IJEEC

International Journal of Electrical Engineering and Computing
Vol. 10, No. 1 (2026)

Case Studies
UDC
DOI

Product Structure: A Data Model in the Context of Small and Medium-Sized Enterprises (SMEs)

Miroslav Dragić[1], Živko Pavlović[2], and David Ištoković[3]

[1] Faculty of Technology, University of Banja Luka, Banja Luka, Bosnia and Herzegovina
[2]Faculty of technical sciences, University of Novi Sad, Novi Sad, Serbia
[3]Faculty of Engineering, University of Rijeka, Rijeka, Croatia

E-mail address: miroslav.dragic@tf.unibl.org, zivkopvl@uns.ac.rs, distokovic@riteh.uniri.hr

Abstract—**In contemporary business environments—particularly amid accelerated digitalization—the capability for systematic management of information on products and production processes has become pivotal to achieving competitive advantage. For manufacturing small and medium-sized enterprises (SMEs), which operate under constraints in informational, material, and financial resources and under constant pressure to reduce costs and shorten delivery lead times, a clearly structured approach to product data management constitutes a key lever for achieving efficiency, quality, and market flexibility. In this context, the product structure component (PSC) and the corresponding data model are prerequisites for consistent, scalable, and interoperable information management throughout the entire product life cycle (from concept, development, and production to maintenance and recycling). Within this work, different approaches to defining PSC and the associated data models are considered, with a particular focus on how SMEs can achieve consistent management of product structure data, change and revision control, support for variant management, along with a rational selection of database technology and system architecture. This paper examines different approaches to defining product-structure data, including relational, object-oriented, and hybrid (multi-model) models. Different approaches to product data modeling for a manufacturing company are presented. The data model will include product structure data (engineering BOM), customer orders, and document management models. Finally, a brief overview of the advantages and disadvantages of each of the proposed solutions is given.**

Keywords- SMEs; data model; RDBMS; OODBMS, MMDBMS

I. INTRODUCTION

The product structure component (PSC) represents a formal description of the hierarchical content (components) of a product (assemblies, subassemblies, parts, and materials), including relationships between constituent components (e.g., "part A is part of assembly B"), and attributes that define these relationships (quantity, unit of measure, variants, revisions, and the like). As such, the PSC provides the foundation for procedures such as defining and tracking changes in product configurations, production planning, procurement planning, cost tracking, and establishing traceability [1].

In practice, the PSC most often appears in two variants:

- Engineering Bill of Materials (EBOM), which contains only the elements belonging to the product's structure (for example, for an electric motor: housing, rotor, stator, control board, etc.), and
- Manufacturing Bill of Materials (MBOM), which, in addition to the elements in the EBOM, includes additional resources required for manufacturing, such as tools, auxiliary materials (cutting inserts, electrodes, surface preparation and finishing materials, energy sources and so on), measuring and inspection equipment, and other items [2].

Managing EBOM/MBOM information is often a key distinguishing issue within the domain of PDM/PLM (Product Data Management / Product Lifecycle Management) systems. This is because PDM/PLM typically provides control over documents and revisions, management of statuses (draft/released/obsolete), product configuration management, and support for change and approval workflows [3].

A data model is a set of concepts, structures, and rules that determine how information about PSCs and manufacturing processes is represented, stored, verified, and exchanged. This model must necessarily include:

- Semantics, defined by a set of entities (materials, parts, assemblies, variants) and interrelations (contains, substitutes, belongs to, etc.);
- Constraints (mandatory attributes, predefined, max/min values, variant rules, etc.);
- Change management model;

This paper is a revised and expanded version of the paper presented at the XXV International Symposium INFOTEH-JAHORINA 2026

- Interoperability and data exchange (e.g., concepts from standards like ISO 10303 STEP or other exchange formats) [4].

In contrast to "simple" data storage, PDM/PLM serves as an organizational and semantic anchor for linking the product structure, the associated documentation, and the product lifecycle.

The most commonly used data models include relational, object-oriented, and hybrid (multi-model) models. When selecting criteria to evaluate the type of model and the DBMS, one should consider the nature and structure of the data, the execution environment, the programming languages used, the programmer's experience with particular DBMSs, cost, duration of support and maintenance, functionality, and performance [5].

Fig. 1 shows a partial view of the engineering BOM for a complex product created in the ERP system NESoft_PG (Developed by Nesoft d.o.o Banja Luka company). In this case, the product has six hierarchical levels of integration, and several hundred components.

Choosing the optimal data model and DBMS is especially important for SMEs. These enterprises, particularly those in manufacturing, operate under chronic resource constraints—financial, material, personnel, and IT resources. Their advantage relative to large organizations lies in a high degree of adaptability and project-oriented operation, often driven by direct customer requirements [6].

Hierarchical levels — *Component data* — *Quantity*

0 1 2 3 4 5 6	Pozici	Sifra	IDPoz	Oznaka	Naziv	JM	Kolicina
		P	400000	BTSK 1x1000	<> Blindirana trafo-stanica BTSK 1x100(	kom	1
	1	S	550001		NOSECA KONSTRUKCIJA BTSK 1x1000	kom	1
	1	D	500012		Postolje noseće konstrukcije - POZ. 1	kom	1
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	3.724
	2	D	500014		Postolje noseće konstrukcije - POZ. 2	kom	2
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	2.76
	3	D	500015		Postolje noseće konstrukcije - POZ. 3	kom	3
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	2.6634
	4	D	500016		Postolje noseće konstrukcije - POZ. 4	kom	6
	1	M	200038	EN 10025 S23	Lim tvl 6mm St37-2 DIN 17100	m2	0.0069
	5	D	500013		Postolje noseće konstrukcije - POZ. 5	kom	1
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	3.724
	6	D	500017		Nosač trafoa - POZ. 1	kom	4
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	2.6438
	7	D	500018		Nosač trafoa - POZ. 2	kom	2
	1	M	200231	EN 10025 S23	UNP 100 St 37-2 DIN 1026	m	2.99
	8	D	500019		Nosač trafoa - POZ. 3	kom	4
	1	M	200038	EN 10025 S23	Lim tvl 6mm St37-2 DIN 17100	m2	0.0174
	9	D	500020		Nosač NN bloka	kom	4
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	0.5607
	10	D	500021		Umetak za nosače blokova	kom	10
	1	M	200038	EN 10025 S23	Lim tvl 6mm St37-2 DIN 17100	m2	0.0098
	11	D	500022		Nosač SN bloka - poz 2	kom	2
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	0.2046
	12	D	500023		Pločica za anker	kom	4
	1	M	200014	EN 10025 S23	Lim tvl 8mm St37-2 DIN 17100	m2	0.0225
	13	D	500044		Nosač SN bloka - poz 1	kom	2
	1	M	200049	EN 10025 S23	Lim tvl 4mm St37-2 DIN17100	m2	0.4452
	14	D	500047		Nosač SN bloka - poz 3	kom	2
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	0.3949
	15	D	500048		Nosač SN bloka - poz 4	kom	2
	1	M	200222	EN 10025 S23	UNP 65 St 37-2 DIN 1026	m	0.5192
	16	S	550028		Omega za merdevine	kom	2
	1	M	200006	DC01 EN 101	Lim hvl 2mm St12 DIN 1623-1	m2	0.0056
	2	M	212285	DIN 32501	Vijak Privarilni M6x16 Cu	kom	1
	17	D	500049		Omega pregrade poz1	kom	1

Figure 1. PSC of the product Shielded transformer station (partial view) in the NESOFT_PG software

From a sustainability perspective, for SMEs it is crucial that the chosen PSC data and data model solutions are:

- easy to implement and maintain,
- interoperable with existing tools (CAD, PDM/PLM, ERP/MRP, MES),
- scalable with increasing complexity and product programs,
- cost-effective and aligned with the real needs of the enterprise, and
- durably supported by both DBMS manufacturers and by system designers and implementers [7,8].

This work primarily addresses SMEs in manufacturing, especially mechanical and electro-mechanical sectors, in defining the requirements for an information system. Some questions this work aims to answer include:

- How to define a PSC that simultaneously supports engineering, production, and service perspectives without data duplication and inconsistency?
- What is the minimal set of concepts and data-model rules necessary for SMEs to achieve traceability of changes and efficient variant management?
- Which data architectures best balance simplicity, performance, and the ability to integrate the data model into SMEs?
- How to measure and demonstrate the benefits (quality, time, cost) of implementing a PSC data model in SME practice?

II. Problem Definition and Challenges

Although available solutions are relatively expensive and complex to implement, SMEs in manufacturing need an efficient information system to manage all elements of business operations. Even if we set aside legally required elements for financial and material accounting, support for product development, manufacturing, and maintenance processes becomes imperative. In addition to describing product structure and manufacturing technology, the system must minimally support customer requirements management (CRM) and management of a wide range of documentation (commercial, design, technological, work, control, and others).

The complexity of an enterprise-wide information system, and especially a system for working with product structure components, is reflected in:

- Use of a large number of components, including raw materials, finished components, standard and non-standard semifinished products (parts and assemblies);
- Relatively high product complexity, with PSCs often having 10 or more levels of assembly in the hierarchy and hundreds of elements in the product structure;
- Production that is practically custom-made or in very small batches (two or three units);
- Products that recur periodically often represent different product variants (e.g., a crane installation in

different objects with different dimensions, floor counts, etc.);

- Due to design variations, in most cases it is necessary to create technical and work documentation from scratch;
- The lead time from receiving a customer requirement to offering a quote and delivering the product is very short today; these companies must often align product configurations and delivery times with the supply chains of larger business entities;
- Because of the specifics of the production program, the company must track its products after delivery to customers through testing, servicing, and maintenance;
- Finally, the company must be prepared to forecast and assess the financial performance of each job in advance;

Challenges facing modern PSC data models include:

- Data fragmentation — Data are created and used across multiple platforms (CAD, CAM, ERP) and data exchanges with customers and suppliers, often without a single source, leading to errors and data inconsistencies;
- Variant management — Managing many variants and options requires explicit rules and efficient structures for variant management of assemblies;
- Traceability and compliance — Breaks in the chain of tagging (transfer of markings), incomplete versioning, and change management complicate audits, quality verification, and compliance with standards; this is especially sensitive in regulated industries such as welded structures, automotive, aerospace, shipbuilding, and similar;
- Interoperability and standardization — The lack of a standardized data model hinders information exchange among supply-chain partners and integration; general formats and standards are useful but often too complex for SMEs;
- Sustainability and circular economy — Embedded information about materials, repairs, and replacements can greatly facilitate maintenance, remanufacturing, and recycling, but require a more complex data model [9,10];

A particularly significant segment of PSC management is variant management. In manufacturing scenarios, SMEs often have products with variants (e.g., different dimensions, specific options, alternative materials, or assembly configuration variants). Variant management requires explicit rules and structural mechanisms that ensure:

- consistent representation of variant trees (variant configurations),
- revision control and variant compatibility checking,
- the correct linkage between EBOM/MBOM items and a given configuration, and
- change traceability in the context of an individual variant [11].

These requirements directly affect the choice of the conceptual data model and its implementation in a database (e.g., relational vs. hybrid approach), as well as integration mechanisms with documentation and workflow processes. Therefore, selecting an appropriate data model and database for engineering data (engineering data) is not only an IT decision, but also a process decision. The literature emphasizes that, in practice, the performance, maintainability, scalability, and maintenance cost of a solution differ depending on the nature of the data: hierarchical structures, versioning, relationships between components and documents, and the variant model all require careful schema design, constraints, and storage/reporting strategies [12].

Relational models are often the most widely used due to ecosystem standardization and maturity, whereas hybrid approaches (e.g., relational plus JSON/XML) are increasingly adopted when it is necessary to efficiently store complex and semi-structured data (e.g., configuration variants or document records) while still supporting strong querying capabilities and integration

III. DATA MODELING

A data model is a model that describes how data elements and their properties are identified, grouped, and interrelated, and which operations can be performed on them. The most commonly recognized and used data models and their DBMSs are: relational, document-oriented, graph database models, object-oriented databases, and increasingly hybrid (multi-model) models [13].

The following sections present different approaches to modeling product data for a manufacturing enterprise. The data model will cover data about product structure (engineering BOM), customer orders, and documentation management models (Fig. 2). This represents the minimum data required for the customer requirements management system [14].

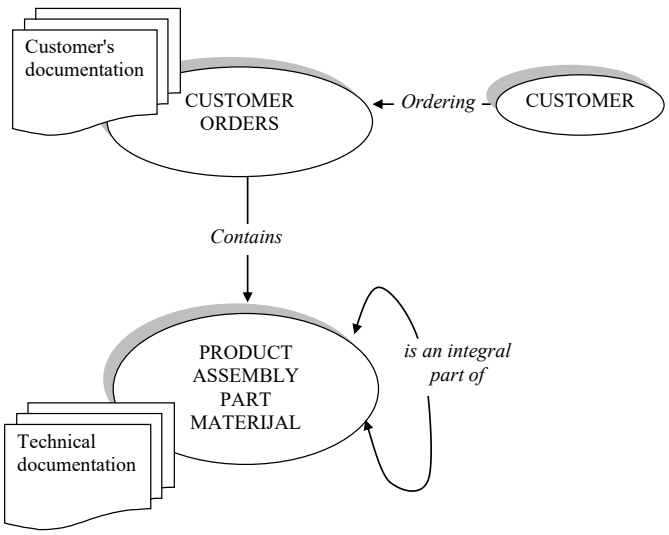

Figure 2. Elements of the CRM data model

A. Relational Model

This technology is standardized, oldest, and widely deployed. A relational database management system (RDBMS) stores data in two-dimensional tables, consisting of rows (records) and columns (fields). Each row corresponds to a record; columns denote the fields (attributes) of that record. RDBMSs typically support a large number of different but essentially simple data types. Relationships are realized via keys, which link entities in one table to those in another. For data manipulation, RDBMSs use SQL (Structured Query Language), which is standardized [15].

Fig. 3 shows an example of a simplified relational data model for the scenario described above. The model also shows how a CRM system is integrated into the data model.

On the other hand, the presented relational model does not include a model for documentation management. It can be realized in two ways:

- storing documentation data as attributes within the database corresponding to the path to documentation associated with customer orders (orders, contracts, invoices, and so on), technical documentation (drawings, specifications, process sheets, etc.), and shop-floor documentation (work orders, cutting lists, CNC programs, etc.),
- or by using predefined algorithms or organized templates for storing data linked to primary keys in tables (e.g., year-order-number, product-id, assembly, part, or material), as shown in Fig. 4.

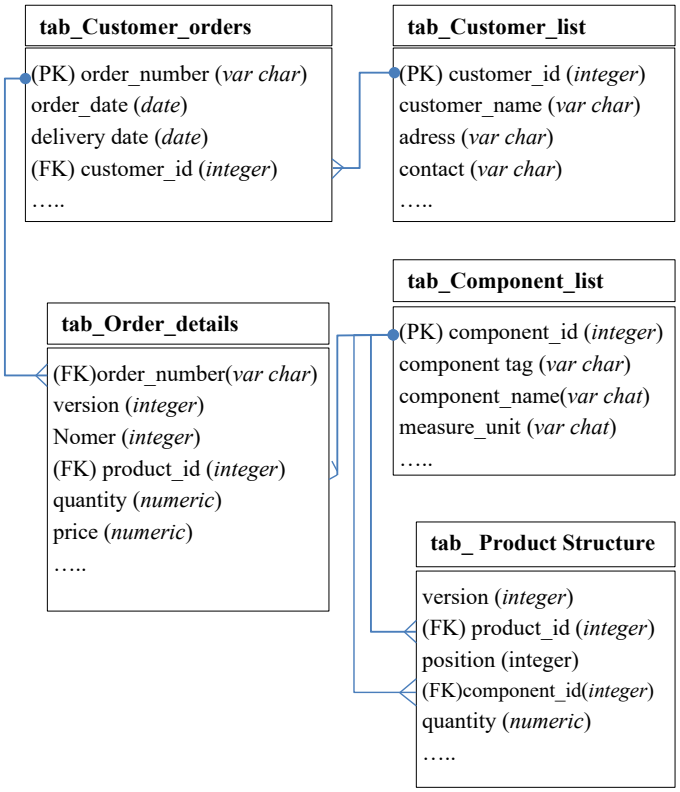

Figure 3. Product documentation storage model based on primary key - id number

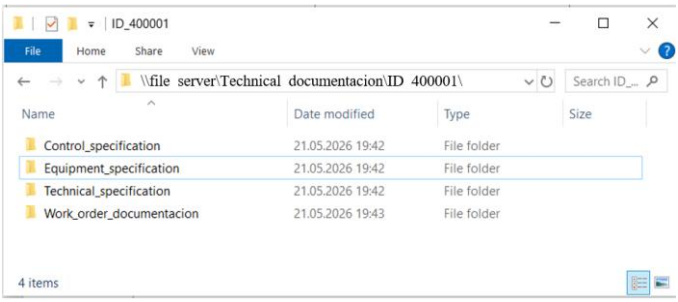

Figure 4. Product documentation storage model based on primary key - id number

B. Object-Oriented Model

In object-oriented databases, records are stored as objects of classes which, in addition to state (values of the fields within the class), possess behavior (methods and properties with get and set methods) and support inheritance. Objects are interlinked via pointers. A key similarity with relational models is that classes in object-oriented programming languages, as well as relationships, have a predefined schema. Object-oriented databases (OODBs) are tightly coupled to the programming language and execution environment (Fig. 5). Although there have been attempts since the 1990s, this technology has never been officially standardized [16].

A user-facing application in conjunction with the data model can work through an OODB by using the native object-oriented language of the application (e.g., C#, C++, Java...), which creates a direct link between application objects and objects stored in the database. Most OODBs also support SQL, primarily via ODBC (Open Data Base Connectivity).

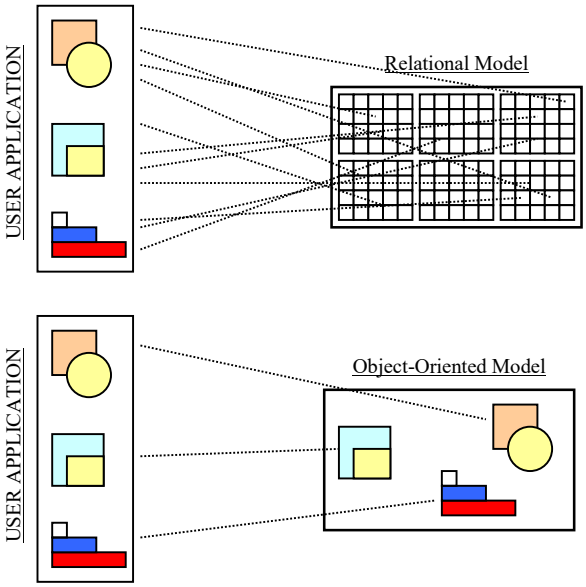

Figure 5. Object-oriented data model vs. relational data model

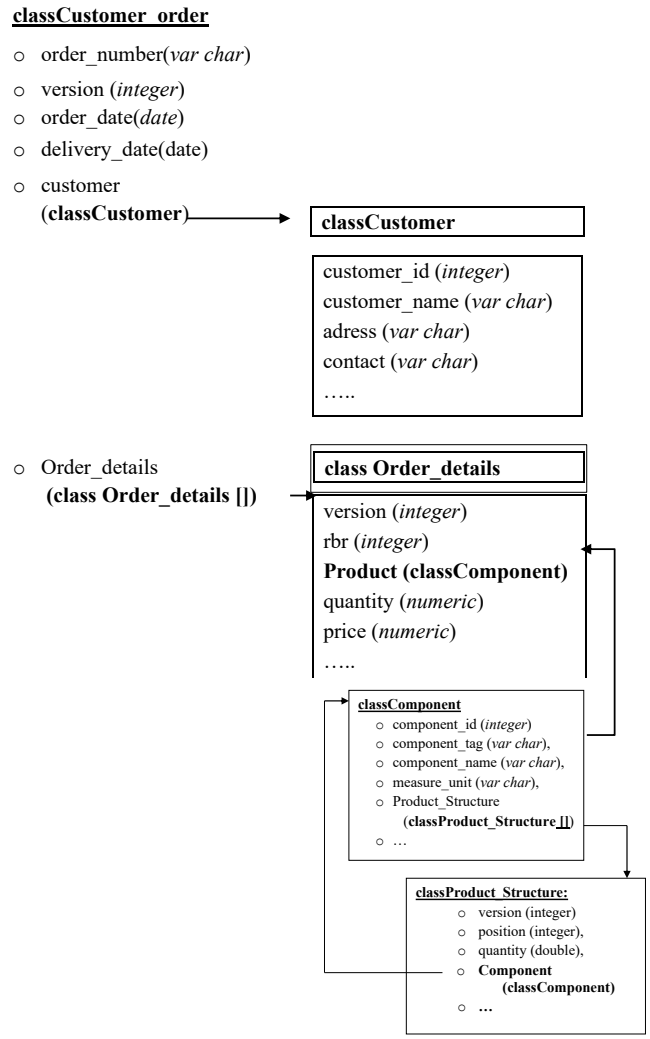

Figure 6. Object-oriented data model vs. relational data model

While OODBs technology was once touted as the inevitable future, today most popular OODBs are not widely maintained or have had their development halted. For example, db4o was last updated on 22 September 2019 and its development was halted; ObjectBox is still actively developed but supports only a limited set of programming languages (Java, Kotlin, C/C++, Python), and other popular languages such as PHP and C# are not supported. For this reason, object-oriented DBMSs are not widely used [17].

A proposed object-oriented data model for the problem is shown in Fig. 6.

In this case, the documentation management model can be realized by:

- directly inserting documentation collections into the database (as objects of type Object, File...) or
- by one of the methods described in the previous case.

Directly inserting files into the database is possible but not recommended due to increased memory usage and challenges in ensuring data integrity.

C. Hybrid Model (Multi-Model Database)

This type of database integrates multiple data models (e.g., relational with document or graph-oriented models) into a single system. Traditionally, different problems benefit from specific data models; such systems offer great flexibility of application (instead of running multiple separate systems), reduce operational costs, but increase operational complexity [18].

In the document-oriented approach data is stored as a collection of documents uniquely identified self-describing data entries. Every entry is represented as a separate XML or JSON document. Typically, documents in a collection could have a non-fixed schema and contain different sets of attributes. Query languages are often based on a mix of JavaScript and JSON syntax. For example, DBMS uses JavaScript to specify operations on the data and JSON to filter, update, insert, or delete data items [19].

The graph data model is an approach to data organization in which information is represented through nodes, relationships, and properties. Nodes represent entities, while relationships describe the connections between them, enabling efficient modeling of complex and highly interconnected systems. Unlike the relational model, the graph model is particularly suitable for analyzing networks of relationships, such as social networks, recommendation systems, and semantic knowledge bases. Owing to its flexibility and expressiveness, this model has found significant application in modern information systems [20].

Some of the more commonly used hybrid platforms today include: ArangoDB (multi-model document + graph + key-value), OrientDB (document + graph + SQL-like language and transactions), Azure Cosmos DB (multiple APIs: Core/SQ, Gremlin, Cassandra, Table, Mongo), MarkLogic (document XML/JSON + full-text with strong security and search), and PostgreSQL (relational model with extensions such as JSONB, XML, and PostGIS). Fig. 7 shows a hybrid data model adapted for PostgreSQL databases. Figures 8, 9, and 10 illustrate a simplified form of JSON and XML objects in a hybrid data model [21].

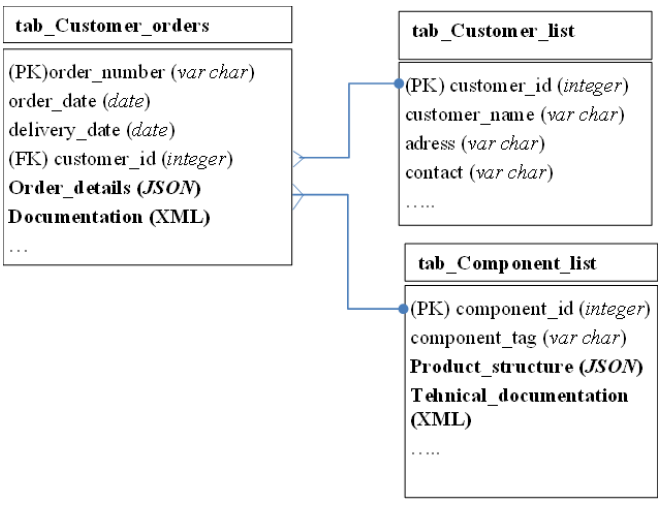

Figure 7. Hybrid Model (Multi-Model Database)

```
{
 "order": {
  "order_number": "M-355/25",
  "version": "1",
  " modification date ": "2025-11-24",
 },
 "order_details": [
  { "position": 1, "component_id": "400245", "quantity": 3, "price": 12.49 },
  { "position": 2, "component_id ": "400322", "quantity": 0.5, "price": 89.98 },
  { "position": 3, "component_id ": "401420", "quantity": 0.8, "price": 7.46 },
  { "position": 4, "component_id ": "422312", "quantity": 1.5, "price": 15.25 },
.....
 ]
}
```


Figure 8. JSON format of data record about the content of the customer's order

In this scenario, the documentation management model can be realized by inserting related documentation data in JSON or XML form, though the previously described documentation management models remain available.


```
<?xml version="1.0" encoding="UTF-8"?>
< documents collection ="Archive-2026">
 <document
  file_name=" Contract _123.pdf"
  file type ="application/pdf"
  file size ="1.25 MB"
  path="/archive/2026/contracts/Contract_123.pdf" />
 < document
  file_name="Customer_specifications_v2.xlsx"
  file type ="excelldocument/xlsx"
  file size ="842 KB"
  path="/projects /alpha/docs/Customer_specifications_v2.xlsx" />
 < document
  file_name="Construction_drawing_v2.acad"
  file type ="autocad/acad"
  file size ="5482 KB"
  path="/projects /alpha/drawings/ Construction_drawing_v2.acad " />
<... />
</ documents collection >
```


Figure 9. JSON XML data record of related documentation


```
{
 "product": {
  "id": "400265",
  "version": "1",
  " modification date ": "2025-11-24",
 },
 "product_structure": [
  { "position": 1, "component_id": "500105", "quantity": 2 },
  { " position ": 2, " component_id ": "200324", " quantity ": 0.5 },
  { " position ": 3, " component_id ": "582001", " quantity ": 4 },
  { " position ": 4, " component_id ": "252346", " quantity ": 1.25 }
  .....
 ]
}
```


Figure 10. JSON format of product structure components data

IV. RESULTS AND DISCUSSION

One of the technical challenges in transferring and exchanging product structure data and associated metadata is the standardization of both format and semantics. In this respect, ISO 10303 (STEP)—particularly the parts relevant to Product Life Cycle Support (PLCS)—enables interoperable exchange of product data between CAD/PDM/ERP/MES systems, thereby reducing the risk of information loss or semantic inconsistencies during data exchange. However, in real-world SME environments (small and medium-sized enterprises), standards are often difficult to "apply fully" without implementation guidelines and practical models that simplify integration.

TABLE I. STRUCTURED COMPARISON

Criteria	Relational model (RDBMS)	Object-oriented model (OODBMS)	Hybrid / multi-model (MMDBMS / hybrid)
Cost	👍 **Usually lowest to moderate** *mature tooling, SQL skills, lower integration cost with ERP/PDM*	⚠️ **Often higher** *smaller ecosystem, specialized skills/tools, high integration overhead*	👍 / ⚠️ **Moderate-to-high** *requires multiple storage/query layers, but may reduce complexity by using "right tool for right data"*
Scalability	👍 **Strong** *scales well with indexing, normalization, and good schema design*	⚠️ **Can be good** *performance depends heavily on object navigation patterns and implementation quality*	👍 **Strong** *if architecture is well-designed: structured parts in RDBMS + semi structured /configuration parts in document/graph components*
Maintainability	👍 **High** *schemas and constraints are explicit; debugging via SQL*	⚠️ **Low** *maintenance can be harder; schema evolution and object graph changes may require careful migration*	👍 / ⚠️ **Moderate-to-high** *clear ownership of data domains (BOM vs variants vs docs). Otherwise, can become complex.*
Traceability	👍 **Strong** *BOM revision IDs, effective dates, change objects, and explicit links between revisions / configurations and documents*	👍 / ⚠️ **Can be strong** *object identity and references may represent lifecycle relationships naturally; but constraints and queries may be more complex*	👍 👍 **Very strong** *different representations (e.g., relational for revision links + graph for impact/traceability paths) can support rich trace queries*
Suitability for SMEs	👍 **Most practical** *cost-effective, standardized skillset, easier deployment and support*	⚠️ **Not suitable** *expertise and tooling availability may be limiting for SMEs*	👍 **Often suitable** *need flexibility (complex variants/documents) and can invest in integration/architecture—but only if scope is controlled*

The Table 1. presents a structured comparison of relational, object-oriented, and hybrid (multi-model) approaches for modeling and managing product structure data (PSC), particularly in the context of EBOM/MBOM handling in SMEs. The comparison is organized around practical evaluation criteria—cost, scalability, maintainability, traceability, and overall suitability for SME constraints. Each criterion is interpreted with regard to the specific information needs highlighted in the paper, such as revision/change control, variant management support, and linking engineering data with documentation and lifecycle workflows. The goal is to identify which model type best balances SME realities (limited IT budgets and expertise) with the functional requirements of modern product data management and exchange.

Relational data models continue to be the most prevalent. This is due to their standardized, well-established nature and the large pool of experienced developers and administrators. Consequently, application development that works with relational models becomes routine, while the conceptual differences with object-oriented databases are significant. The existence of many DBMSs suitable for various platforms, along with the standardized query language (SQL) and support for data exchange between DBMSs, further justifies the continued dominance of relational databases. Nevertheless, there is a noticeable trend toward upgrading these systems, primarily by supporting more complex data types and complex objects such as JSON and XML documents. Overall, the relational data model, while apparently the most complex, remains the most logical solution for information systems in SMEs.

The object-oriented data model represents a model that best captures the problem of managing product data within a company. However, although OO databases were expected to prevail over relational databases, that did not happen. With the strong development of widespread object-oriented programming, it was natural to expect that OO databases would prevail. It can be argued that the original concept of OOP-based databases is now virtually dead, so their use is limited to large and specialized data-collection systems. The main reasons for the weak adoption of OODBMSs are that relational databases are widespread and well-developed over many years, and this technology was never officially standardized. The tight coupling of OO databases to programming languages and execution environments, which initially seems advantageous, proved problematic for using data across different platforms and environments (desktop applications, web applications, mobile applications, PDA devices, and others). Why object models are risky for SMEs? In general, it can be concluded that Object-oriented models can naturally model entities and relationships as objects, which can be attractive for representing PSC constructs. However, for SMEs:

- tooling/skills can be scarce, and
- migrations and cross-system integrations can cost more.

The hybrid model (MMDBMS) represents a good transitional solution between purely relational databases and the needs for storing and processing complex data types. Almost all major relational DBMS vendors (notably Oracle and PostgreSQL) continually broaden their capabilities to support more complex data models. These hybrid-model elements are now integrated into contemporary enterprise information systems and are financially and operationally accessible to SMEs. This data model is particularly valuable in situations where there are many versions of the same data and traceability is required. The paper explicitly notes the need to manage PSC across the full lifecycle and support variant management and interoperability. In practice, hybrid models are often best when:

- BOM/revisions are handled relationally, while
- variant rules/configurations and document semi-structured content benefit from multi-model storage (e.g., graph/document hybrid patterns).

Therefore, in the future, increasing use of hybrid data models in SME information systems is to be expected.

REFERENCES

[1] J. Stark, JProduct Lifecycle Management (Volume 1): 21st Century Paradigm for Product Realisation (4th ed.). Springer 2020.

[2] W. M. Wang, P. K. Kumar ar, “Engineering Change Management in Product Lifecycle Management Systems,” Journal of Intelligent Manufacturing, vol. 26, no. 6, pp. 1341–1356, 2015.

[3] ISO 10303-239:2012. Industrial automation systems and integration — Product data representation and exchange — Part 239: Product Life Cycle Support (PLCS).

[4] ISO 10303-1:2020, Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles, International Organization for Standardization, 2020.

[5] Popularity ranking of database management systems. DB Engines [online]. [Accessed 7.12.2025]. Available from: https://db-engines.com/en/ranking.

[6] M. Sorak, M, Dragic. Challenges for the Future – Engineering Management: Supply Chain Management of Small and Medium-Sized Enterprises, Chapter 15. Editors: Hans-Jörg Bullinger and Dieter Spath. Published by: Faculty of Technical Sciences (Novi Sad, Serbia), Fraunhofer IAO(Stuttgart, Germany) and DAAAM International (Vienna, Austria).ISBN 978-3-902734-01-3,2013. DOI: 10.2507/daaam.scibook.2013.59

[7] S. Borojević, D. Matić, M. Dragić. An Integrated Intelligent CAD/CAPP Platform: Part II-Operation Sequencing Based on Genetic Algorithm. Tehnički vjesnik 29 (5), 2022, pp.1686-1695. https://doi.org/10.17559/TV-20211012084632

[8] M.Z. Ouertani, S. Baïna, L. Gzara, G. Morel, Traceability and management of dispersed product knowledge during design and manufacturing, Computer-Aided Design, Volume 43, Issue 5, 2011, pp 546-562, https://doi.org/10.1016/j.cad.2010.03.006.

[9] J.S. Myneni. Trends and Challenges in Database Management Systems for Business Analytics. IRE Journals. Vol 8 (7), 2025

[10] A. Fawzy, A. Tahir, M. Galster, et al. Exploring data management challenges and solutions in agile software development: a literature review and practitioner survey. Empir Software Eng 30, 77 (2025). https://doi.org/10.1007/s10664-025-10630-4

[11] K. Kegel, S. Gotz, "Towards Variability-Aware Instance Handling for Model Evolution at Runtime," in 2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Durres, Albania, 2023, pp. 183-190, doi: 10.1109/SEAA60479.2023.00036.

[12] Hedberg, T., Jr., Feeney, A. B., Helu, M., and Camelio, J. A. (February 16, 2017). "Toward a Lifecycle Information Framework and Technology in Manufacturing." ASME. J. Comput. Inf. Sci. Eng. June 2017; 17(2): 021010. https://doi.org/10.1115/1.4034132

[13] G. Wojarnik. “Selection of working database for the genetic algorithm processing data of exchange quotations”. Information Systems in Management. Vol. 5, 2016. DOI: 10.26634/jsim.10.4.1481

[14] A. Fabijan,HH. Olsson, J. Bosch. The lack of sharing of customer data in large software organizations: challenges and implications. In: Proceedings of the 17th international conference on agile software development (XP). Springer, 2016,pp 39–52, DOI: 10.1007/978-3-319-33515-5_4

[15] H. Zhang, MA. Babar, P. Tell. Identifying relevant studies in software engineering. Inf Softw Technol, 2011, 53(6):625–637

[16] S. McClure, "Object Databases v.s.Object Relational Databases" , IDC Bulletin #14821E - August 1997.

[17] A. Torres, R. Galante, M.S. Pimenta, A.J.B. Martins. "Twenty years of object-relational mapping: A survey on patterns, solutions, and their implications on application design". Information and software technology, 2017, 82: p.1-18. https://doi.org/10.1016/j.infsof.2016.09.009

[18] M. Macak, M. Stovick, B. Buhnova, M. Merjavy. "How well a multi-model database performs against its single-model variants: Benchmarking orientdb with neo4j and mongodb". Proceedings of the 2020 Federated Conference on Computer Science and Information Systems. 2020. DOI 10.15439/2020f76.

[19] J Lu, I Holubová. Multi-model databases: a new journey to handle the variety of data, ACM Computing Surveys (CSUR), 2019, vol. 52 (3), 1-38, https://doi.org/10.1145/3323214

[20] R. Angles, C. Gutierrez, Survey of graph database models. ACM Computing Surveys, 2008, 40(1), Article 1.

[21] C. Zhang, L. Jiaheng, X, Pengfei, Y. Cheng. "Unibench: A benchmark for Multi-model Database Management Systems". Performance Evaluation and Benchmarking for the Era of Artificial Intelligence. 2019. P. 7–23. DOI 10.1007/978-3-030-11404-6_2.

[22]

Miroslav Dragić is an Associate Professor at the Faculty of Technology, University of Banja Luka (UNIBL), Bosnia and Herzegovina. He operates within the narrow scientific field of Industrial Engineering and Management. Parallel to his academic work, he has over 25 years of professional industry experience as a production manager, IT developer, and management systems expert

Živko Pavlović is a Full Professor in the Department of Graphic Engineering and Design at the Faculty of Technical Sciences, University of Novi Sad, Serbia. His primary research interests include printing technologies, prepress workflows, color management, and graphic system optimization. Professor Pavlović has extensive industry experience as a production technologist, which complements his academic research.

David Ištoković is an Associate Professor at the Faculty of Engineering (RITEH), University of Rijeka, Croatia. He operates within the Department of Industrial Engineering and Management, where he serves as a member of the Chair of Process Design and the Head of the Laboratory for Intelligent Machines and Machining Systems.