E JOURNAL

Design Automation: Term Definition and Methodical Analysis for Software Selection

Johannes Willkomm[a] · Vincent Weller[a] · Arthur von Bredow[a] · Carsten Putz[a] · Johannes Henrich Schleifenbaum[a]

[a] Chair of Digital Additive Production (DAP) – RWTH Aachen University
Campus-Boulevard 73, 52074 Aachen, Germany

https://doi.org/10.58134/fh-aachen-rte_2025_008

Abstract This paper presents a comprehensive methodology for evaluating and selecting design automation (DA) software tailored to the needs of mechanical engineering applications. As the demands on product properties, life cycles, and variant diversity increase, the integration of advanced technologies such as additive manufacturing (AM) and DA becomes crucial. A thorough literature review was conducted to compile and define key terms related to DA, followed by the creation of a list of relevant software. This software was then analysed and clustered based on key criteria including user interface, application, design scripting, storage method, integration, and sales model. The clustering results were visualized using Venn diagrams and portfolio matrices to provide clear decision-making tools for engineers and companies. The analysis highlights the prevalence of CAD/graphical user interfaces and the growing trend towards cloud-based and subscription-model software. Design scripting emerged as a key differentiator in DA software, offering advanced functionalities that are becoming increasingly important in complex engineering environments. This paper offers a practical guide for selecting the most appropriate DA software, tailored to specific industrial needs, and addresses the current gaps in systematic software evaluation.

Introduction and Motivation

The increasing demands on product properties, shorter product life cycles, and greater variant flexibility present significant challenges in the product development process. Additionally, new emerging technologies like additive manufacturing (AM) offer cost-effective production regardless of part complexity, making them ideal for low-volume or highly customized products [1]. However, these advantages also result in a more intricate and costly design process in addition to the mentioned higher requirements from the application.

One promising solution to streamline and enhance the product development process is design automation (DA). In addition, DA enables the creation of complex, biomorphic structures that can be efficiently produced using AM [2]. This synergy between DA and AM not only accelerates product development but also reduces costs, especially for products with high variant flexibility.

A critical aspect of implementing a design configurator for DA effectively is the selection of appropriate software. Given the diverse applications of DA, ranging from architectural pattern generation to heat exchangers and structural components, the effectiveness and performance of different software solutions can vary significantly.

The evaluation of DA software has been approached from various perspectives, emphasizing their educational, industrial, and functional applications. In the educational domain, studies assessed software based on integration into the design process, computational architecture, and criteria such as availability of student versions, hardware requirements, and costs [3]. For industrial applications, DA software solutions have been categorized by their role in the design process and evaluated through case studies, such as the comparison of Autodesk Fusion 360 and Siemens Solid Edge, which revealed differences in manufacturing methods and output quality. In these evaluations, cloud-based solutions showed advantages in computational speed and variety [4]. In terms of functionality, research highlighted software features like low entry barriers, stand-alone capabilities, and advanced design iteration tools. Studies demonstrated the importance of intuitive user interfaces, robust simulation capabilities, and workflow efficiency, exemplified by the design of an automotive shock absorber using low-complexity DA tools [5] and the standalone nature of Autodesk Fusion 360, which achieved significant time savings through constraint definition [6]. Additionally, categorizations such as "Assisted CAD Modelling" versus "Functional CAD Programming Environments" shed light on the trade-offs between low-code environments and traditional CAD modules [7].

A further focus lies on optimization capabilities, with topology optimization software being evaluated across 70 solutions based on cycles, design weight, and structural performance [8]. Moreover, challenges in DA adoption for mid-sized companies were identified, emphasizing critical criteria such as non-structural constraints, iterative algorithms, and simulation integration [9].

Despite the insights provided by these studies on the differences between various DA software, a systematic guide for software selection remains lacking. Developers rarely publish detailed descriptions of their software's capabilities or underlying algorithms, making quantitative comparisons challenging. As a result, most studies rely on case studies with a limited number of software tools, leaving a gap in comprehensive comparative analyses. To assist users in selecting the right software for their needs, a detailed

examination, categorization, and evaluation of all available DA software is necessary, considering additional features such as fluid dynamics, heat transfer, and specialized tools or plugins.

In the field of computer-aided design numerous terms with overlapping or similar meanings are frequently used for DA. Terms like generative design, algorithmic design, parametric design, and DA lack agreed-upon definitions. To be able to categorize the software, these terms must first be defined. Putz et al. offered an initial categorization and explanation of these terms [10]. The idea of DA has been pursued in architecture for a long time, so that the definition from this research field is also considered in this work. Caetano et al. used the overarching term "computational design" and collected, analyzed, and compared different definitions of parametric, generative, and algorithmic design through a literature review based on architecture studies [11]. Building on this research, we expand the definitions with perspectives from mechanical engineering. This integrated approach aims to provide a comprehensive definition and categorization of these terms in the following section. Following this, we present the methodology and results of the software classification for design automation.

The aim of this paper is to develop a structured framework for the classification and evaluation of DA software, to support a methodical and application-oriented software selection process in the context of mechanical engineering and DA. First the key terms related to DA are defined and then a clustering methodology based on relevant criteria derived from literature and industry practice are defined.

Definitions of the Term Design Automation

The product development process can be simplified to the steps specification of requirements, determination of functions and their structure, developing of initial design concept and developing of detailed design [12]. The authors of this work define design automation as follows. *In design automation, one or more of the methods parametric, algorithmic, and generative design are used to automate the steps conceptualizing, designing, and elaborating of the product development process to create a 3D CAD model of a part. By applying these methods in combination with the right software, design configurators are set up.* In the following, the three methods parametric, algorithmic, and generative design are also defined.

In the context of architecture, parametric design is defined by Caetano et al. as “an approach that describes a design symbolically based on the use of parameters” [11]. This definition is derived from various architectural perspectives, where parametric design is seen as the process of manipulating parameters and the relationships between them to control and generate complex forms automatically [13–19]. In other definitions from architecture, parametric design is described as a dynamic, rule-based approach that leverages algorithmic thinking to define the relationship between design intent and response. By using parameters and constraints, it enables the parallel development of multiple design solutions and fosters flexibility in exploration. The boundaries between parametric and generative design are often fluid, with both concepts frequently used interchangeably or in combination [20–22]. Extending this definition to the field of mechanical engineering, parametric design utilizes parameters to define and manipulate the geometry of a model by establishing rules and relationships between different components, so that the design can generate mechanical engineering parts and improve

the overall design quality [23, 24]. Combining these insights and creating a clear differentiation to the other methods, *we define parametric design as a process where interdependent parameters define the geometry of the design. By assigning discrete values to these parameters, the corresponding geometry is created.*

Algorithmic design, as defined by Caetano et al., involves a direct correlation between the algorithm and the generated model, ensuring traceability. It is considered a subset of generative design, with the focus on algorithm development to achieve specific design goals, often at the expense of producing fewer unexpected results compared to generative design [11]. This definition is supported by architectural perspectives that see algorithmic design as the systematic application of logical principles to generate space and form through rule-based logic, inherent in architectural programs, typologies, and building codes [25]. It often involves the use of scripting languages that allow designers to manipulate code directly, initiating computational procedures that create digital forms [26, 27]. Expanding this definition within the context of engineering, algorithmic design is further described as a process driven by a defined and logical set of rules that handle computational complexity [28]. It involves creating step-by-step procedures that ensure reliability, efficiency, and correctness in design solutions. Data plays a crucial role in this process, serving as both input and output, guided by mathematical rules within the algorithm [25]. This approach is widely used in mechanical engineering to automate and optimize the design process, employing a variety of deterministic and stochastic algorithms to solve complex engineering challenges [29, 30]. *In summary, algorithmic design is a design approach based on manually implemented rules or algorithms, creating a direct and traceable causation between the algorithm and the design elements. This method allows algorithmic design to be used independently of other methods like generative design, offering a tool for generating, optimizing, and refining designs.*

Caetano et al. define generative design as a design paradigm that employs algorithmic descriptions with a higher degree of autonomy compared to parametric design. Generative design-based methods can generate complex outputs from simple algorithmic descriptions, resulting in a non-traceability between the program and the generated designs [11]. This definition aligns with various architectural views where GD is characterized by its capacity to enable indirect interaction between designers and the resulting products through generative systems. It leverages computational power to autonomously develop and evolve complex designs from simple inputs, often mimicking evolutionary processes in virtual environments [31–34]. GD's iterative nature facilitates the exploration of multiple design solutions, requiring designers to select the optimal outcome based on predefined constraints [35, 36]. By closely integrating with algorithmic and parametric design, GD offers an advanced framework for addressing complex design challenges and overcoming traditional limitations [37–39]. Extending the definition from Caetano, GD is viewed as a designer-driven, parametrically constrained process that operates on top of parametric CAD systems. It aims to create novel designs by leveraging modern computing and manufacturing capabilities [40]. Artificial intelligence tools can play a crucial role in generating and optimizing multiple design solutions based on imposed constraints and goals, typically through iterative processes [41]. Recent advancements include the use of topology optimization and cloud computing to produce multiple optimized designs under various conditions [42]. GD further supports the exploration of feasible design options based on user-defined constraints, enhancing conceptual design and product development [43]. Taking this additional information into account, we claim that *generative design is a*

method that autonomously generates multiple geometric variants using techniques like evolutionary algorithms, optimization algorithms, and artificial intelligence. These approaches enable the creation of complex designs, making GD a versatile and powerful tool in contemporary design practices.

Methodology

This section outlines the methodology used to evaluate and cluster relevant design automation software, culminating in a framework to assist companies in selecting appropriate software for their needs. The approach is based on a comprehensive literature review, followed by the identification, clustering, and analysis of design automation software specific to the mechanical engineering industry. The results of this clustering are visualized through Venn diagrams, providing clear insights into the software's capabilities and aiding in decision-making.

The first step involved compiling an exhaustive list of design automation software that meets the functional criteria identified in the literature review. This list was generated through a combination of academic publications, online research, and with the help of AI tools like Google Gemini to identify relevant software providers. Given the varying demands across different industries, the focus was narrowed to proprietary software commonly used in mechanical engineering. Implementation of algorithms in open-source tools like python or matlab, which require extensive programming of the user, were excluded to maintain consistency and relevance for industrial applications. Tyflopoulos et al. already offers a comprehensive list of open source software and plug-ins for topology optimization [8].

Building on the literature for software selection, the clustering was based on six key criteria [44]: user interface, application, design scripting, storage method, integration, and sales model. Each software was evaluated against these criteria, which were deemed relevant for the mechanical engineering sector.

The **user interface** determines how engineers interact with design automation software, directly influencing its usability and adoption within existing workflows. Three primary types of interfaces are defined:

- CAD/Graphical: These interfaces allow users to define parameters such as dimensions, material properties, or load conditions through graphical menus. They align closely with workflows already familiar to many engineers, particularly in CAD environments, which simplifies adoption. Certain geometric regions can be constrained or optimized, ensuring that critical design constraints are respected. This approach is ideal for engineers with limited programming expertise, offering a smooth transition into design automation workflows.
- Visual Scripting: In this approach, users interact with the software through graphical programming “nodes”. Nodes can represent parameters, variables, or geometric operations, and are connected to define workflows and generate geometry. Visual scripting is particularly advantageous for tasks involving repeated operations, as node configurations can be reused across different projects, saving time, which enables rapid iteration and customization of designs while remaining accessible to non-programmers.

- Code Scripting: These allow users to write text-based scripts to define geometries and automate workflows. This method offers the greatest degree of flexibility and customization but requires more advanced programming knowledge.

DA software is evaluated based on its ability to address specific mechanical engineering challenges. These are divided into the following key **applications**:

- Mechanical: This includes the optimization of parts subjected to mechanical loads, ensuring that designs meet performance criteria under stress.
- Fluid Dynamics: Software capable of optimizing designs to manage fluid flow is essential for components like heat exchangers or aerodynamic surfaces.
- Thermal: DA for heat transfer focuses on designing components to handle thermal loads effectively, such as cooling systems or heat sinks.

Each of these applications ensures that DA software is capable of generating designs that meet the functional requirements of various engineering fields. Software limited to optimizing visual or aesthetic aspects is excluded from this analysis.

Design scripting refers to automating geometry creation and design process through rule-based scripting mechanisms, enabling the rapid generation of design alternatives based on changing input parameters. While it overlaps conceptually with visual or code scripting, it focuses on automation logic and system integration rather than interface. Algorithms define the relationships between design requirements and geometric output, ensuring the generated designs meet specified criteria under varying boundary conditions. For example, parameterized sliders or input fields can adjust dimensions or loads, triggering the software to produce updated geometries without requiring manual intervention.

Design scripting functionality often includes the definition of logical or geometrical relationships through algorithms, parameter mapping, and user-controlled input fields (e.g., sliders or input boxes). From an engineering perspective, it is especially valuable for managing product variants, configuring customer-specific requirements, or generating data-driven geometry transformations.

In this study, a software is categorized as offering design scripting if it allows users to:

- Define rules and conditions that govern geometric or structural behavior,
- Generate or update entire designs automatically in response to varying parameters,
- Do so without requiring reentry into the manual design process for each change.

This distinguishes design scripting as a high-level design automation method, which supports both mass customization and parametric engineering workflows. As such, it represents a core enabler for advanced digital product development pipelines in mechanical engineering and beyond. As no external programming interfaces (APIs) or third-party scripting plug-ins are excluded from the study, only scripting environments that are natively embedded within the respective design software are considered compared to the description of CAD design scripting languages from Celani et al. [45]

The **storage method** has significant implications for workflow efficiency, data security, and computational power. Two primary characteristics are evaluated:

- Local: Data is stored and processed on local devices or company servers. This often requires significant investments in hardware infrastructure.

- Cloud-based: Data is processed and stored on remote servers, leveraging cloud infrastructure. Cloud solutions reduce hardware demands, allow real-time collaboration, and facilitate large-scale computational tasks through scalable resources.

Cloud-based solutions are particularly advantageous for distributed teams and projects requiring high computational power. However, concerns about data security and internet dependency must be addressed.

Integration evaluates whether the software can seamlessly extend its capabilities across additional stages of the design-to-manufacturing workflow:

- Standalone: These are designed to perform specific DA tasks without offering additional features like simulation or AM pre-processing.
- Integrated: These combine DA functionality with other critical steps, such as simulation, topology optimization, or AM-specific adaptations like support structure generation. Some solutions also offer API-based integration for custom workflows. This ensures smooth transitions between design and production, reducing time and effort in transferring data between platforms.

The **sales model** of DA software influences its cost-effectiveness, accessibility, and suitability for different organizational needs:

- Subscription-based: Users pay a recurring fee, typically monthly or annually. These often include regular updates and support services.
- One-time purchase: Software is purchased outright, granting indefinite use without recurring costs.
- Pay-per-design: Charges are based on the number of designs generated or processed.

For each software, information was gathered through an extensive review of manufacturer websites, third-party reviews, tutorials, and white papers. Where necessary, direct inquiries were made to the software developers to clarify specific features. Each criterion was then evaluated as "fulfilled," "partially fulfilled," or "not fulfilled." The results were compiled into a clustering table, with criteria visually represented in Venn diagrams using the BioVenn tool [46]. The diagrams illustrate the overlap between software features, with software meeting multiple criteria positioned in overlapping regions.

The methodology is designed to be adaptable across industries and evolving software landscapes. It can be extended by incorporating additional criteria (e.g. AI integration, compliance standards) and scaled through automated tools or integration into decision-support systems.

Software List and Clustering of the Results

This chapter presents the compiled software list and the results of the clustering analysis. The outcome of this work is a decision-making tool for engineers and companies to select the most suitable DA software for their specific applications. Table 1 illustrates the clustering results, providing a comprehensive overview of design automation software relevant to engineers. The table not only showcases the software but also demonstrates the developed clustering methodology, with the criteria used for clustering listed in the top rows. To ensure the continued relevance of this study, the clustering table is maintained

and regularly updated online at dap-aachen.de[1], allowing users to access the latest software landscape and submit suggestions for modifications or additions directly to the author.

Table 1: Tabular summary of the cluster, taking into account all software under consideration and fulfillment of the criteria.

Legend:
● fulfilled
○ not fulfilled
◐ partially fulfilled

		Criteria													
		User Interface			Application				Storage Method		Inte-gration		Sales Model		
Developer	Software name	CAD/graphical	visual scripting	code scripting	fluid dynamics	mechanical	heat transfer	design scripting	cloud-based	local	integrated	stand-alone	subscripting	one-time purchase	pay-per-design
Altair	Inspire	●	○	◐	○	●	○	○	●	●	○	●	●	○	○
	Optistruct	●	○	◐	○	●	○	○	●	●	●	○	●	○	○
Ansys	Mechanical	●	○	◐	○	●	○	○	●	●	○	●	●	●	●
	Discovery	●	○	◐	○	●	○	○	●	●	●	○	●	●	●
	Fluent	●	○	◐	●	○	●	○	●	●	○	●	●	●	●
Autodesk	Fusion360	●	○	◐	●	●	○	○	●	○	●	○	●	○	●
	Inventor	●	○	◐	○	●	○	○	●	●	●	○	●	○	●
BETA	ANSA	●	○	◐	○	●	○	○	○	●	○	●	●	○	○

[1] https://dap-aachen.de/project/design-automation-software-overview

Carbon	CogniCAD	●	○	○	○	●	●	○	●	○	●	○	●	○	●
COMSOL	Multiphysics	●	○	◐	●	●	●	○	○	●	○	●	○	●	○
Dassault Systemes	3DExperience	●	●	◐	●	●	●	●	●	○	●	○	●	○	○
	Abaqus	●	○	◐	○	●	○	○	●	●	○	●	●	○	○
	Catia 3ds	●	●	◐	●	●	○	●	●	●	●	○	●	○	○
	Solidworks	●	○	◐	○	●	○	○	○	●	●	○	●	●	○
Diabatix	ColdStream	●	○	○	○	○	●	○	●	○	○	●	●	○	○
DYNAmore (Ansys)	LS-TaSC	●	○	◐	○	●	○	○	○	●	○	●	●	○	○
ENGYS	HELYX-Adjoint	●	○	○	●	○	○	○	○	●	○	●	○	●	○
FEMTools	DDS	●	○	◐	○	●	○	○	○	●	○	●	●	●	○
FRIENDSHIP SYSTEMS	CAESES	●	○	◐	●	○	○	●	○	●	○	●	●	○	○
Hexagon	MSC Apex	●	○	◐	○	●	○	○	○	●	○	●	●	○	○
	MSC Nastran	●	○	◐	○	●	○	○	●	●	○	●	●	○	○
InfiniteForm, Inc	InfiniteForm	●	○	○	○	●	○	○	●	○	○	●	n/a	n/a	n/a
INTES	PERMAS	●	○	◐	○	●	●	○	○	●	○	●	●	○	○
LimitState	FORM	●	○	○	○	●	○	○	○	●	○	●	●	○	○

Marius Kintel	OpenSCAD	○	○	●	○	○	○	●	○	●	○	●	/	/	/
nTopology	nTop	○	●	◐	●	●	●	●	●	●	○	●	●	○	○
PTC	Creo Parametric	●	○	◐	○	●	○	○	●	●	●	○	●	○	○
Quint	OPTISHAPE-TS	●	○	◐	○	●	○	○	○	●	○	●	●	○	○
Robert McNeel & Associates	Rhino 8	●	●	●	●	●	●	●	○	●	●	○	○	●	○
Salome	Salome 9	●	○	●	○	○	○	●	○	●	●	○	/	/	/
Siemens	NX	●	●	◐	○	●	○	●	○	●	●	○	●	○	○
	SolidEdge	●	○	◐	○	●	○	○	○	●	●	○	●	○	○
Simright	Toptimizer	●	○	○	○	●	○	○	●	○	○	●	●	○	○
Synera	Synera	○	●	●	●	●	●	●	○	●	●	○	●	○	●
ToffeeX	ToffeeX	●	○	○	●	○	●	○	●	○	○	●	●	○	○
trinckle	PARAMATE	○	○	●	○	○	○	●	●	○	●	○	●	○	○
VR&D	GENESIS	●	○	◐	○	●	○	○	○	●	○	●	●	○	○

The clustering analysis reveals several key trends in the DA software landscape. 91 % of the software options provide a CAD or graphical user interface, making it the most common interface type. In contrast, only 17 % of the software offers a visual scripting interface, and code scripting is the least commonly available, offered by just 14 % of the software. In terms of application-specific functionalities, 80 % of the DA software supports applications related to mechanical loads, making it the most widely supported application type. Software supporting fluid dynamics applications is available in 29 % of the cases, while only 29 % of the software includes features for thermal transfer applications, making

it the least common. Design scripting capabilities, which allow for automated design tasks, are provided by 29 % of the software. Regarding storage solutions, 4 % of the software supports cloud storage, whereas 80 % offers local storage options, indicating a preference for locally managed data among the software analyzed. The integration of DA software into existing workflows is split, with 43 % of the software being integrated solutions and 60 % stand-alone solutions. Due to the criteria used, there is no overlap between these categories. Finally, 86 % of the software operates on a subscription-based model, making it the predominant sales model. A one-time purchase option is available for 23 % of the software, while 20 % offer a pay-per-design or use model. Open-source software, such as OpenScad, was excluded from this analysis as it was the only software of its kind considered. A graphical representation of the results using Venn diagrams is shown in Figure 1.

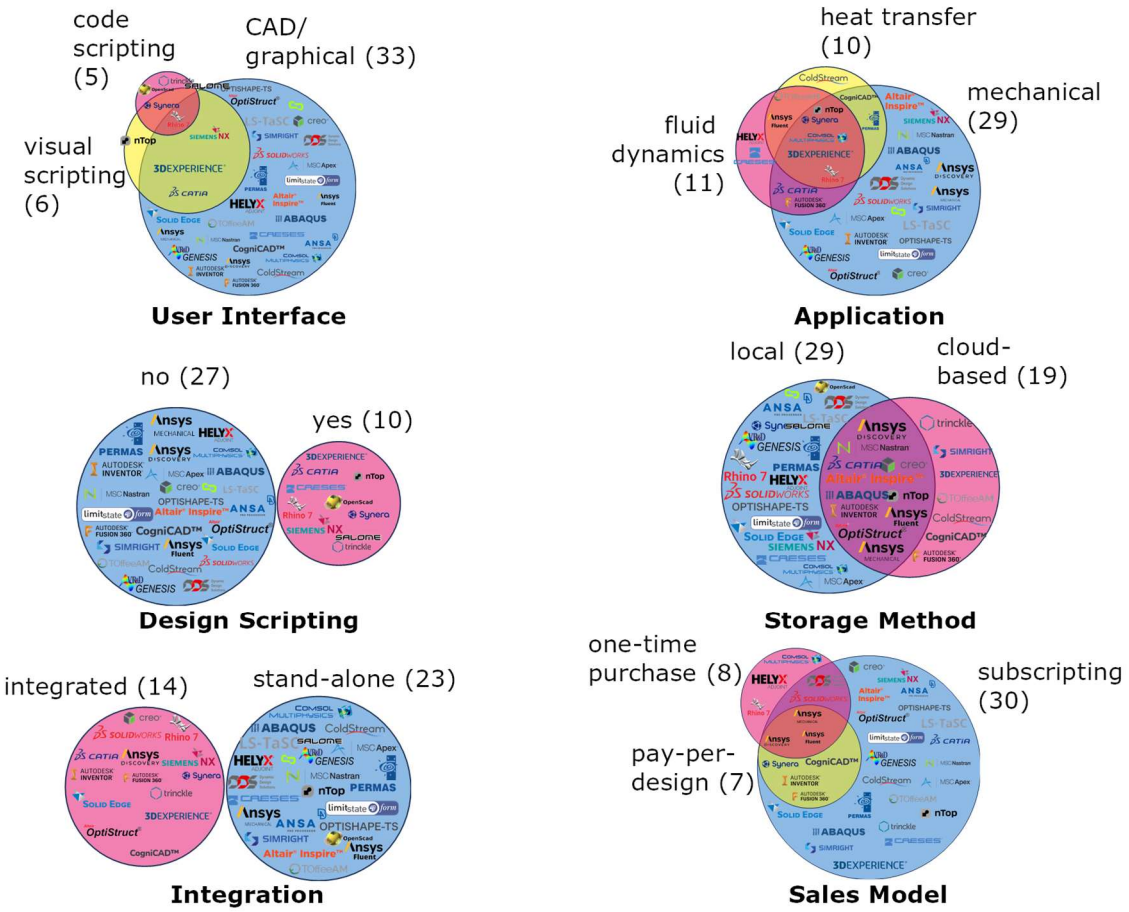

Figure 1: Graphical representation of the software classification results using Venn diagrams (absolute numbers in brackets)

Discussion

The analysis indicates that DA software can meet all the key requirements of engineers and companies, with each criterion being fulfilled by at least one software solution. A notable trend is the strong prevalence of CAD/graphical user interfaces, which suggests that many DA software solutions prioritize ease of use and accessibility by offering familiar interfaces. The mechanical application is the most frequently supported, while fluid

dynamics and thermal transfer applications are also well-represented. The significant presence of cloud-based solutions, offered by 51 % of the software, highlights the growing accessibility of computational power through cloud computing. Subscription models are the most common, reflecting the broader industry trend towards software-as-a-service (SaaS). The equal representation of integrated and stand-alone solutions provides companies with flexibility in choosing software that either focuses solely on DA or includes additional workflow steps.

Design scripting stands out as a distinct and advanced feature within DA software, often supporting a broad range of applications, including mechanical, fluid dynamics, and thermal transfer. This makes design scripting a forward-looking technology, potentially representing the highest level of DA. The dominance of the CAD market by large companies such as Autodesk, Dassault Systèmes, Siemens, Hexagon, and PTC is evident, with these companies controlling 83 % of market revenue (2017) [47]. However, smaller companies have carved out a niche by adopting innovative approaches like design scripting, which is currently supported by only 9 out of 35 analysed software solutions. Historically, design scripting tools are relatively recent developments, with companies like Synera (2018), nTop (2015), and Trinckle (2012) leading the way [48–50]. Larger companies have only recently integrated design scripting tools, with Dassault Systèmes and Siemens implementing them in 2022 and 2020, respectively [51, 52]. This delay could be due to the higher entry barriers associated with design scripting, such as the need for specialized visual or code scripting interfaces, which require additional training and thus present a steeper learning curve [53]. While established software companies may perceive this as a challenging market to enter, smaller companies have leveraged design scripting to differentiate themselves and capture new market segments. Additionally, design scripting appears to be driven more by a technology push than by market demand, as evidenced by the scientific activity surrounding it and the fact that companies like Synera and Trinckle originated from research projects [48, 50].

The clustering results provide engineers and companies with valuable insights for selecting the most appropriate DA software. A suggested procedure for software selection would consider three steps. The process begins with a requirements analysis, where the user identifies which clustering characteristics are essential or desirable. These clusters can then be used to narrow down the list of DA software options by focusing on the most critical features. The Venn diagrams allow users to quickly identify software that meets these criteria. By progressively filtering through the diagrams, starting with the most important feature, the list can be reduced to approximately three software options. This shortlist can be further refined through additional literature review and, if available, comparative studies.

For a more precise selection, the portfolio matrix shown in Figure 2 can be used. This matrix plots software options based on the diversity of variants they support and the complexity of the applications they handle. In this case, complexity can be defined as the number of functions the part should fulfil.

The classification into four categories is derived from the clustering results:

- **Basic DA Software** addresses low variant diversity and low complexity. These tools typically support at least one application and offer basic simulation features.

- **Specialized DA Software** handles high complexity but low variant diversity. These tools must support at least one specific application and include advanced simulation capabilities tailored to dedicated use cases.
- **Extended DA Software** targets scenarios with high variant diversity and lower complexity. These tools support at least the functionality of design scripting, enabling flexible and efficient variant generation.
- **Performance DA Software** supports both high complexity and high variant diversity. These tools include design scripting, multiple application domains, and advanced optimization or simulation tools (e.g., topology optimization or FEM).

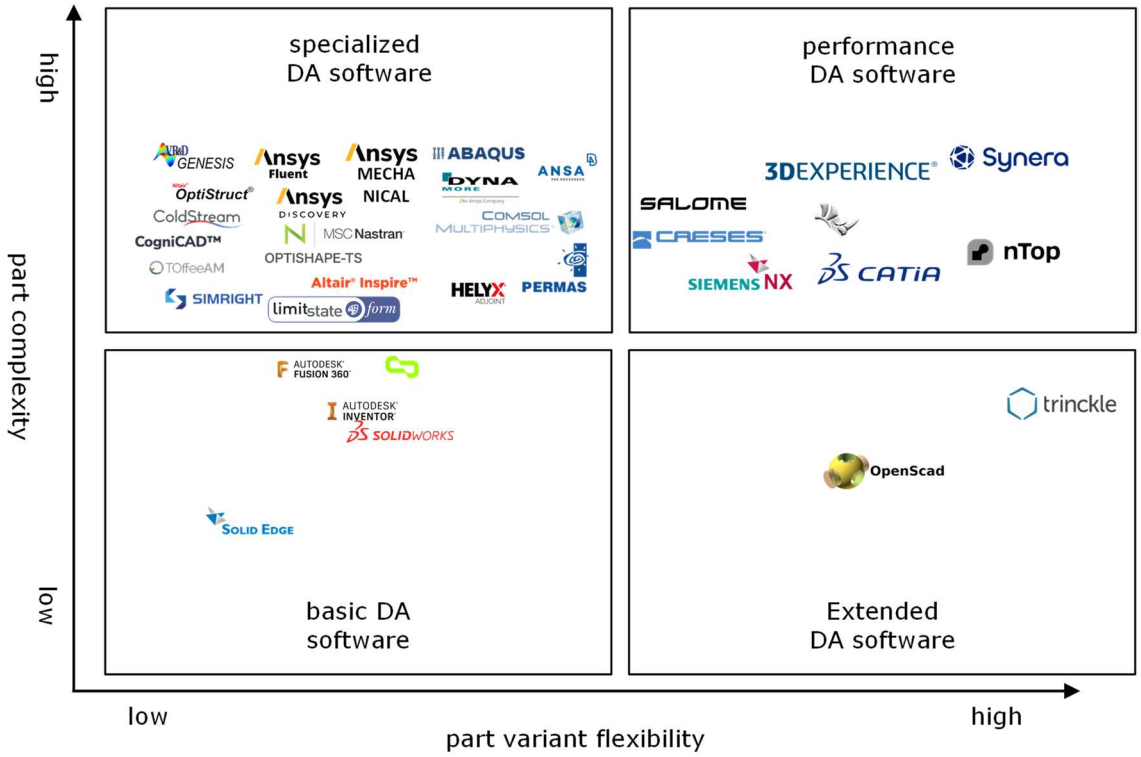

Figure 2: Portfolio matrix for DA software selection

Conclusion and outlook

In summary, the integration of DA with Additive Manufacturing presents a transformative approach to product development, addressing the increasing complexity and customization demands in modern engineering. This paper analyzed and clustered commercial software solutions for DA.

The clustering analysis conducted reveals significant trends within the DA software landscape, emphasizing the predominance of user-friendly CAD interfaces and a notable shift towards cloud-based solutions. The findings indicate that while established companies dominate the market, innovative smaller firms are making strides by incorporating advanced features like design scripting, which enhances flexibility and efficiency in design tasks.

This work proposes a systematic approach for engineers to evaluate their options of DA software based on key criteria such as user interface type, application focus, integration

capabilities, and sales models. By employing the developed clustering methodology and visual aids like Venn diagrams, users choose an appropriate DA software suited their specific needs.

Future work should update the clustering methodology with new software and user feedback, explore applicability across industries, and include non-commercial tools. Developing a software-based clustering approach and enabling re-evaluation during implementation could further enhance the process.

Moreover, emerging technologies such as machine learning-assisted simulation tools (e.g., Ansys SimAI) and generative AI systems are poised to significantly impact the DA software landscape. These technologies may enable predictive, data-driven design automation and further reduce manual input through intelligent design space exploration. Their integration should be closely monitored and incorporated into future evaluations of DA software capabilities.

Literature

[1] G. Moroni, S. Petro, and W. Polini, Geometrical product specification and verification in additive manufacturing, CIRP Annals, vol. 66, no. 1, pp. 157–160, 2017, doi: 10.1016/j.cirp.2017.04.043.

[2] Lithgow, D., Morrison, C., Pexton, G., Panarotto, M., Müller, J. R., Almefelt, L., McLaren, A., Design automation for customised and large-scale additive manufacturing: a case study on custom kayaks, Proceedings of the Design Society: International Conference on Engineering Design. Vol. 1. No. 1., pp. 699–708, 2019, doi: 10.1017/dsi.2019.74.

[3] S. Junk and L. Burkart, Comparison of CAD systems for generative design for use with additive manufacturing, Procedia CIRP, vol. 100, no. 31, pp. 577–582, 2021, doi: 10.1016/j.procir.2021.05.126.

[4] K. Szabó and G. Hegedűs, Brief overview of generative design support software, Design of Machines and Structures, vol. 10, no. 2, pp. 123–132, 2020, doi: 10.32972/dms.2020.023.

[5] M. Pollák, M. Kočiško, and J. Dobránsky, Analysis of software solutions for creating models by a generative design approach, IOP Conf. Ser.: Mater. Sci. Eng., vol. 1199, no. 1, p. 12098, 2021, doi: 10.1088/1757-899X/1199/1/012098.

[6] N. Babel and M. Metzger, Untersuchung künstlicher Intelligenz im Bereich der Konstruktion mit Generativer Design Software. 1258 KB, Hochschule für angewandte Wissenschaften Landshut, Landshut, 2021.

[7] D. Gerhard, T. Köring, and M. Neges, Generative Engineering and Design – A Comparison of Different Approaches to Utilize Artificial Intelligence in CAD Software Tools in *IFIP Advances in Information and Communication Technology*, vol. 667, *Product Lifecycle Management. PLM in Transition Times: The Place of Humans and Transformative Technologies: 19th IFIP WG 5.1 International Conference, PLM 2022, Grenoble, France, July 10–13, 2022, Revised Selected Papers*, F. Noël, F. Nyffenegger, and L. Rivest, Eds., 1st ed., Cham: Springer Nature Switzerland; Imprint Springer, 2023, pp. 206–215.

[8] E. Tyflopoulos and M. Steinert, A Comparative Study of the Application of Different Commercial Software for Topology Optimization, Applied Sciences, vol. 12, no. 2, p. 611, 2022, doi: 10.3390/app12020611.

[9] C. E. Westerveld, Generative Design Recommended actions to smooth the way for production of generative designs with additive manufacturing. Masterarbeit, University of Twente, Twente, 2021.

[10] Putz, C., Willkomm, J., Reich, S., Ziegler, S., Schleifenbaum, J., Digitale Assistenzsysteme für die effiziente Entwicklung einer neuen Produktgeneration, Proceedings of the 19th Rapid. Tech 3D Conference Erfurt, Germany, 2023, doi: 10.3139/9783446479425.016.

[11] I. Caetano, L. Santos, and A. Leitão, Computational design in architecture: Defining parametric, generative, and algorithmic design, Frontiers of Architectural Research, vol. 9, no. 2, pp. 287–300, 2020, doi: 10.1016/j.foar.2019.12.008.

[12] *Design of technical products and systems - Model of product design*, VDI 2221 Blatt 1, VDI Verein Deutscher Ingenieure e.V., 2019-11-00.

[13] Aish, R., Woodbury, R., Transformations on Parametric Design Models: A Case Study on the Sagrada Famialia Columns in *Smart Graphics*5th International symposium on smart graphics, Frauenwörth Cloister, Germany, 2005.

[14] C. Barrios, Transformations on parametric design models: A case study on the Sagrada Familia columns, Computer Aided Architectural Design Futures 2005: Proceedings of the 11th International CAAD Futures Conference held at the Vienna University of Technology, Vienna, Austria, on June 20–22, 2005, 2005, doi: 10.1007/1-4020-3698-1_37.
[15] Elghandour, A., Saleh, A., Aboeineen, O., Elmokadem, A., Using parametric design to optimize building's façade skin to improve indoor daylighting performance, In Proceedings of the 3rd IBPSA-England Conference BSO, pp. 353–361, 2016.
[16] B. Kolarevic, *Architecture in the digital age*: Taylor & francis, 2003.
[17] J. Monedero, Parametric design: a review and some experiences, Automation in construction, vol. 9, no. 4, pp. 369–377, 2000, doi: 10.1016/S0926-5805(99)00020-5.
[18] K. Nassar, W. Thabet, and Y. Beliveau, Building assembly detailing using constraint-based modeling, Automation in construction, vol. 12, no. 4, pp. 365–379, 2003, doi: 10.1016/S0926-5805(02)00090-0.
[19] M. A. Zboinska, Hybrid CAD/E platform supporting exploratory architectural design, Computer-Aided Design, vol. 59, pp. 64–84, 2015, doi: 10.1016/j.cad.2014.08.029.
[20] N. Gu, R. Yu, and P. A. Behbahani, Parametric design: Theoretical development and algorithmic foundation for design generation in architecture, Handbook of the Mathematics of the Arts and Sciences, pp. 1361–1383, 2021, doi: 10.1007/978-3-319-57072-3_8.
[21] W. Jabi, *Parametric design for architecture*: Hachette UK, 2013.
[22] C. Lee, S. Shin, and R. R. Issa, Rationalization of free-form architecture using generative and parametric designs, Buildings, vol. 13, no. 5, p. 1250, 2023, doi: 10.3390/buildings13051250.
[23] Cucos, M. M., Pista, I. M., Ripanu, M. I., Product engineering design enhancing by parameterizing the 3D solid model, MATEC Web of Conferences, 2018, doi: 10.1051/matecconf/201817805011.
[24] F. Hoisl and K. Shea, Three-dimensional labels: a unified approach to labels for a general spatial grammar interpreter, AI EDAM, vol. 27, no. 4, pp. 359–375, 2013, doi: 10.1017/S0890060413000188.
[25] Tedeschi, A., Lombardi, D., *The algorithms-aided design (AAD)*: Le Penseur, 2017.
[26] R. Oxman, Thinking difference: Theories and models of parametric design thinking, Design studies, vol. 52, pp. 4–39, 2017, doi: 10.1016/j.destud.2017.06.001.
[27] Queiroz, N., Dantas, N., Nome, C., Vaz, C., Designing a Building envelope using parametric and algorithmic processes, Proceedings of the 19th Conference of the Iberoamerican Society of Digital Graphics, pp. 797–801, 2015, doi: 10.5151/despro-sigradi2015-sp90284.
[28] T. Hnin, *Getting Started With Algorithmic Design In Architecture in 2024: A Comprehensive Guide.* [Online]. Available: https://www.novatr.com/blog/algorithmic-design-in-architecture#0
[29] B. Cheong, P. Giangrande, X. Zhang, M. Galea, P. Zanchetta, and P. Wheeler, Evolutionary multiobjective optimization of a system-level motor drive design, IEEE Transactions on Industry Applications, vol. 56, no. 6, pp. 6904–6913, 2020, doi: 10.1109/TIA.2020.3016630.
[30] J. O. Agushaka and A. E. Ezugwu, Advanced arithmetic optimization algorithm for solving mechanical engineering design problems, Plos one, vol. 16, no. 8, e0255703, 2021, doi: 10.1371/journal.pone.0255703.

[31] Fischer, T., Herr, C. M., Teaching generative design, Proceedings of the 4th Conference on Generative Art, pp. 147–160, 2001.
[32] Frazer, J., Frazer, J., Liu, X., Tang, M., Janssen, P., Generative and evolutionary techniques for building envelope design, Generative Art 2002, 5th International Conference GA2002, 2002.
[33] J. Krause and U. S. BArch, The Creative Process of Generative Design in Architecture, GA2003 (6th), 2003.
[34] J. McCormack, A. Dorin, and T. Innocent, Generative Design: A Paradigm for Design Research in *Futureground - DRS International Conference 2004*Futureground - DRS International Conference 2004, Melbourne, Australia, 2004.
[35] M. Bernal, J. R. Haymaker, and C. Eastman, On the role of computational support for designers in action, Design studies, vol. 41, pp. 163–182, 2015, doi: 10.1016/j.destud.2015.08.001.
[36] A. Chaszar and S. C. Joyce, Generating freedom: Questions of flexibility in digital design and architectural computation, International Journal of Architectural Computing, vol. 14, no. 2, pp. 167–181, 2016, doi: 10.1177/1478077116638945.
[37] S. Abrishami, J. Goulding, F. P. Rahimian, and A. Ganah, Integration of BIM and generative design to exploit AEC conceptual design innovation, Information Technology in Construction, vol. 19, pp. 350–359, 2014.
[38] F. A. Bukhari, A Hierarchical Evolutionary Algorithmic Design (HEAD) system for generating and evolving building design models, Queensland University of Technology, 2011.
[39] Humppi, H., Österlund, T., Algorithm-aided BIM, Complexity & simplicity–Proceedings of the 34th eCAADe conference, pp. 601–609, 2016.
[40] S. Krish, A practical generative design method, Computer-Aided Design, vol. 43, no. 1, pp. 88–100, 2011, doi: 10.1016/j.cad.2010.09.009.
[41] F. Buonamici, M. Carfagni, R. Furferi, Y. Volpe, and L. Governi, Generative design: an explorative study, Computer-Aided Design and Applications, vol. 18, no. 1, pp. 144–155, 2020, doi: 10.14733/cadaps.2021.144-155.
[42] S. Oh, Y. Jung, S. Kim, I. Lee, and N. Kang, Deep generative design: Integration of topology optimization and generative models, Journal of Mechanical Design, vol. 141, no. 11, p. 111405, 2019, doi: 10.1115/1.4044229.
[43] M. M. Nisar, S. Zia, M. Fenoon, and O. Alquabeh, Generative design of a mechanical pedal, International Journal of Engineering and Management Sciences, vol. 6, no. 1, pp. 48–58, 2021, doi: 10.21791/IJEMS.2021.1.5.
[44] J. Becker, A. Winkelmann, and M. Philipp, Entwicklung eines Referenzvorgehensmodells zur Auswahl und Einführung von Office Suiten, Arbeitsberichte des Instituts für Wirtschaftsinformatik, 2007.
[45] G. Celani and C. E. V. Vaz, CAD scripting and visual programming languages for implementing computational design concepts: A comparison from a pedagogical point of view, International Journal of Architectural Computing, vol. 10, no. 1, pp. 121–137, 2012, doi: 10.1260/1478-0771.10.1.12.
[46] T. Hulsen, J. de Vlieg, and W. Alkema, BioVenn–a web application for the comparison and visualization of biological lists using area-proportional Venn diagrams, BMC genomics, vol. 9, pp. 1–6, 2008, doi: 10.1186/1471-2164-9-488.
[47] Jon Peddie Research, & Business Advantage, *Computer-aided design (CAD) market revenue share worldwide, in 2016, by vendor.* [Online]. Available: https://

web.archive.org/web/20230412082730/https://www.statista.com/statistics/779090/worldwide-cad-market-revenue-share/ (accessed: Aug. 28 2024).
[48] Freie Univerität Berlin, *Gründungsprojekt Trinckle 3D gehört zu den besten Ideen für Start-ups in Deutschland: Die Ausgründung der Freien Universität Berlin wurde vom Bundeswirtschaftsministerium auf der IFA ausgezeichnet / mit Pressefoto.* [Online]. Available: https://web.archive.org/web/20220302040322/https://www.fu-berlin.de/presse/informationen/fup/2012/fup_12_246/index.html (accessed: Nov. 22 2023).
[49] nTop, *Our story: How we got here.* [Online]. Available: https://web.archive.org/web/20231014222506/https://www.ntop.com/company/about-us/ (accessed: Nov. 22 2023).
[50] Synera, *Über uns - Synera: Meilensteine auf dem Weg in eine neue Ära des Engineerings.* [Online]. Available: https://web.archive.org/web/20231122153558/https://de.synera.io/about (accessed: 22.11.23).
[51] Y. Barbier, *NEW CATIA Visual Scripting | R2023x native app, cloud or on premise.* [Online]. Available: https://web.archive.org/web/20230528120921/https://blog.3ds.com/brands/catia/new-catia-visual-scripting-r2023x-native-app-cloud-or-on-premise/ (accessed: Nov. 22 2023).
[52] W. Chanatry, *Algorithmic Modeling coming soon to NX.* [Online]. Available: https://web.archive.org/web/20221204113251/https://blogs.sw.siemens.com/nx-design/algorithmic-modeling-coming-soon-to-nx/ (accessed: Nov. 22 2023).
[53] D. Vannusov, V. Dadonov, and M. Tereshchenko, Organizational features of innovative CAD implementation in existing production systems, MATEC Web Conf., vol. 311, p. 2009, 2020, doi: 10.1051/matecconf/202031102009.

Kontaktangaben

Johannes Willkomm
RWTH Aachen University - Digital Additive Production DAP
Campus Boulevard 30, 52074 Aachen, Germany
E-Mail: johannes.willkomm@dap.rwth-aachen.de
WEB: http://www.dap-aachen.de