28].

The MPS is a production plan that establishes the quantity and the period (or time bucket) in which the end item is required. The MPS is derived from demand forecasts, customer orders, or orders needed to replenish inventories. The BOM, on the other hand, is a document that contains a list of assemblies, sub-assemblies, and elementary parts that are required to produce a unit of the end item. The main feature of this list is its hierarchical nature, which facilitates the identification of the required quantities of parts to assemble or produce a higher grade, or parent, component. Because of this feature, the BOM is represented by the so-called product structure tree. Inventory records, finally, provide information concerning the inventory status of each part, as well as the lead times required for procurement and lot-sizing policies. As for the main outputs, the MRP produces so-called primary reports, i.e., main results, and secondary reports, i.e., optional but useful results for monitoring and evaluating system performance. The primary reports generally include the planning of future orders, the authorization of their execution and the review of previously planned orders. Secondary reports, on the other hand, include reports that are useful for evaluating system performance in terms of planning effectiveness and forecasting future inventory needs [28]. It is noteworthy that only primary reports have been considered in the context of the present work.

Accordingly, in the developed Python algorithm, the three main inputs of the MRP have been modelled, along with cumulative Lead Time (LT) for each component, which is essential for the correct time scheduling of orders.

As for MRP processing, it has followed the logical sequence of four steps proposed by [29], i.e., explode, net, lot size, and offset.

- Exploding: is the calculation, for each time bucket considered in the planning (t), of the gross requirements G(t) of the considered component. It is determined differently depending on the level of the BOM at which the analyzed component is located. For level 0 components such as the end item, G(t) is derived directly from the MPS. On the other hand, for lower-level components, the G(t) is determined by jointly considering the demand for the component at the upper level and the quantity needed to produce it, derived from the BOM.
- Netting: this step consists of adjusting the gross demand by also considering the quantities available in stock at the previous period, i.e., the Projected-on hand H(t-1), and those that are planned to be available at the beginning of period t, defined as Scheduled receipts S(t). The net demand for component N(t) is then determined by subtracting H(t-1) and S(t) from the G(t). Depending on whether this quantity is less than or equal to zero, or greater than zero, the actual needs of the analyzed component are determined.
- Lot sizing: this step consists of determining the batch size to be produced and ordered. As is well known, this is a strategic choice, as it directly influences production costs. In the context of this work, however, the so-called lot-for-lot policy, in which an order is placed equal to the demand [30], was considered as the only lot sizing policy. Once the lot-for-lot dimensioning policy has been defined, it is possible to determine, for each time bucket, a quantity that is called Planned order receipts P(t), i.e., the quantity of the component that is planned to be received in time bucket t. Once P(t) has been determined, the inventory level for each time bucket H(t) can be updated. It is obtained by jointly considering the quantities already

in stock from the previous period H(t-1), the orders that are planned to arrive in period t S(t), those needed to cover the net requirement P(t) and the gross requirement G(t), which, in contrast to the previous quantities, represents items leaving the warehouse.

- Offsetting: this last step consists of the time placement of orders P(t) considering the cumulative LT of each component. The quantity named Planned-order releases R(t) corresponds to the order placement of a quantity equal to P(t) at a time instant equal to t-LT.

The outputs of the algorithm are therefore N(t), P(t), H(t) and R(t).
The developed Python algorithm is reported in Fig 2.

```python
import pandas as pd

BOM = {
    'A': {'B': 4},
    'B': {'C': 2},
    'C': {}
}

MPS = {
    'A': [0, 0, 0, 0, 150, 0]
}

H(0) = {
    'A': 0,
    'B': 0,
    'C': 0
}

S(t) = {
    'A': [0, 0, 0, 0, 0, 0],
    'B': [0, 0, 50, 0, 0, 0],
    'C': [0, 0, 0, 150, 0, 0]
}

LT = {
    'A': 1,
    'B': 1,
    'C': 1
}

n_periods = 6

def calculate_mrp(item):
    N(t) = [0] * n_periods
    P(t) = [0] * n_periods
    R(t) = [0] * n_periods
    H(t) = [H(0)[item]] + [0] * (n_periods - 1)

    for period in range(n_periods):
        if period > 0:
            H(t)[period] = H(t)[period - 1]
        N(t)[period] = MPS.get(item, [0]* n_periods)[period] - S(t)[item][period] - H(t)[period]

        if N(t)[period] > 0:
            P(t)[period] = N(t)[period]
        else:
            P(t)[period] = 0
            N(t)[period] = 0

        release_period = period - LT[item]
        if release_period >= 0:
            R(t)[release_period] = P(t)[period]

        if period > 0:
            H(t)[period] = (H(t)[period - 1] +
                                    P(t)[period] +
                                    S(t)[item][period] -
                                    MPS.get(item, [0]* n_periods)[period])

    mrp_data = {
        'Net Requirements': N(t),
        'Planned Order Receipts': P(t),
        'Planned Order Releases': R(t),
        'Projected On Hand': H(t)
    }

    return pd.DataFrame(mrp_data)

mrp_results = {}

def calculate_mrp_for_all_components(component, component_demand):
    MPS[component] = component_demand
    mrp_results[component] = calculate_mrp(component)

    for sub_component, quantity in BOM[component].items():
        sub_component_demand = [quantity * mrp_results[component]['Planned Order Releases'][period] for period in
range(n_periods)]
        calculate_mrp_for_all_components(sub_component, sub_component_demand)

calculate_mrp_for_all_components('A', demand['A'])

for component, df in mrp_results.items():
    print(f"MRP for {component}:\n{df}\n")
```

Fig. 2. Python algorithm developed.

As can be observed in Figure 2, the script has been developed using the *pandas* library, as it allows the manipulation of data arranged in numerical tables and time series. In particular, the *DataFrame* function has been used, which allowed the results of the MRP calculations to be recorded and illustrated in table form. As for the inputs of the MRP, they have been modelled as dictionaries. To this concern, the BOM has been modelled as a dictionary where each key represents a component, and its value is another dictionary representing the sub-components required to produce one unit of the component. MPS is a dictionary that defines the demand for the top-level component (A) over six periods. H (0) is a dictionary that provides the starting inventory levels for each component. S (t) dictionary, i.e., the scheduled receipts, contains the quantity of each component that is expected to be received in each period, based on previous orders. Finally, the LT dictionary contains the cumulative LTs of each component. For MRP calculation for each component (referred to as item), the function *calculate_mrp* has been defined. As can be seen, the function first initializes the lists to record, for each period and for each component, the values of net requirements, planned order receipts, planned order releases and projected on hand. Subsequently, for each period after zero, the function calculates the net requirements as the difference between the expected demand, the scheduled receipts for the considered period and the projected on hand, i.e. the quantity already in stock. If the calculated net requirements are less than or equal to zero, the planned order receipts are set equal to zero, otherwise, a quantity in planned order receipts is set equal to the net requirements. It is noteworthy that this is valid since only a lot-for-lot sizing policy is being considered in the context of the present work. For periods in which a quantity of planned order receipts greater than zero has been set, an order is then planned in a slot time equal to the difference between the period in which P(t)>0 and the LT of the considered component. Finally, for each slot time, the projected on-hand inventory for the current period is updated by considering the inventory from the previous period, the planned order receipts, the scheduled receipts, and the current period's demand. The developed script, therefore, allows recursively calculating MRP entries for components at all levels of the BOM starting from the component demand at the top level. Finally, the results obtained for each component are printed.

The developed script has been validated using a progressive approach. It has been applied first to very simple and well-known numerical cases (e.g., a single-level BOM and a question over a few periods) and the results obtained have been compared with those calculated manually. Subsequently, edge cases have been considered (e.g., no demand in the MPS, significant initial inventory, etc.). Once these aspects have been verified, the developed algorithm has been applied to more complex cases, one of which is reported in the following section.

3. Numerical application

The developed Python algorithm has been numerically applied to the case whose product structure tree is illustrated in Fig 3.

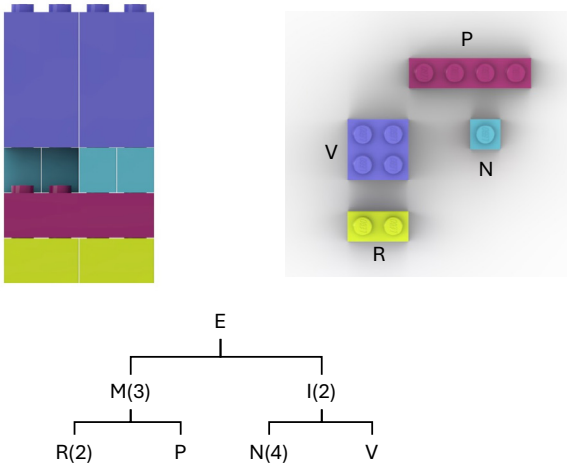

Fig. 3. Product structure tree of the analysed case. Adapted from [28]

Since it has been used a numerical example, an abstract product has been considered, resulting from the assembly of different elementary bricks (i.e., N, P, R, V). Components R and P (i.e., purple, and yellow bricks) form subassembly M and components N and V (i.e., light blue and violet bricks) form subassembly I. M and I, assembled in turn, form the finished product.

The objective is to obtain the MRP for component R, which, as can be seen, is second level and is required in two units to produce one unit of M. In the given example, it is also assumed that level zero and level one components have a LT of 1 week and level two components have a LT of 2 weeks, that there is no initial stock, but that it is planned to receive 60 units of M at the beginning of week two and 100 units of R at the beginning of week one. The input data for MRP calculation through the developed algorithm are illustrated in Fig 4.

```
BOM = {
    'E': {'M': 3}, MPS = {
    'M': {'R': 2},      'E': [0, 0, 0, 0, 0, 120]
    'R': {}         }
}
H(0) = {      S(t) = {
    'E': 0,         'E': [0, 0, 0, 0, 0, 0],    LT = {
    'M': 0,         'M': [0, 0, 60, 0, 0, 0],       'E': 1,
    'R': 0          'R': [0, 100, 0, 0, 0, 0]       'M': 1,
}             }                                     'R': 2
                                                  }
```

Fig. 4. Input data for the considered numerical application

```
MRP for E:
   Net Requirements  Planned Order Receipts  Planned Order Releases  \
0                 0                       0                       0
1                 0                       0                       0
2                 0                       0                       0
3                 0                       0                       0
4                 0                       0                     120
5               120                     120                       0

   Projected On Hand
0                  0
1                  0
2                  0
3                  0
4                  0
5                  0

MRP for M:
   Net Requirements  Planned Order Receipts  Planned Order Releases  \
0                 0                       0                       0
1                 0                       0                       0
2                 0                       0                       0
3                 0                       0                     300
4               300                     300                       0
5                 0                       0                       0

   Projected On Hand
0                  0
1                  0
2                 60
3                 60
4                  0
5                  0

MRP for R:
   Net Requirements  Planned Order Receipts  Planned Order Releases  \
0                 0                       0                       0
1                 0                       0                     500
2                 0                       0                       0
3               500                     500                       0
4                 0                       0                       0
5                 0                       0                       0

   Projected On Hand
0                  0
1                100
2                100
3                  0
4                  0
5                  0
```

Fig. 5. Results obtained from the numerical application of the developed algorithm.

From the application of the developed algorithm, the result shown in Fig 5 has been obtained, which, as can be observed, is consistent with the results obtainable from the manual execution of the required MRP (Tables 1-4).

Table 1. MPS for end item E

Week	Current	1	2	3	4	5
Quantity						120

Table 2. MRP records for end item E

E (LT=1 week)	Current	1	2	3	4	5
G(t)						120
S(t)						
H(t)	0					
N(t)						120
P(t)						120
R(t)					120	

Table 3. MRP records for component M

M (LT=1 week)	Current	1	2	3	4	5
G(t)					360	
S(t)			60			
H(t)	0		60	60		
N(t)					300	
P(t)					300	
R(t)				300		

Table 4. MRP records for component R

R (LT=2 weeks)	Current	1	2	3	4	5
G(t)				600		
S(t)		100				
H(t)	0	100	100			
N(t)				500		
P(t)				500		
R(t)		500				

It is therefore possible to observe that the developed algorithm simultaneously speeds up production planning operations in SMEs and makes them more efficient, thus promoting the digitization process at all company levels. Through the implementation of the developed algorithm, indeed, it is possible to automate MRP drafting operations by means of a low-cost tool. In this way, the company that were to implement a Python-based planning system could begin to optimize procurement and production and, likewise, could begin to track information, generating the basis for the correct implementation of more complex systems. Ultimately, the user-friendliness of the proposed tool allows to lay the foundations for the cultural change that, as mentioned, underlies the successful implementation of computerized business management systems.

4. Contributions to knowledge and practice

The present work offers significant contributions both to knowledge and to practitioners, i.e., managers of SMEs who intend to adopt effective techniques to begin a path of digitization and efficiency in operations management. First, in terms of the contributions offered to knowledge, the present work has made an initial attempt for bridging the gap that has emerged regarding the lack of sources that have dealt with the evaluation of the use of Python for the simultaneous easy and effective automation of operations management processes in contexts such as those of SMEs, where barriers to the implementation of complex information systems are encountered. Through the development of the proposed algorithm, indeed, it has been demonstrated that it is possible to develop simple, low-cost solutions that can be easily shared and transferable among industrial contexts. The development of a Python-based tool, moreover, allows for creating a baseline that can be easily improved by researchers and experts. Regarding the contributions offered to practitioners, it is noteworthy that the proposed tool can be easily used in any manufacturing context and has characteristics of generality that make it customizable and adaptable to any context. It could also be adopted as a pilot solution to assess how flexible and change-ready an organization is, as well as used to conduct training campaigns preliminary to the implementation of more complex operations management information systems. This tool could also be used, in a first phase, for measuring system performance and as a starting point for the creation of a comprehensive data tracking system.

5. Conclusions

The objective of this study was to investigate the benefits offered by implementing a Python-based algorithm to facilitate the adoption of an MRP system in SMEs. To this end, a Python algorithm has been developed based on a sequence of four logical steps: exploding, netting, lot sizing, and offsetting. The algorithm takes as input the product's BOM, the cumulative lead times of all components, and inventory records, enabling the planning of orders for individual components that constitute the finished product and updating the inventory status. A numerical application demonstrated how the developed algorithm significantly speeds up planning operations, even in cases of high complexity. Starting from the product's BOM, the algorithm can plan considering the hierarchical structure of each component and their characteristics (e.g., lead time, initial inventory, etc.). The developed tool is highly flexible and serves as a starting point for overcoming the barriers commonly encountered in the digitalization of SMEs.

Although this study offers significant contributions to both knowledge and practice, it has limitations. The primary limitation is the lack of application of the developed algorithm to a real case study. Moreover, the developed algorithm shares the same limitations as traditional MRP systems. It relies on LTs and demand forecasts and does not account for capacity constraints. From a technical perspective, Python, while versatile, is not the fastest language for computational tasks, which could lead to performance issues with very complex BOMs. In this respect, a structured debugging phase is essential to prevent the propagation of logical and calculation errors. Python-based system implementation could pose challenges for SMEs. They would have to train the necessary expertise, as well as invest in the systematization of the data management and foster a change of organizational culture. Future studies could therefore improve the tool based on the results obtained from its application to a real case study and enhance the integration of the MRP module with other planning modules, including aspects related to strategic order decisions. In addition, supplements to the developed code could be developed to include aspects neglected in the present work, such as the stochasticity of lead times and capacity limit calculations. The expansion of this work could constitute a comprehensive body of knowledge on the digitalization of business process management in SMEs.

References

[1] V. A. Mabert, “The early road to material requirements planning,” *Journal of Operations Management*, vol. 25, no. 2, pp. 346–356, Mar. 2007, doi: 10.1016/j.jom.2006.04.002.

[2] B. Gudmundsson, “Material Requirements Planning MRP Meaning: A Comprehensive Guide.” [Online]. Available: https://www.inecta.com/blog/material-requirements-planning-mrp-meaning-a-comprehensive-guide

[3] C. Öztürk and A. M. Örnek, “A MIP based heuristic for capacitated MRP systems,” *Comput Ind Eng*, vol. 63, no. 4, pp. 926–942, Dec. 2012, doi: 10.1016/j.cie.2012.06.005.

[4] A. Iasya and Y. Handayati, “Material requirement planning analysis in micro, small and medium enterprise case study: grooveline

- an apparel outsourcing company final project," *Journal of Business and Management*, vol. 4, no. 3, pp. 317–329, 2015.
[5] M. J. Shofa, A. O. Moeis, and N. Restiana, "Effective production planning for purchased part under long lead time and uncertain demand: MRP Vs demand-driven MRP," *IOP Conf Ser Mater Sci Eng*, vol. 337, p. 012055, Apr. 2018, doi: 10.1088/1757-899X/337/1/012055.
[6] G. Coviello, G. Avitabile, A. Florio, C. Talarico, and J. M. Wang-Roveda, "A Novel Low-Power Time Synchronization Algorithm Based on a Fractional Approach for Wireless Body Area Networks," *IEEE Access*, vol. 9, pp. 134916–134928, 2021, doi: 10.1109/ACCESS.2021.3115440.
[7] A. Adarsh, "Demand Driven Material Requirements Planning," 2020. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:1509015/FULLTEXT01.pdf
[8] W. Brown, "Enterprise resource planning (ERP) implementation planning and structure," in *Proceedings of the 32nd annual ACM SIGUCCS conference on User services*, New York, NY, USA: ACM, Oct. 2004, pp. 82–86. doi: 10.1145/1027802.1027824.
[9] G. Coviello, A. Florio, G. Avitabile, C. Talarico, and J. M. Wang-Roveda, "Distributed Full Synchronized System for Global Health Monitoring Based on FLSA," *IEEE Trans Biomed Circuits Syst*, vol. 16, no. 4, pp. 600–608, Aug. 2022, doi: 10.1109/TBCAS.2022.3173586.
[10] W.-H. Tsai, S.-W. Chien, P.-Y. Hsu, and J.-D. Leu, "Identification of critical failure factors in the implementation of enterprise resource planning (ERP) system in Taiwan's industries," *International Journal of Management and Enterprise Development*, vol. 2, no. 2, p. 219, 2005, doi: 10.1504/IJMED.2005.006312.
[11] N. Mignoni, P. Scarabaggio, R. Carli, and M. Dotoli, "Control frameworks for transactive energy storage services in energy communities," *Control Eng Pract*, vol. 130, p. 105364, Jan. 2023, doi: 10.1016/j.conengprac.2022.105364.
[12] M. Baymout, "ERP systems in supply chain management," *International Journal of Advance Research*, vol. 2, no. 3, 2014.
[13] Panorama Consulting Solutions, "2015 ERP Report: A Panorama Consulting Solutions Research Report," 2015. [Online]. Available: https://www.panorama-consulting.com/wp-content/uploads/2016/07/2015-ERP-Report-3.pdf
[14] World Bank Group, "Small and Medium Enterprises (SMEs) Finance." [Online]. Available: https://www.worldbank.org/en/topic/smefinance
[15] M. Ayyagari, A. Demirguc-Kunt, and V. Maksimovic, "Small vs. Young Firms across the World: Contribution to Employment, Job Creation, and Growth," 2011. [Online]. Available: https://documents.worldbank.org/en/publication/documents-reports/documentdetail/478851468161354807/small-vs-young-firms-across-the-world-contribution-to-employment-job-creation-and-growth
[16] T. Tambunan, "SME Development in Indonesia: Do Economic Growth and Government Supports Matter?," *SSRN Electronic Journal*, 2008, doi: 10.2139/ssrn.1218922.
[17] UN trade and development, "Micro-, Small, and Medium-sized Enterprises Day: Trade for Climate Change." [Online]. Available: https://unctad.org/meeting/micro-small-and-medium-sized-enterprises-day-trade-climate-change
[18] T. Beck, A. Demirgüç-Kunt, and D. Singer, "Is Small Beautiful? Financial Structure, Size and Access to Finance," *World Dev*, vol. 52, pp. 19–33, Dec. 2013, doi: 10.1016/j.worlddev.2013.05.014.
[19] C. Marnewick and L. Labuschagne, "A conceptual model for enterprise resource planning (ERP)," *Information Management & Computer Security*, vol. 13, no. 2, pp. 144–155, Apr. 2005, doi: 10.1108/09685220510589325.
[20] H. M. Beheshti, "What managers should know about ERP/ERP II," *Management Research News*, vol. 29, no. 4, pp. 184–193, Apr. 2006, doi: 10.1108/01409170610665040.
[21] E. W. T. Ngai, C. C. H. Law, and F. K. T. Wat, "Examining the critical success factors in the adoption of enterprise resource planning," *Comput Ind*, vol. 59, no. 6, pp. 548–564, Aug. 2008, doi: 10.1016/j.compind.2007.12.001.
[22] A. S. Shatat, "Critical Success Factors in Enterprise Resource Planning (ERP) System Implementation: An Exploratory Study in Oman," *The Electronic Journal of Information Systems Evaluation*, vol. 18, no. 1, pp. 36–45, 2015.
[23] D. L. Olson and S. Kesharwani, *Enterprise Information Systems: Contemporary Trends And Issues*. 2009.
[24] M. M. Ahmad and R. Pinedo Cuenca, "Critical success factors for ERP implementation in SMEs," *Robot Comput Integr Manuf*, vol. 29, no. 3, pp. 104–111, Jun. 2013, doi: 10.1016/j.rcim.2012.04.019.
[25] G. van Rossum and F. L. Jr. Drake, *An Introduction to Python*. 2003.
[26] T. E. Oliphant, "Python for Scientific Computing," *Comput Sci Eng*, vol. 9, no. 3, pp. 10–20, 2007, doi: 10.1109/MCSE.2007.58.
[27] T. Rossi, R. Pozzi, M. Pero, and R. Cigolini, "Improving production planning through finite-capacity MRP," *Int J Prod Res*, vol. 55, no. 2, pp. 377–391, Jan. 2017, doi: 10.1080/00207543.2016.1177235.
[28] W. J. Stevenson, *Operations Management*, 14th ed. McGrawHill, 2021.
[29] K. R. Baker, "Chapter 11 Requirements planning," 1993, pp. 571–627. doi: 10.1016/S0927-0507(05)80191-4.
[30] W. Florim, P. Dias, A. S. Santos, L. R. Varela, A. M. Madureira, and G. D. Putnik, "Analysis of lot-sizing methods' suitability for different manufacturing application scenarios oriented to MRP and JIT/Kanban environments," *Brazilian Journal of Operations & Production Management*, vol. 16, no. 4, pp. 638–649, Nov. 2019, doi: 10.14488/BJOPM.2019.v16.n4.a9.