applied sciences

MDPI

Article

Adaptive Structural Productivity Modeling and Analytics for Software Projects

Claudio Antonelli [1,*] and Maria Luisa Villani [2,*]

[1] Department of Engineering Sciences, Università degli Studi Guglielmo Marconi, Via Plinio 44, 00193 Rome, Italy
[2] Italian National Agency for New Technologies, Energy and Sustainable Economic Development (ENEA), Via Anguillarese 301, 00123 Rome, Italy
* Correspondence: claudio.antonelli.it@gmail.com (C.A.); marialuisa.villani@enea.it (M.L.V.)

Abstract

Productivity evaluation in industrial software projects is still predominantly based on aggregate Function Point-to-effort ratios. Such an approach overlooks the internal structural and functional heterogeneity of software projects, potentially leading to misleading comparisons between systems characterized by similar functional sizes but markedly different structural complexities. This work investigates the hypothesis that projects with similar functional structures, within the same application domain, tend to exhibit comparable productivity behavior. For this purpose, an adaptive and structurally informed productivity assessment approach is proposed, based on a Structural Productivity Index that integrates domain-specific productivity information with project functional composition. The proposed model is validated through real-world and synthetic project datasets, combined with consistency indicators and clustering-based analyses to examine its behavior under different conditions and its capability to distinguish regular productivity patterns from structurally anomalous projects. Experimental results suggest that the proposed adaptive productivity modeling approach provides a more coherent interpretation of productivity estimates compared with traditional productivity metrics. The proposed structural representation provides an interpretable basis for future AI-based productivity analytics.

Keywords: software project productivity; function point analysis; functional size measurement; structural analytics; domain adaptive modeling

Academic Editor: Christos Bouras

Received: 28 July 2026
Revised: 30 August 2026
Accepted: 7 September 2026
Published: 10 September 2026

1. Introduction

Measuring productivity in software development projects is a central topic in software engineering, with implications for project planning, benchmarking, supplier evaluation, and contractual management [1,2]. In industrial practice, productivity is often treated as a static comparison indicator: an average value, derived from benchmarks or a company baseline, applied indiscriminately to projects that, despite having similar sizes, present different functional characteristics. Consequently, heterogeneous projects are compared as if they were structurally equivalent, with the risk of rewarding simpler ones and penalizing more complex ones.

Indeed, despite the emergence of data-driven estimation techniques, research on software productivity evaluation generally continues to rely on static Function Point-

based models [3]. These methods estimate software productivity by quantifying the functional size of a system and relating it to the labor effort required for software development. Standardized functional size measurement methods, such as IFPUG [4], COSMIC [5], and NESMA [6], provide formalized approaches for measuring software functionality and deriving these productivity indicators.

This paper proposes an adaptive model that extends traditional aggregate productivity evaluation. The model does not introduce a new functional metric nor a new prediction model. Instead, it reinterprets functional measurement from a structural perspective, shifting productivity from a purely aggregate statistical quantity to a property emerging from the internal functional structure of the software project. The proposed model integrates the Structural Index of the project (i.e., how the project is composed) with the Productivity Profile of a reference baseline (i.e., how individual components influence productivity). Their combination produces a Structural Productivity Index, which enables structural and productive comparability between projects within the same application domain. The proposed approach is adaptive in the sense that the productivity model is calibrated from domain-specific baseline projects, yielding productivity profiles tailored to different software application domains.

The model is validated through real-world and synthetic project datasets and an experimental pipeline, publicly available in [7], combined with consistency indicators and statistical-based analyses, to assess its behavior under different experimental conditions and its capability to distinguish regular productivity patterns from structurally anomalous projects. Experimental results suggest that the proposed adaptive productivity modeling approach provides a more structurally informed and context-aware interpretation of productivity estimates than traditional aggregate productivity metrics alone. Furthermore, the proposed structural representation offers an analytical layer that may complement purely data-driven AI approaches by explicitly incorporating knowledge about the functional composition of software projects.

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 introduces the background concepts and the proposed analysis model. Section 4 describes the experimental evaluation and results. Section 5 discusses the findings in relation to the research hypotheses and previous studies, examines their implications for industrial productivity assessment, and addresses the main threats to validity. Finally, Section 6 summarizes the conclusions and identifies future research directions.

2. Related Work

The related literature encompasses Function Point-based productivity and effort estimation approaches, as well as AI-based and data-driven estimation methods that learn productivity patterns from historical project data. Both perspectives are relevant for positioning the proposed approach and are discussed in the following sections, after a brief overview of the relevant background concepts.

2.1. Background on Function Point Analysis and Productivity

IFPUG Function Point Analysis is the most widely used international functional measurement standard, also formalized as ISO/IEC 20926 [8]. The IFPUG metric classifies software functions into five fundamental component classes, listed in Table 1.

https://doi.org/10.3390/app16189000

Table 1. IFPUG functional component classes.

Abbreviation	Name	Description
ILF	Internal Logical File	Internal logical files managed by the system
EIF	External Interface File	External logical files used by the system
EI	External Input	Data input functionalities
EO	External Output	Processed output functionalities
EQ	External Inquiry	Query-only functionalities

A functional measurement produces two complementary outcomes: the decomposition of the software project into functional components, each associated with a specific class, and an aggregate scalar value expressed in Function Points, representing the overall functional size of the delivered functionalities.

A common definition of productivity, here named Raw Productivity, is the ratio between the functional size delivered by the project and the expended effort:

$$P_R = \frac{FP}{Effort} \quad (1)$$

where *FP* denotes the total functional size expressed in Function Points and *Effort* denotes the effort measured in person-days. By construction, Raw Productivity represents the functional production rate without preserving the contribution of individual functional components within the project structure, making purely numerical productivity comparisons potentially less informative.

2.2. Function Points, Effort and Productivity

Software productivity measurement and effort estimation based on Function Point Analysis (FPA) have been extensively investigated in the literature.

Previous studies have mainly focused on the relationship between functional size and development effort, the predictive capability of Function Points, and the statistical properties of the underlying functional components (a review is presented by Vo et al. [9]). Several authors have analyzed the independence of the five IFPUG basic functional component categories and the possibility of simplifying Function Point counting without significantly affecting estimation accuracy [10]. These studies generally assessed Function Points as predictors of development effort and explored whether subsets of functional components could provide estimation performance comparable to aggregate Function Point measures [11]. The research question addressed in this work differs from these studies in that the focus is not on how Function Points can be used to predict development effort but rather on whether the distribution of IFPUG functional components is associated with distinct productivity patterns within application domains.

At the same time, previous research has also questioned the exclusive use of aggregate Function Point measures. It has been argued that the individual functional component counts may provide a richer characterization of software systems when considered as a vector of measures rather than being combined into a single size indicator [12]. This perspective motivates the analysis of the internal functional composition of software projects rather than relying exclusively on aggregate values.

A complementary perspective emerges from empirical research conducted in industrial settings. An empirical case study conducted within a large financial services organization [13] showed that productivity is influenced not only by functional size but also by application characteristics, technology platforms, organizational conditions, and human factors. These findings reinforce the importance of interpreting productivity within an appropriate project and application context.

The research question investigated in this work is therefore motivated by the combination of these two perspectives: the preservation of the information provided by individual IFPUG functional components and the interpretation of productivity within the corresponding application domain.

Recent studies have confirmed that traditional Function Points provide adequate estimates for software projects but exhibit a progressive reduction in accuracy as project complexity grows. The study by Lavazza et al. [14] explored simplified functional measures, with the aim of reducing measurement complexity while preserving estimation capability. However, the work shows that neither traditional nor simplified Function Point approaches effectively capture software complexity, especially in heterogeneous and complex projects. The authors conclude that improving productivity and effort estimation requires complementary representations of software complexity beyond aggregate functional size measures. Other works [15] highlight how effort estimation models, including those based on regressions and machine learning techniques, use Function Points primarily as a scalar variable, without exploiting the underlying structural information.

2.3. AI-Based Estimation

To address weaknesses of traditional productivity models, recent research has investigated AI-based estimation techniques, including neural networks, regression trees, and locality-aware machine-learning methods, mainly to improve estimation accuracy through data-driven learning from historical project datasets [16]. Systematic reviews indicate that different techniques exhibit different strengths and weaknesses depending on the estimation context, with no single approach consistently outperforming the others [17,18]. Function Points have also been used in data-driven effort estimation models. In [19], the authors show that different learning techniques can derive different estimation capabilities from the same functional-size information, highlighting the distinction between functional measurement and the estimation algorithm.

More recent research has investigated ensemble estimation techniques. Cabral et al. [20] found that ensemble approaches generally outperform individual models, although no single strategy appears universally superior, confirming that predictive performance remains dependent on model configuration and estimation context.

The study [16] also emphasizes limitations of the analyzed AI-based approaches, particularly their dependence on high-quality historical datasets and the growing complexity of optimization-oriented prediction models. Within this research direction, Azzeh et al. [21] proposed a locality-based productivity prediction approach for Use Case Point estimation, where software projects are partitioned into homogeneous groups through environmental factors and clustering techniques to calibrate productivity estimation from similar historical cases.

In contrast to purely data-driven approaches, the proposed model addresses adaptability through an explicit structural characterization of software functionality, leveraging the IFPUG representation. The approach does not rely exclusively on aggregate functional size or on learning productivity patterns directly from historical observations, but it explicitly preserves the contribution of different functional component categories, providing a context-aware representation of productivity through domain-specific functional weights.

To the best of our knowledge, in previous works, less attention has been devoted to exploiting the internal composition of functional measurements as an explicit representation for software productivity analysis.

Prediction accuracy and structural interpretation represent different analytical objectives. Machine-learning models may capture complex relationships and provide

accurate estimates, whereas the present work focuses on deriving interpretable contributions of individual functional components to productivity. These two perspectives can therefore be regarded as complementary. Accordingly, the statistical and analytical techniques adopted in the experimental pipeline serve to derive and assess these relationships. More advanced statistical or machine-learning techniques may complement or extend the proposed approach, provided that such interpretability is preserved.

3. Model Formulation

This section presents the proposed modeling approach, which builds upon the standard IFPUG framework and extends traditional productivity evaluation by explicitly incorporating the structural composition of software functionality.

The approach is organized as a sequence of progressively refined productivity representations. First, a Productivity Profile is derived from a baseline dataset to characterize the contribution of different functional component categories to observed productivity. This profile is then applied to derive a baseline-referenced productivity value associated with the functional composition of individual software projects. Finally, project size is separated from functional composition through a normalized structural representation, leading to the definition of the Structural Productivity Index, a scale-independent indicator that enables structural comparisons across projects of different sizes.

3.1. The IFPUG Metric-Based Expected Productivity

To explicitly account for the functional composition of software projects, productivity is reformulated as a linear combination of the individual functional component classes defined by the IFPUG model. The linear formulation is adopted as an interpretable first-order approximation that enables estimation of the weight of each functional component category to the observed productivity.

A baseline set of completed projects is represented through a matrix X, whose rows correspond to projects and whose columns contain the number of functional components belonging to each IFPUG category (EI, EO, EQ, ILF, EIF). The observed raw productivity values of the baseline projects are collected in the vector P_R. Under the assumption that each functional component category contributes differently to productivity, the relationship between project composition and observed productivity is modeled through a multivariate linear regression:

$$P_R = X \cdot \beta, \tag{2}$$

and the coefficient vector is estimated using ordinary least squares:

$$\beta = (X^T \cdot X)^{-1} \cdot X^T \cdot P_R. \tag{3}$$

The regression model is intentionally formulated without an intercept term so that the estimated productivity depends exclusively on the contribution of the functional components included in the project. Each row of the baseline matrix X corresponds to the functional composition vector of an individual software project. In the following, such a vector is denoted by n when referring to a single project.

Definition 1. *Productivity Profile. Let = (β_{EI}, β_{EO}, β_{EQ}, β_{ILF}, β_{EIF}) be the coefficient vector estimated from the baseline projects through the regression model. The vector β is defined as the Productivity Profile of the baseline dataset. Each coefficient β_i represents the estimated weight associated with the corresponding functional component category within the adopted linear model.*

https://doi.org/10.3390/app16189000

Under the linear approximation adopted in this work, the Productivity Profile can also be interpreted as an estimate of the productivity gradient in the functional space:

$$\nabla P_R = \left(\frac{\partial P_R}{\partial EI}, \frac{\partial P_R}{\partial EO}, \frac{\partial P_R}{\partial EQ}, \frac{\partial P_R}{\partial ILF}, \frac{\partial P_R}{\partial EIF}\right) \approx (\beta_{EI}, \beta_{EO}, \beta_{EQ}, \beta_{ILF}, \beta_{EIF}). \quad (4)$$

Given a software project characterized by the functional composition vector:

$$n = (n_{EI}, n_{EO}, n_{EQ}, n_{ILF}, n_{EIF}), \quad (5)$$

the Productivity Profile can be applied to estimate the productivity expected from its functional structure.

Definition 2. *Expected Productivity. Let n be the functional composition vector as in (5). The Expected Productivity is defined as*

$$P_E = \beta \cdot n \quad (6)$$

To be noted that the term Expected Productivity refers to a baseline-conditioned productivity reference and not a prediction of the project's observed Raw Productivity.

3.2. The Structural Productivity Model

The Expected Productivity provides an estimate of the productivity associated with a given project configuration but remains influenced by the absolute size of the project. Therefore, projects sharing a similar functional structure but differing significantly in scale may exhibit different expected productivity values. To isolate the effect of functional composition from project size, a normalized representation of the project structure is introduced.

Definition 3. *Structural Index. Let N be the total number of functional components in the project. The Structural Index of component k is defined as*

$$S_k = \frac{n_k}{N} \quad (7)$$

where n_k is the number of functional components of type k. The Structural Index vector is

$$S = (S_{EI}, S_{EO}, S_{EQ}, S_{ILF}, S_{EIF}) \quad (8)$$

where the sum of the vector's elements is equal to 1.

The Structural Index defines a normalized vector representation of the functional composition of a software project. Together with the Productivity Profile, it establishes a vector space in which each project is represented by its functional composition and each dimension corresponds to one IFPUG functional component category.

The Structural Productivity Index is obtained by combining these two vector representations through the scalar product.

Definition 4. *Structural Productivity Index. Given the Productivity Profile β and the Structural Index vector S, the Structural Productivity Index is defined as follows:*

$$P_S = \beta \cdot S \quad (9)$$

The Structural Productivity Index provides a scale-independent characterization of how the project's functional composition productivity profile is associated with the project's normalized functional composition, according to the baseline coefficients.

3.3. Structural Consistency Assessment

To support structural consistency assessment, i.e., to quantify the degree of similarity between the project and the baseline population, a distance-based indicator is introduced.

Definition 5. *Structural Distance. Let $S_{Project}$ and $S_{Baseline}$ denote the Structural Index vectors of a project and of the baseline dataset, respectively. The Structural Distance is defined as the Euclidean distance between the two vectors:*

$$D_S = \left|\left|S_{Project} - S_{Baseline}\right|\right|_2 \quad (10)$$

Lower values indicate stronger structural alignment with the baseline, while higher values reveal increasingly atypical functional compositions. Since the Structural Index is normalized, the measure is scale-independent and captures only structural differences.

The Structural Distance naturally defines a structural deviation measure, expressing the degree to which a project deviates from the functional composition of the reference baseline. Based on the empirical distribution of Structural Distance values within the baseline dataset, projects are assigned to one of three consistency classes using the 40th and 80th percentile thresholds (P40–P80):

- High Consistency: strong structural similarity with the baseline population.
- Medium Consistency: moderate deviations compatible with the normal variability of the domain.
- Low Consistency: significant structural deviations suggesting an atypical or anomalous project configuration.

3.4. Model Application Procedure

The proposed model can be applied to a software project through the following sequence:

1. Define the reference domain and baseline. Identify the application domain of the project and select a baseline of completed projects representative of the same functional context.
2. Estimate the Productivity Profile. Using the baseline projects, estimate the domain-specific Productivity Profile β according to Equations (2)–(4). The resulting coefficients represent the contribution of the IFPUG functional component categories within the reference domain.
3. Represent the project functional composition. For the project under analysis, collect the EI, EO, EQ, ILF, and EIF component counts and derive its functional composition vector.
4. Compute the baseline-derived productivity and structural indicators. Apply the domain-specific Productivity Profile to obtain the Expected Productivity according to Equation (6). Normalize the functional composition through the Structural Index defined in Equations (7) and (8) and combine it with the Productivity Profile to obtain the Structural Productivity Index according to Equation (9).
5. Compute the Structural Distance. Compare the Structural Index of the project with the reference baseline according to Equation (10). The resulting Structural Distance quantifies the degree of structural similarity between the project and the reference domain.
6. Assess structural consistency. Compare the Structural Distance with the percentile thresholds derived from the baseline distribution to assign the project to the corresponding High, Medium, or Low Consistency class. The resulting indicators can then be used to interpret the observed project productivity in relation to its structural position within the reference domain.

https://doi.org/10.3390/app16189000

4. Experimental Evaluation

The experimental evaluation aims to verify three key hypotheses underlying the proposed model:

H1. *Domain-specific productivity profiles: different application domains are characterized by distinct Productivity Profiles, indicating that functional component categories contribute differently to productivity depending on the domain.*

H2. *Structural comparability: projects structurally aligned with the reference domain exhibit more consistent baseline-derived productivity characteristics than structurally atypical projects.*

H3. *Discriminative capability: structural and productivity indicators provide additional discriminative information beyond aggregate Raw Productivity. This allows projects with similar observed productivity but different functional structural distances from the domain baseline to be distinguished, as well as supporting the identification of projects that deviate from the domain baseline.*

The evaluation process combines real-world and synthetic datasets with data analytics techniques, including clustering and consistency-based analyses, to examine the behavior of the proposed model.

To support the experimental evaluation, a prototype implementation was developed in Python v3.13.15. The architecture comprises a data layer for dataset management and SQLite-based persistence, a processing layer implementing the proposed productivity and structural analysis procedures, and a presentation layer based on Google Colab for result visualization and reporting. The implementation, datasets, and experimental artifacts are publicly available in the project repository [7].

The validation strategy is based on four complementary experimental settings:

S1. Domain Profile Validation. Baseline profile analysis across three application domains;
S2. Discriminative Validation. Consistent versus Outlier projects comparison;
S3. Structural Discrimination Analysis against Raw Productivity. To assess the incremental structural information provided by the proposed model with respect to conventional Raw Productivity.
S4. Structural Stress-Test Analysis. Synthetic stress testing through Standard, Moderate Structural Shift, Mismatch, and Extreme scenarios.

4.1. Dataset Design

4.1.1. Real-Projects Dataset

The experimental evaluation was conducted across three distinct application domains, each representing a relatively homogeneous functional context. The baseline dataset consists of real software projects collected from different industrial sources and subsequently anonymized and aggregated to preserve confidentiality.

This diversity of domains was intended to assess the applicability of the proposed model under different functional configurations while maintaining domain-level consistency. Table 2 summarizes the composition of the baseline dataset. The Billing domain comprises systems supporting service billing processes, including consumption data acquisition, tariff calculation, and invoice generation. The Data Platform domain includes systems dedicated to data integration, transformation, and quality assurance.

The Integration Platform domain encompasses systems responsible for orchestrating processes and facilitating information exchange across heterogeneous services and applications.

Table 2. Dataset composed of real projects.

Domain	N. Projects	Effort	Function Points	Effort (Median)	Effort (IRQ)	Function Points (Median)	Function Points (IRQ)	Raw Productivity (Median)	Raw Productivity (IRQ)
Billing	62	1759	2535	20.1	20.5	33.0	19.8	1.7	0.8
Data Platform	67	1558	3222	16.5	22.9	39.0	36.5	2.2	1.3
Integration Platform	78	1474	2433	13.9	13.2	26.0	22.0	1.7	0.6

The baseline datasets provide the reference functional contexts used to derive the domain-specific Productivity Profiles and Structural Indices. The descriptive statistics show different distributional characteristics across the three application domains. Data Platform exhibits the greatest variability in functional size and Raw Productivity, while Integration shows more compact distributions, particularly for Effort and Raw Productivity. Billing presents an intermediate pattern, with greater variability in Effort but a more concentrated distribution of functional size compared with Data Platform. These differences highlight the heterogeneous functional and productivity characteristics of the datasets and support the domain-specific organization adopted in the analysis.

To evaluate the behavior of the proposed model on previously unseen projects, a subset of approximately 25% of the available projects was excluded from baseline construction and reserved for testing. The resulting test sets comprise 16 projects for the Billing domain, 17 for Data Platform, and 21 for Integration Platform, with a balanced distribution between consistent and outlier cases.

As shown in Figure 1, in each boxplot, the central line represents the median, while the box represents the interquartile range (Q1–Q3). The whiskers extend to the most extreme observations within 1.5 times the IQR. Values beyond the whiskers are not displayed to improve graphical readability but are retained in the dataset and included in the descriptive statistics.

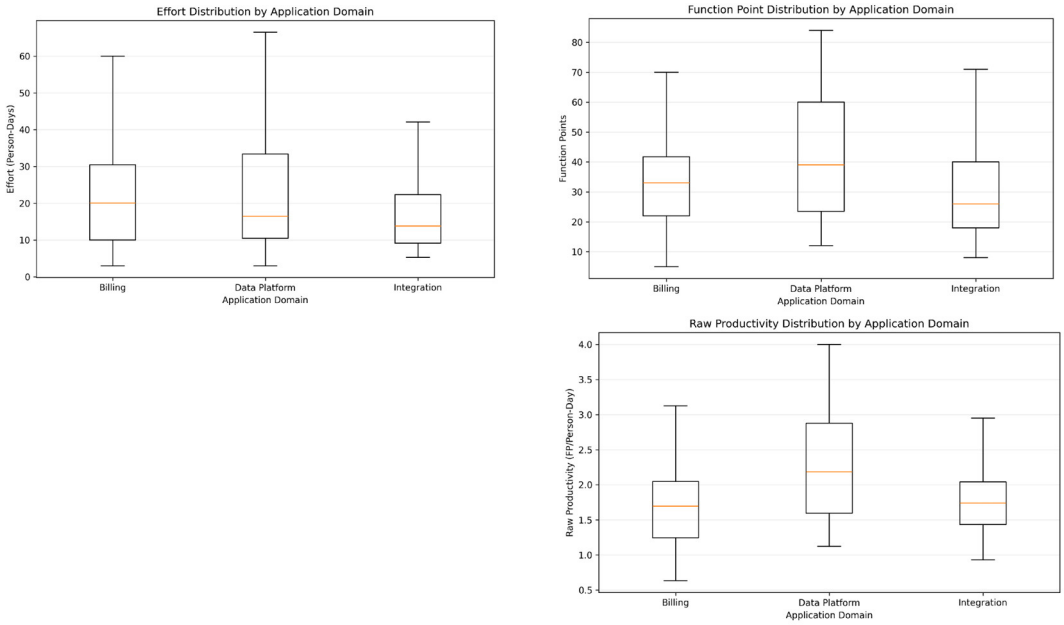


Figure 1. Distributions of Effort, Function Points, and Raw Productivity across the application domains.

4.1.2. Structural Discrimination Dataset

The Structural Discrimination Dataset consists of 203 real software projects belonging to the Digital application domain. This domain includes customer-facing digital

applications and services through which users interact directly with business services, including self-service and digital purchase processes. The projects were collected from industrial sources and subsequently anonymized and aggregated to preserve confidentiality.

Projects were selected within the same application domain to ensure a comparable application context for evaluating whether the proposed structural representation provides additional discriminative information beyond aggregate Raw Productivity. However, no preliminary filtering based on functional composition, structural similarity, or productivity was applied, thus avoiding any selection that could artificially favor the proposed method over Raw Productivity.

Its main descriptive characteristics in terms of Effort, Function Points, and Raw Productivity are reported in Table 3.

Table 3. Structural Discrimination Dataset composed of real projects.

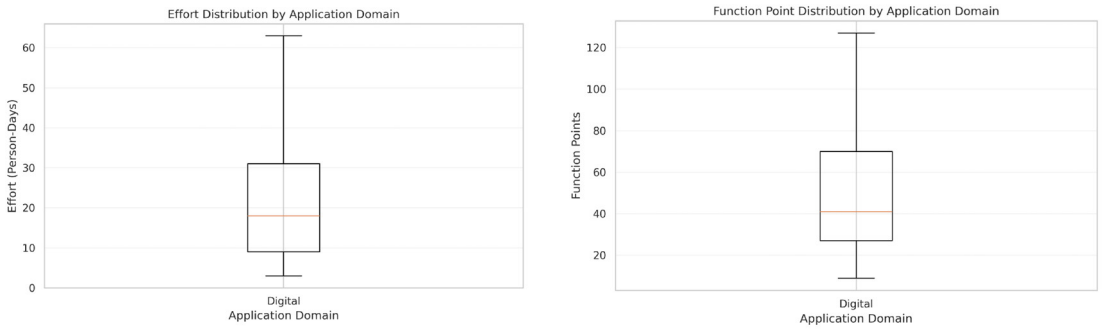
Domain	N. Projects	Effort	Function Points	Effort (Median)	Effort (IRQ)	Function Points (Median)	Function Points (IRQ)	Raw Productivity (Median)	Raw Productivity (IRQ)
Digital	203	5419	11,346	18.0	22.0	41.0	43.0	2.4	1.1

The descriptive statistics show that the Digital dataset includes projects with heterogeneous functional sizes and observed productivity levels. The median project size is 41 FP, with an interquartile range of 43 FP, while the median Effort is 18, with an interquartile range of 22. Raw Productivity has a median value of 2.43 FP/Effort and an interquartile range of 1.14. These distributions provide a sufficiently varied real-project population for evaluating structural differences among projects within the same application domain.

As shown in Figure 2, in each boxplot, the central line represents the median, while the box represents the interquartile range (Q1–Q3). The whiskers extend to the most extreme observations within 1.5 times the IQR. Values beyond the whiskers are not displayed to improve graphical readability but are retained in the dataset and included in the descriptive statistics.

To reduce the dependence of the analysis on a single baseline composition, three random experimental configurations were generated from the Digital project population. In each configuration, 100 projects were assigned to the baseline subset and 103 different projects to the test subset. The baseline projects were used to derive the domain Productivity Profile and structural reference, while the test projects were evaluated against the corresponding baseline.

The same Structural Discrimination Analysis was independently applied to each configuration. This repeated design allows the observed discrimination patterns to be examined under different baseline compositions, rather than relying on a single random partition of the Digital dataset.



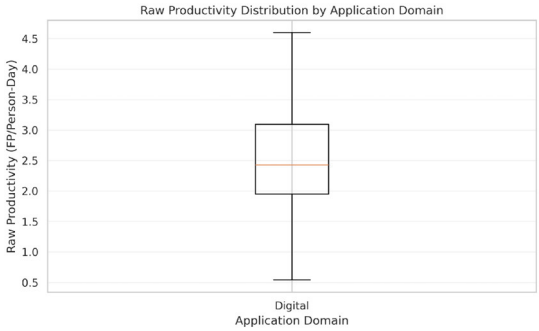


Figure 2. Distributions of Effort, Function Points, and Raw Productivity in the Digital Structural Discrimination Dataset.

4.1.3 Synthetic Stress-Test Datasets

To examine the model response under controlled structural conditions, four synthetic datasets were generated. The datasets were designed to represent progressively different scenarios, ranging from projects fully consistent with the baseline domain to projects exhibiting structural and productivity characteristics outside the calibration space. Table 4 summarizes the composition and purpose of the stress test datasets.

Table 4. Stress test dataset composition.

Type	N. Projects	Validation Purpose
Standard	50	Verify model stability under expected operating conditions
Moderate	80	Introduce moderate structural changes while preserving plausible functional compositions
Mismatch	80	Evaluate sensitivity to productivity–structure inconsistencies
Extreme	80	Assess model behavior outside the calibration domain

The synthetic datasets were generated by controlling functional composition, project size, and productivity behavior. Standard projects were generated around a reference IFPUG functional signature with limited structural variability and productivity values constrained around the reference behavior. Moderate Structural Shift projects were generated by introducing controlled deviations from the reference functional composition while preserving structurally plausible IFPUG configurations and productivity behavior consistent with the reference domain. Mismatch projects preserved a structurally coherent functional composition while productivity was generated independently of the structure, thereby introducing controlled structure–productivity inconsistencies. Extreme projects were generated with substantially higher structural dispersion and an intentionally extreme productivity profile to reproduce conditions outside the baseline calibration space. Project size was varied within predefined ranges in all scenarios. The complete generation algorithms, including parameter ranges, constraints, and acceptance criteria, are publicly available in the repository [7].

Since the proposed Productivity Profiles are derived from regression-based productivity weights, additional analyses were conducted before evaluating the research hypotheses. These analyses include a multicollinearity assessment and a bootstrap-based stability analysis of the estimated productivity weights. Their objective is to assess the robustness of the proposed approach and to provide additional support for the use of regression-based productivity weights in the analyzed datasets and application domains.

4.2. Multicollinearity Assessment

Given the documented correlations among Function Point component categories reported in the literature, multicollinearity has been assessed to verify that dependencies

among the components do not compromise the reliability of the estimated weights in the analyzed project datasets. In Table 5, the VIF values are reported for each IFPUG functional component. These have been computed as

$$VIF_i = \frac{1}{1 - R_i^2} \tag{11}$$

where R_i^2 is the coefficient of determination obtained by regressing the i-th component against all remaining components. To further examine the behavior of the regression-based formulation, the multicollinearity analysis was extended to two additional application domains, namely Sales (applications supporting sales channels and commercial processes) and Digital (applications supporting customer-facing digital channels, including direct interaction, self-service, and online purchasing).

The corresponding VIF results are therefore reported across five domains in Table 5, while the experimental evaluation remains focused on the three baseline domains, for which larger and more representative project populations were available.

Table 5. VIF analysis results for the IFPUG functional components by application domain.

Domain	ILF	EIF	EI	EQ	EO
Billing	1.10	1.04	2.59	1.13	2.60
Data Platform	2.13	1.03	2.61	1.08	1.59
Integration	N/A	N/A	3.06	1.28	2.87
Sales	1.16	1.00	1.93	1.60	2.07
Digital	1.07	N/A	1.63	1.42	1.64

All VIF values remain below 3.1, indicating no evidence of problematic multicollinearity, as commonly adopted thresholds range between 5 and 10 for identifying potentially problematic predictor dependencies. For the Integration domain, VIF values could not be computed for ILF and EIF because these components exhibit near-zero variance, reflecting the absence of local data storage functions in the analyzed projects. Overall, the results indicate that multicollinearity is unlikely to substantially affect the estimation and interpretation of the domain-specific productivity weights.

To be noted that this finding does not imply that the IFPUG functional components are independent in an absolute sense, nor is such an assumption required by the proposed model. Dependencies among components may represent inherent characteristics of a specific application domain. For instance, in the Integration domain, the frequent association of input and output functionalities is consistent with systems designed to acquire, process, orchestrate, and exchange information across multiple applications and services.

4.3. Results

4.3.1. Results for S1: Domain Profile Validation

To evaluate H1, the baseline domains are compared to identify similarities and differences in their structural and productivity signatures.

Figure 3a illustrates the Structural Index distributions for the three application domains. Billing is characterized by a balanced functional composition, with EI and EO representing the dominant component categories, complemented by a significant presence of ILF and a marginal EIF contribution. Data Platform exhibits a similar overall structure but with a higher incidence of data-related components, particularly ILF and EIF. In contrast, Integration Platform presents a strongly specialized configuration dominated by input-output functionalities (EI and EO), with EQ playing a secondary role and no contribution from data-oriented components (ILF and EIF). These results indicate that the three domains are characterized by distinct functional structures.

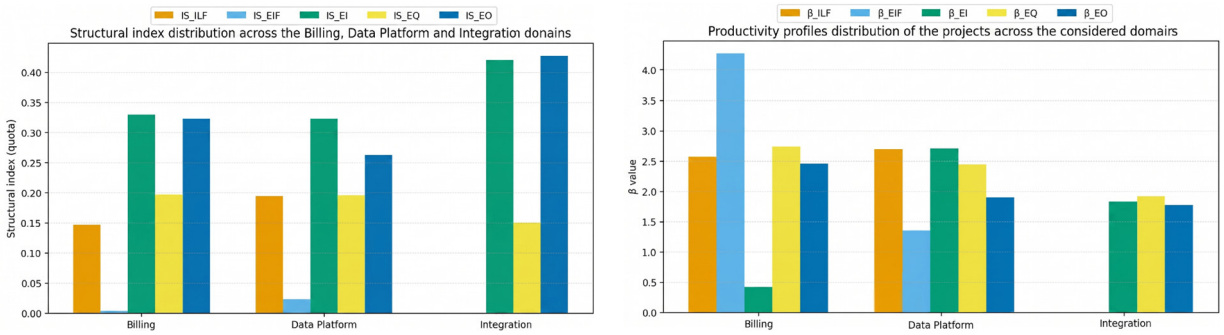


Figure 3. Baseline structural profiles: (**a**) structural index distribution across the Billing, Data Platform and Integration domains; (**b**) productivity profile distribution of the projects across the considered domains (beta values).

Figure 3b shows the Productivity Profiles of the same projects, estimated for the three domains. Unlike the Structural Index, which describes the composition of the projects, the Productivity Profile reflects the relative contribution of each functional component category to observed productivity within the corresponding domain. The Billing domain exhibits the most heterogeneous profile, characterized by a dominant contribution of EIF components and a comparatively limited contribution from EI. This suggests that productivity in this domain is primarily associated with functionalities involving the reuse and integration of external data sources. The Data Platform domain shows a more balanced profile, so that productivity emerges from the combined presence of data management, transformation, and processing functionalities. The Integration Platform domain displays the most specialized profile. Productivity is almost entirely associated with transactional components (EI, EO, and EQ), while data-oriented components (ILF and EIF) exhibit negligible contributions. This behavior is consistent with the structural characteristics of the domain, which focuses primarily on information exchange and process orchestration. Overall, the observed differences indicate that the same functional component category may contribute differently to productivity depending on the application context. The resulting Productivity Profiles therefore support the use of domain-specific baselines.

Figure 4 reports the goodness-of-fit analysis for the three baseline application domains. In all cases, the estimated effort values follow the overall trend of the observed effort values, with most project instances distributed around the ideal 1:1 relationship. The adjusted R^2 values range from 0.760 to 0.906, indicating that the regression models explain a substantial proportion of the variability observed in the analyzed datasets. The highest explanatory capability is observed for the Billing domain (Adj. R^2 = 0.906),

followed by the Integration domain (Adj. R^2 = 0.834), while the Data Platform domain exhibits a lower but still satisfactory fit (Adj.R^2 = 0.760). These results suggest that the estimated productivity weights capture relevant relationships between functional composition and development effort.

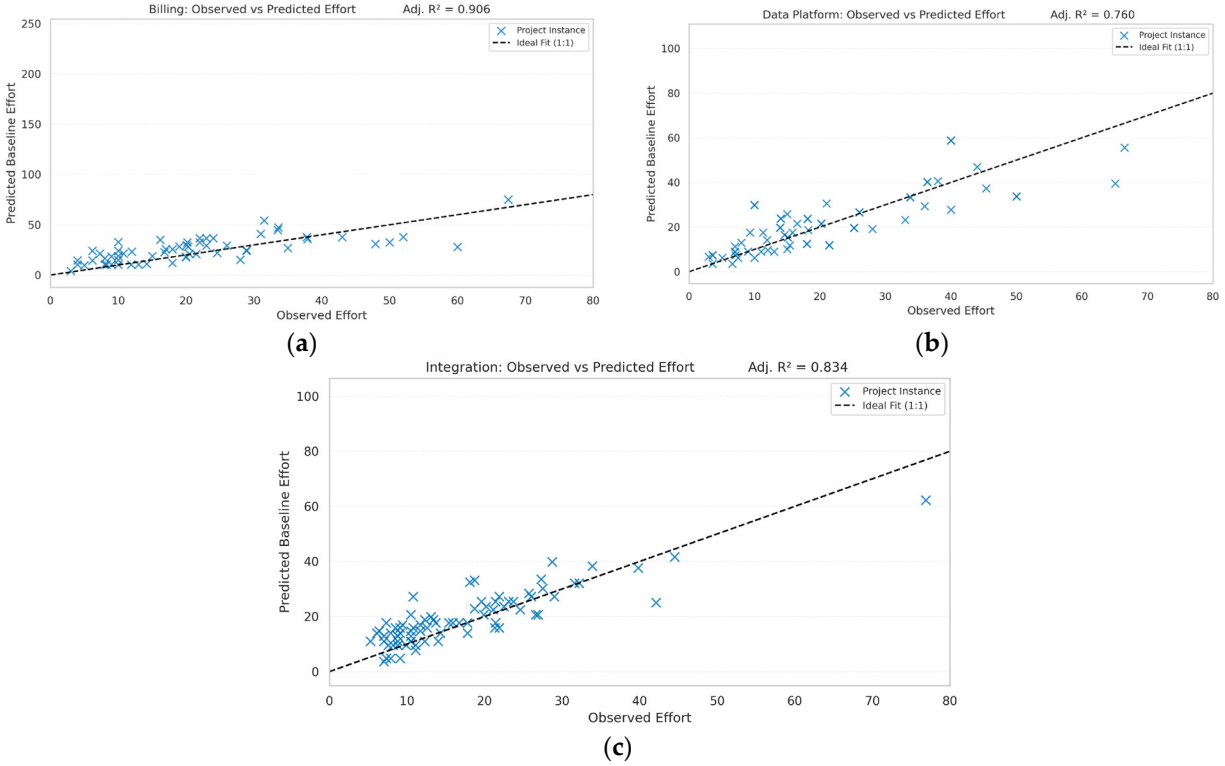


Figure 4. Goodness-of-fit analysis of the regression-based productivity model for the three baseline application domains: (**a**) Billing; (**b**) Data Platform; (**c**) Integration Platform. The dashed line represents the ideal 1:1 relationship between observed and predicted values. Adjusted values are reported for each domain to quantify the proportion of effort variability explained by the model.

Figure 5 reports the results of the bootstrap stability analysis (N = 1000). The bootstrap mean coefficients closely match the corresponding estimates reported in Figure 3b, indicating limited estimation bias. For most functional components, confidence intervals remain relatively narrow, suggesting stable productivity weights across resampled datasets. Wider intervals are observed for EIF in the Billing and Data Platform domains, reflecting the lower representativeness of this component within the corresponding project samples. Overall, the results indicate that the observed productivity profiles are reasonably stable with respect to sampling variability.

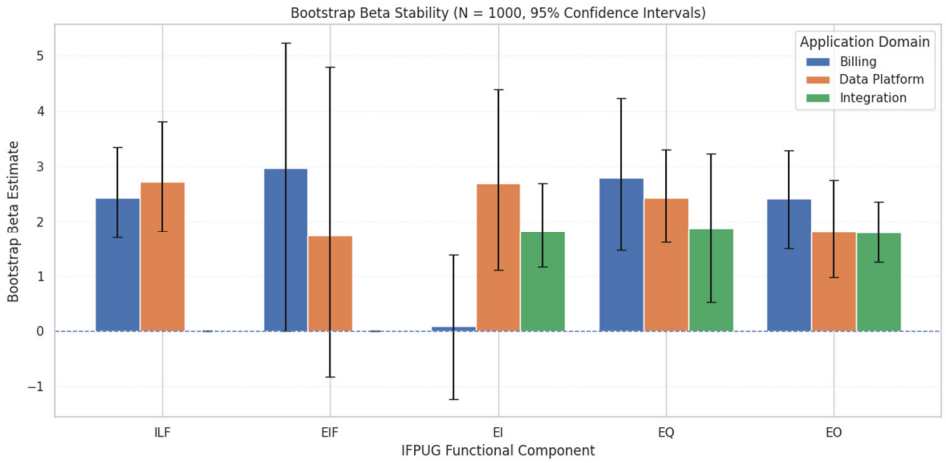


Figure 5. Bootstrap Stability of Domain-Specific Productivity Weights (N = 1000, 95% Confidence Intervals).

4.3.2. Results for S2: Discriminative Validation

To address H2, the proposed model was applied to the test projects and the resulting Expected Productivity values were analyzed with respect to their Structural Distance from the baseline. The objective was to verify whether projects characterized by functional structures compatible with the reference domain exhibit comparable expected productivity behavior regardless of their absolute size. Figure 6 reports the distribution of baseline projects, structurally consistent test projects, and outliers with respect to Structural Distance and Expected Productivity.

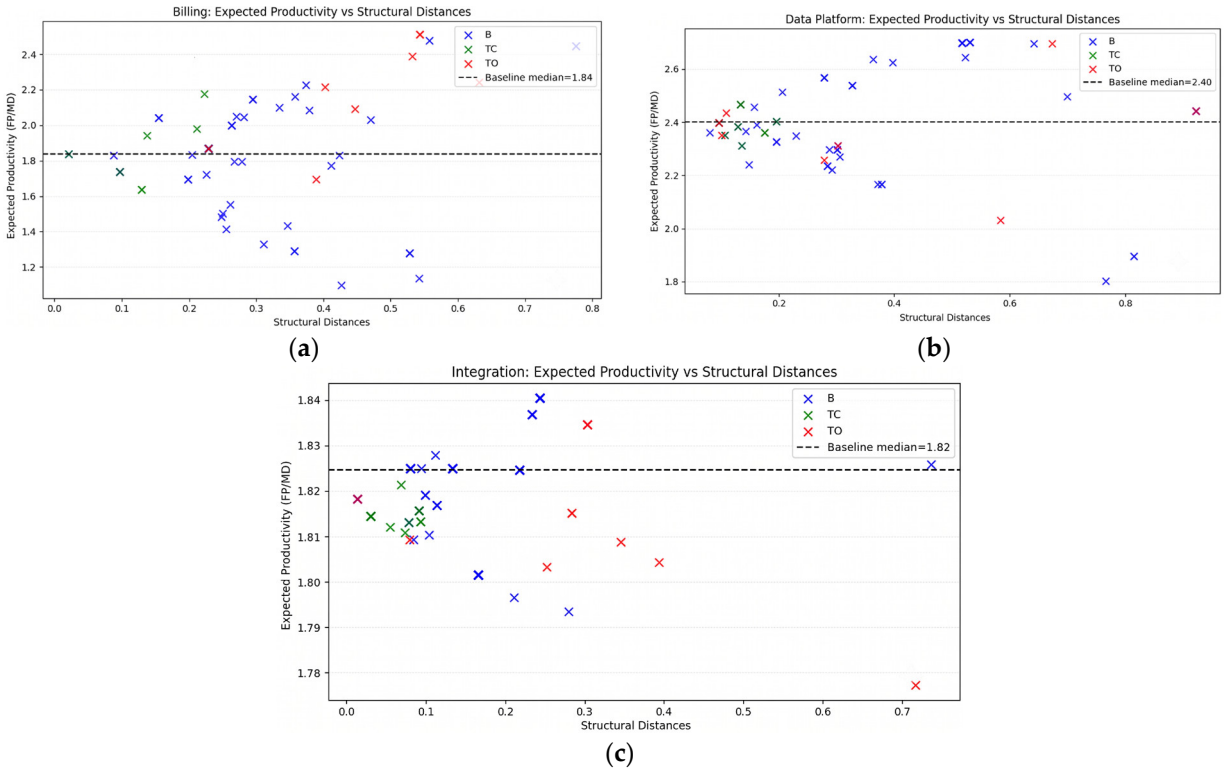


Figure 6. Distribution of baseline projects (B), consistent test projects (TC), and outlier projects (TO) in terms of Structural Distance and Expected Productivity: (**a**) Billing; (**b**) Data Platform; (**c**) Integration Platform.

In the Billing domain (baseline Expected Productivity ≈ 1.84 FP/MD), as shown in Figure 6a, Consistent Test projects are concentrated around the baseline productivity level despite differences in project size, suggesting that structurally similar projects remain comparable with respect to productivity. Outlier Test projects, on the other hand, exhibit larger Structural Distances and less coherent productivity behavior. The two populations occupy distinct regions of the distribution, highlighting the discriminative capability of the proposed indicators. Figure 6b shows that in the Data Platform domain (baseline Expected Productivity ≈ 2.40 FP/MD), Consistent Test projects exhibit a highly concentrated distribution around the baseline productivity level and are associated with minimal Structural Distances. Outlier projects show only moderate deviations in most cases, with a few exceptions. Figure 6c shows that the Integration Platform domain (baseline Expected Productivity ≈ 1.82 FP/MD) provides the strongest evidence in support of H2. Consistent Test projects are highly concentrated around the baseline productivity level and exhibit minimal Structural Distances, despite differences in project size. Outlier projects, on the other hand, clearly depart from baseline behavior, resulting in the most pronounced separation observed among the analyzed domains.

To complement the visual analysis reported in Figure 6, a statistical separation analysis was performed on the Consistent and Outlier cohorts. Figure 7 reports the distribution of Structural Distance values together with the results of the Mann–Whitney U test and the corresponding Silhouette coefficients. The results indicate a clear separation between the two populations across all application domains. The Mann–Whitney U test indicates that the observed differences are statistically significant in all domains ($p < 0.01$). The Silhouette coefficients, ranging from 0.22 to 0.44, quantify the degree of separation, with the strongest separation observed in the Integration domain. Since Structural Distance is also used to define the consistency classes, these results are interpreted as a quantitative characterization of the separation obtained on the unseen test projects.

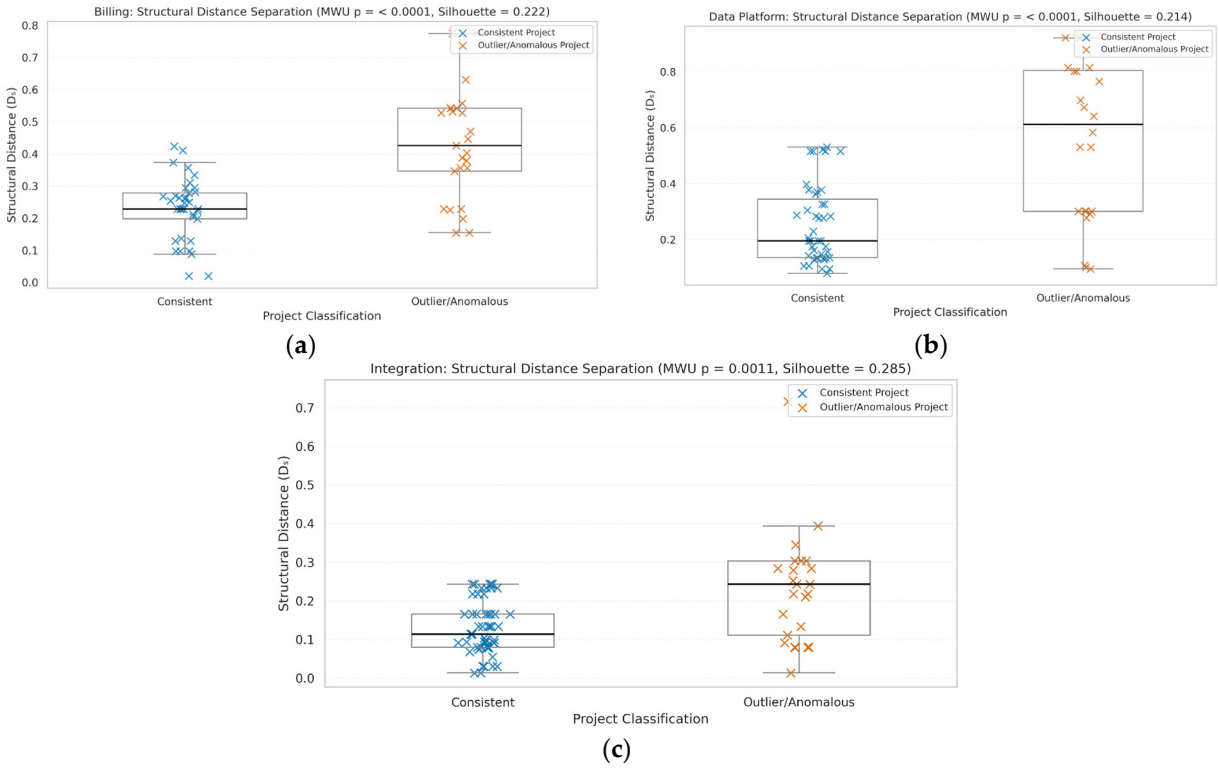


Figure 7. Structural Distance distributions for Consistent and Outlier project cohorts across the analyzed application domains: Billing (**a**), Data Platform (**b**), and Integration (**c**). The reported Mann–Whitney U test *p*-values characterize the statistical differences in Structural Distance between

the two cohorts, while the Silhouette coefficients quantify the degree of separation associated with the adopted consistency classification.

Overall, the observed distributions show that structurally aligned projects exhibit more comparable productivity behavior and that the proposed structural indicators characterize different degrees of project alignment with the reference domain. These findings provide empirical support for H2 and partial support for H3, specifically regarding the identification of projects that deviate from the domain baseline.

Since the identification of consistency classes of the projects relies on percentile-based thresholds, a sensitivity analysis was conducted to assess the impact of threshold selection on the discriminative capability of the proposed approach. Three alternative configurations were considered: a stricter configuration using the 33rd and 66th percentile thresholds (P33–P66), the reference configuration using the 40th and 80th percentiles (P40–P80), and a more permissive configuration using the 50th and 90th percentiles (P50–P90). The corresponding sensitivity analysis results are reported in Table 6.

Table 6. Coherence-class sensitivity analysis results under alternative percentile threshold configurations by application domain.

Configuration	Domain	Deviation High	Deviation Medium	Deviation Low	Separation Ratio
Narrow (P33–66)	Billing	0.0554	0.1818	0.4958	89.574
Narrow (P33–66)	Data Platform	0.088	0.2461	0.4343	4.938
Narrow (P33–66)	Integration	0.0506	0.1439	0.3598	71.067
Baseline (P40–80)	Billing	0.0634	0.2602	0.5471	86.238
Baseline (P40–80)	Data Platform	0.104	0.285	0.5009	48.186
Baseline (P40–80)	Integration	0.0637	0.1955	0.4407	69.182
Wide (P50–90)	Billing	0.077	0.3545	0.6866	89.164
Wide (P50–90)	Data Platform	0.1298	0.3291	0.6192	47.715
Wide (P50–90)	Integration	0.0816	0.2667	0.4877	59.759

As expected, relaxing the thresholds increases the number of projects classified as highly consistent by enlarging the admissible region around the baseline centroid. However, the separation ratio between the high- and low-consistency classes remains relatively stable across the considered percentile configurations. This indicates that the discriminative capability of the proposed approach is not strongly dependent on the specific threshold values adopted. The proposed P40–P80 configuration therefore represents a pragmatic compromise between selectivity and coverage. It identifies a limited set of highly consistent projects while maintaining a clear separation from structurally or productively atypical projects.

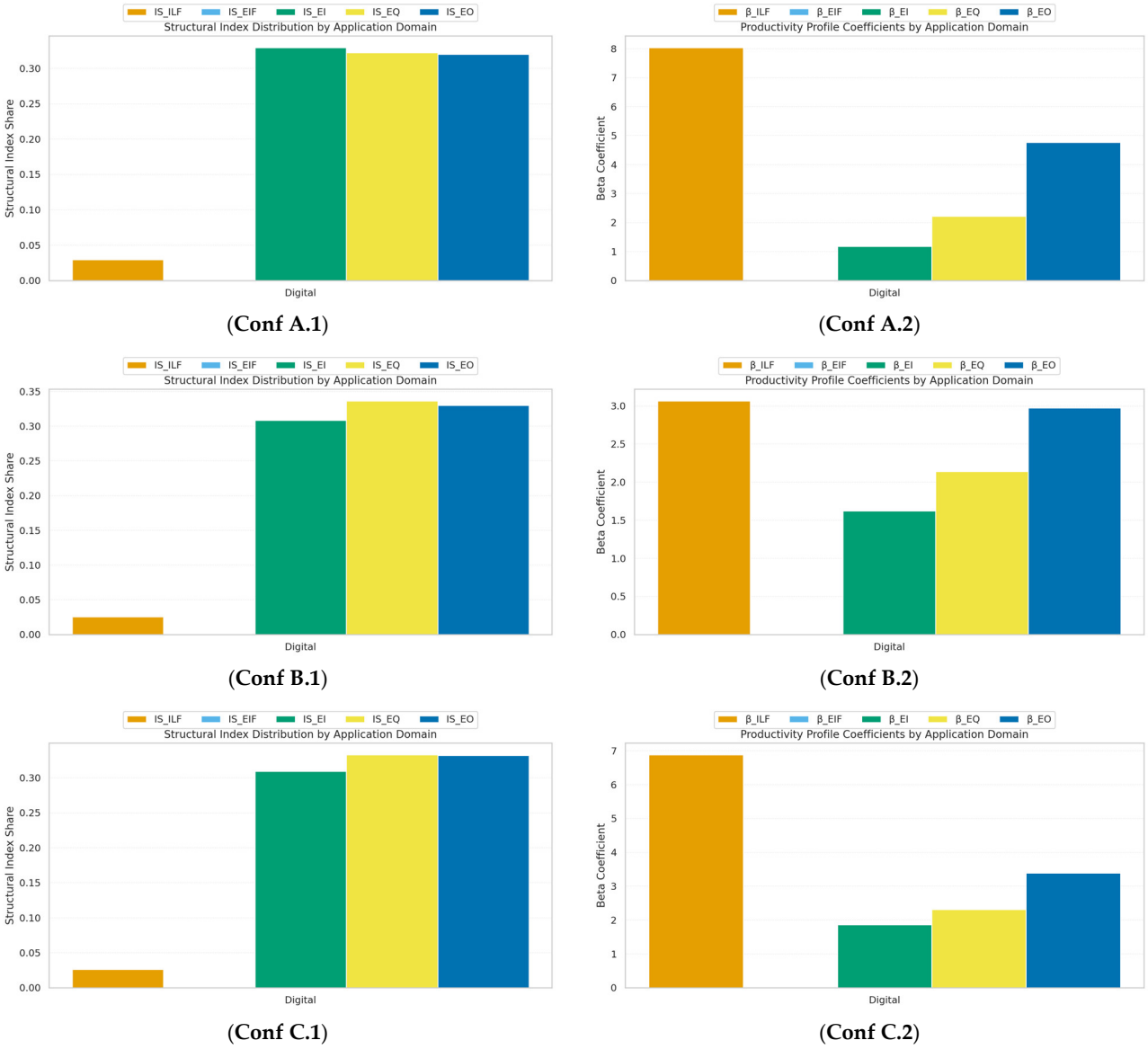
4.3.3. Results for S3: Structural Discrimination Analysis Against Raw Productivity

To examine whether the Structural Discrimination Analysis depends on a specific baseline composition, three experimental configurations, denoted A, B, and C, were generated from the same Digital project population. Each configuration uses the same set of 203 real projects for the analysis, while a different subset of 100 projects is used to construct the corresponding domain baseline.

The three configurations therefore differ only in the composition of the baseline used to derive the structural reference and the domain-specific Productivity Profile.

This design allows the same project population to be evaluated against three distinct baseline configurations and supports the assessment of whether the observed structural discrimination patterns remain consistent when the baseline composition changes.

As shown in Figure 8, the three baseline configurations were generated from the same application domain and the same population of projects. As expected, the structural component of the three baselines shows an overall similar geometry, indicating substantial consistency in the distribution of the IFPUG functional components. The productivity profiles also show a generally consistent pattern across the three configurations, although some variations can be observed in the contributions of individual coefficients. These differences show the influence of the sample composition used to build the baseline and confirm the importance of its selection. Overall, however, the three configurations maintain comparable functional structures and productivity profiles.



(**Conf A.1**) (**Conf A.2**)

(**Conf B.1**) (**Conf B.2**)

(**Conf C.1**) (**Conf C.2**)

Figure 8. Comparison of the structural components and productivity profiles across three baseline configurations generated from the same application domain and project population: (**Conf. A.1**, **B.1**, **C.1**) structural index distributions for configurations A, B, and C, respectively; (**Conf. A.2**, **B.2**, **C.2**) productivity profile distributions for configurations A, B, and C, respectively (beta values).

As shown in Figure 8, by selecting pairs of projects characterized by very similar Raw Productivity values but different structural distances from the baseline, it can be observed that their functional composition may differ significantly. Projects that are therefore almost indistinguishable when considering only the aggregate ratio between Function

Points and Effort may occupy different structural positions and show different distributions of the five IFPUG components.

This initial evidence shows that Raw Productivity, while describing the observed productivity level, does not preserve information about the functional structure that contributed to determining it. The proposed representation therefore introduces a complementary level of information, allowing projects with similar aggregate productivity but different structural characteristics to be distinguished. The selected cases provide illustrative examples of this phenomenon, as shown in Figure 9, whose presence is subsequently verified across the complete set of project pairs in the three baseline configurations.

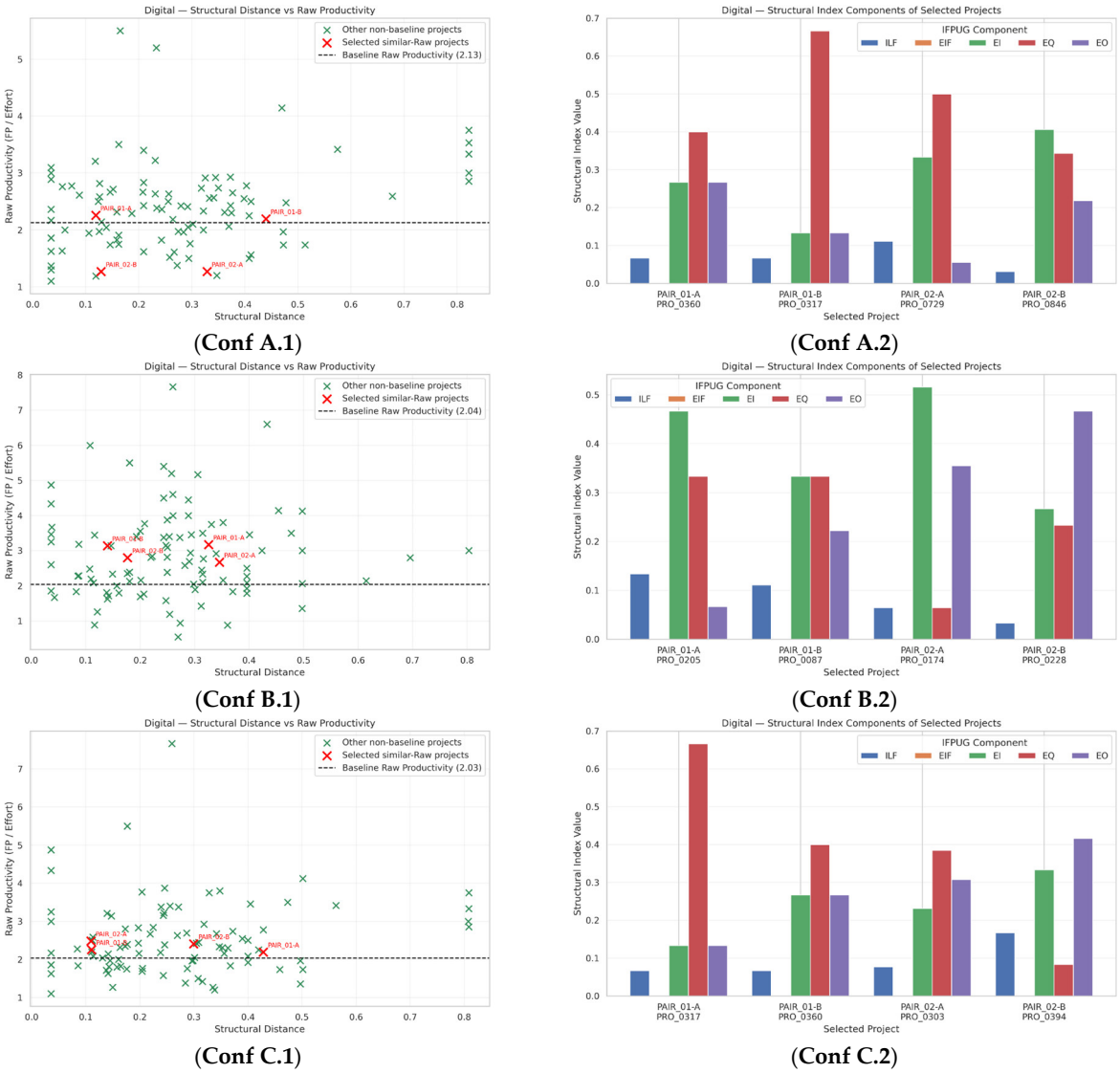


Figure 9. Comparison of the functional composition of project pairs with similar Raw Productivity but different structural distances from the baseline: (**Conf. A1**, **B1**, **C1**) Structural Distance vs. Raw Productivity for configurations A, B, and C, respectively; (**Conf. A2**, **B2**, **C2**) Structural Index components of the selected projects for configurations A, B, and C, respectively.

To verify that the phenomenon observed in the selected cases was not limited to example project pairs, the analysis was extended to all eligible pairs across the three baseline configurations. Pairs with similar Raw Productivity were defined as those characterized by a relative difference not exceeding 5%. For each of these pairs, the

absolute difference in Structural Distance (Δ D_S) was then calculated, without using structural distance as a prior selection criterion.

As reported in Table 7, the results show a similar and consistent pattern across the three configurations. In configuration A, 6 pairs with similar Raw Productivity were identified, compared with 8 in configuration B and 10 in configuration C. The median Δ D_S values are 0.107, 0.116, and 0.139, respectively, while the maximum values reach 0.214, 0.238, and 0.242. In particular, 50% of the pairs in configurations A and B and 70% in configuration C show a structural difference of at least 0.10. Differences equal to or greater than 0.20 are also observed in 16.7%, 25.0%, and 20.0% of the pairs, respectively.

Table 7. Structural distance differences between project pairs with similar raw productivity.

Configuration	Eligible Projects	Total Pairs	Similar-P_R Pairs	Similar-P_R Pairs (%)	Median Δ Ds	IQR Δ Ds	Δ D_S ≥ 0.10	Δ D_S ≥ 0.20	Max Δ D_S
A	9	36	6	16.7%	0.107	0.077	3 (50.0%)	1 (16.7%)	0.214
B	14	91	8	8.8%	0.116	0.105	4 (50.0%)	2 (25.0%)	0.238
C	13	78	10	12.8%	0.139	0.126	7 (70.0%)	2 (20.0%)	0.242

These results indicate that the phenomenon highlighted through the representative cases is not limited to an occasional selection of projects. Across the three different configurations, a relevant proportion of the pairs that appear similar according to Raw Productivity nevertheless occupy different structural positions with respect to the baseline. Similarity in aggregate productivity therefore does not necessarily imply structural similarity.

This evidence supports the complementary role of the proposed representation with respect to Raw Productivity: P_R describes the observed productivity level, while D_S adds information about the structural position of the project relative to the reference domain. The two measures should therefore not be considered alternatives, but rather as describing different aspects of a project that cannot be distinguished through the FP/Effort ratio alone.

4.3.4. Results for S4: Structural Stress-Test Analysis

Synthetic stress testing was performed using four controlled scenarios (Standard—STD, Moderate Structural Shift—MST, Mismatch—MIS, and Extreme datasets—EXT), to examine the model response under different levels and types of perturbation relative to the baseline structural conditions. The purpose of this analysis is not to demonstrate generalization to arbitrary unseen conditions, but to characterize the behavior and validity limits of the baseline-derived model when applied to progressively more challenging structural configurations.

As shown in Figure 10, the Standard scenario (STD) produces stable baseline-derived Expected Productivity values concentrated around the reference productivity level. The Moderate Structural Shift scenario (MOD), introduced to represent projects that deviate from the baseline while maintaining structurally plausible functional compositions, also preserves a relatively stable Expected Productivity response. This intermediate scenario is particularly relevant because it examines model behavior outside the closest baseline conditions without introducing deliberately incompatible structures.

A different behavior emerges under the Mismatch (MIS) and Extreme (EXT) scenarios. In the Mismatch scenario, Expected Productivity remains centered close to the reference level but shows a strongly compressed distribution, indicating that stability of the mean does not necessarily imply an equally informative model response. Under Extreme conditions, the Expected Productivity distribution becomes substantially more dispersed and includes non-plausible negative values. These values indicate that the

linear Productivity Profile is being applied under structural conditions for which the baseline-derived coefficients no longer provide a meaningful representation of productivity behavior.

Across the four scenarios, the Structural Distance distributions remain broadly comparable, while the Expected Productivity distributions show different patterns according to the imposed structural perturbations. Therefore, the observed changes cannot be attributed only to a systematic increase in average Structural Distance. Standard and Moderate conditions produce relatively stable Expected Productivity distributions, Mismatch conditions result in a more compressed response, and Extreme conditions produce substantially greater dispersion, including non-plausible negative values. Overall, the stress test provides a descriptive characterization of the model response under the four controlled structural conditions considered in the experiment.

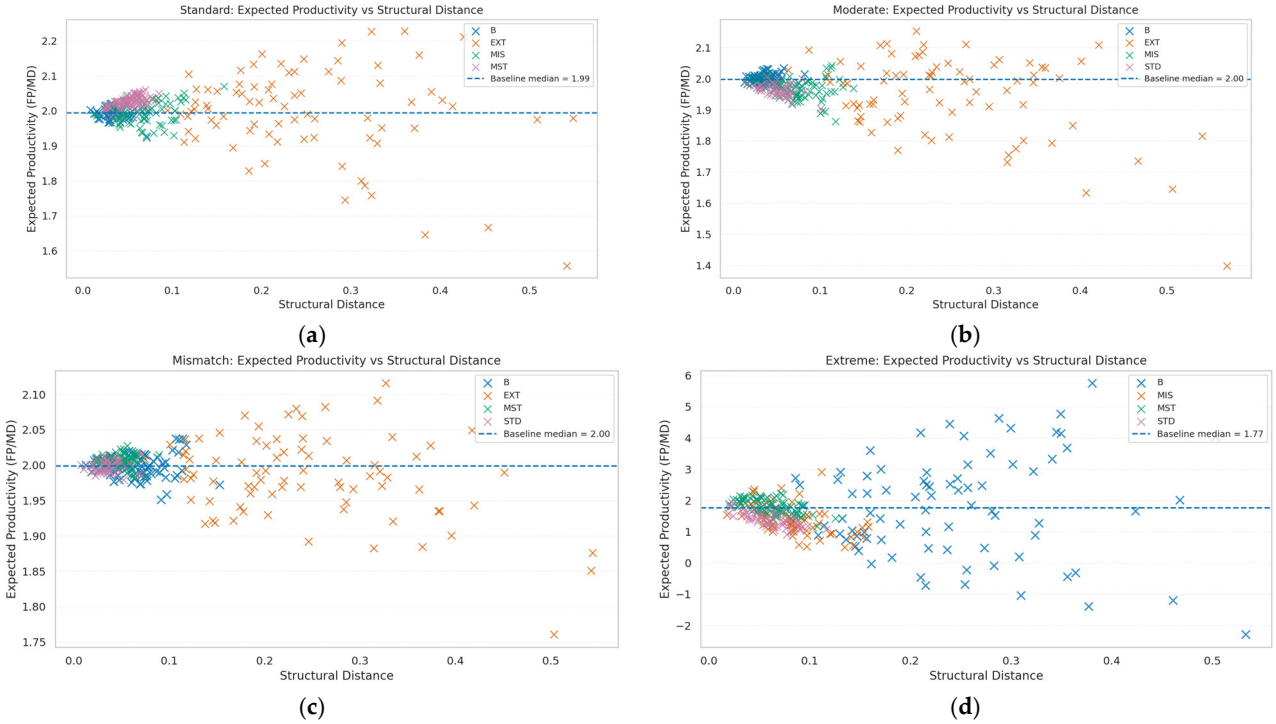


Figure 10. Distribution of baseline projects (B) in terms of Structural Distance and Expected Productivity across the four synthetic stress-test scenarios: (**a**) Standard, (**b**) Moderate Structural Shift, (**c**) Mismatch, and (**d**) Extreme.

As summarized in Table 8, the Structural Distance distributions remain comparable across the four scenarios, both in terms of mean values and dispersion. Standard and Moderate show similar Expected Productivity distributions, while Mismatch maintains a comparable mean but produces a more concentrated response. Extreme shows a different pattern, with a lower mean Expected Productivity, substantially greater dispersion, and a range that includes negative values. The comparison therefore highlights different model responses across the four controlled structural conditions, complementing the patterns observed in Figure 7.

These results also reflect the adaptive nature of the proposed approach, since Expected Productivity represents a baseline-conditioned productivity reference derived from domain-specific Productivity Profiles rather than from a universal productivity model.

Table 8. Structural Distance and Expected Productivity statistics across synthetic stress-test scenarios.

Scenario	*Ds* (Mean)	*Ds* (SD)	Expected Prod. (Mean)	Expected Prod. (SD)	Min	Max
Standard	0.1063	0.1045	2.0047	0.0703	1.5569	2.2277
Moderate	0.1071	0.1043	1.9693	0.0731	1.3981	2.1537
Mismatch	0.1055	0.1042	1.9966	0.0337	1.7608	2.1161
Extreme	0.1163	0.0960	1.6238	0.9171	−2.2830	5.7580

5. Discussion

This section discusses the implications of the experimental findings and examines the main threats to the validity of the study.

Overall, the experimental findings support the research hypotheses from complementary perspectives. The domain profile analysis indicates that different functional structures are associated with distinct domain-specific productivity patterns. The analysis of unseen projects shows that projects structurally aligned with the reference domain exhibit more comparable productivity behavior, while the structural discrimination analysis shows that similar aggregate Raw Productivity does not necessarily imply similar structural alignment with the domain baseline. The robustness analysis further shows that the proposed representation remains stable within the structural space represented by the baseline, while its interpretability progressively decreases outside this range. Taken together, these findings support the use of functional composition as an additional dimension for interpreting software productivity beyond aggregate Function Point-based measures.

These results are consistent with previous studies suggesting that individual IFPUG functional components may provide information that is not preserved by aggregate functional size measures. They are also aligned with industrial evidence indicating that Function Point-based productivity should be interpreted in relation to the application context. The proposed approach connects these perspectives by explicitly relating functional composition to domain-specific productivity behavior.

Beyond the scientific objective of investigating the relationship between functional composition and productivity, the proposed approach was conceived as a practical tool for industrial productivity assessment. The performed experiments indicate that the Structural Productivity Index provides an interpretable reference for assessing whether the productivity of a new project is consistent with patterns observed in projects exhibiting similar functional structures within the same application domain.

In an industrial context, the construction of a productivity baseline requires the selection of projects that are representative of, and consistent with, the current functional, technological, and architectural characteristics of the application domain, rather than the simple aggregation of all projects belonging to that domain. Projects exhibiting highly peculiar configurations or reflecting solutions that are no longer representative may require specific evaluation and, where appropriate, exclusion to avoid introducing distortions into the resulting reference profile. However, in this study, a more inclusive experimental strategy was intentionally adopted. Less frequent functional configurations were retained in the analyzed datasets to assess their impact on the Structural Productivity Index and the stability of the estimated productivity weights. Consequently, the observed variability of some coefficients should not necessarily be interpreted as a limitation of the baseline itself. Rather, it provides useful information for identifying the conditions under which a productivity baseline remains representative and statistically stable.

As with any empirical study, these findings should be interpreted considering the assumptions and limitations of the adopted datasets and evaluation methodology.

Threats to construct validity relate to limitations in the experimental setup. Construct validity may be limited by the fact that the evaluation was not performed on the ISBSG repository, which is one of the most widely adopted benchmarking datasets in software engineering research. However, access to the complete ISBSG data is subject to licensing restrictions, which may limit its use in empirical studies conducted outside funded academic research initiatives. The present study relies on anonymized and aggregated industrial project data collected from real software development contexts. To improve transparency, reproducibility, and future comparisons, both the datasets and the source code used in the experimental evaluation have been made publicly available in [7].

An additional indication of dataset realism is provided by the coherence of the derived Structural and Productivity Profiles. The observed distributions of IFPUG functional components are consistent with the expected characteristics of the corresponding application domains. Billing systems are characterized by a strong transactional component combined with significant data management functionalities. Data Platforms exhibit a more balanced distribution of data and processing functions, whereas Integration Platforms are dominated by transactional functionalities and show negligible contributions from data persistence components. A similar consistency can be observed in the derived Productivity Profiles. Billing exhibits higher productivity weights for ILF, EQ, and EO than for EI; Data Platforms show a more balanced distribution of productivity contributions across the functional components, whereas Integration Platforms concentrate productivity almost exclusively on transactional functions.

Threats to internal validity concern both the composition of the analyzed datasets and the robustness of the statistical procedures used to derive the productivity profiles. The dataset size is intentionally limited, as projects were selected according to data quality and consistency criteria to ensure meaningful comparisons across projects and application domains and to support the definition of representative domain-specific productivity baselines. The adopted criteria were derived from established Function Point Analysis and software productivity measurement practices. In addition, because the productivity profiles are obtained from regression-based models, their reliability depends on the stability of the estimated weights. For this reason, multicollinearity and coefficient stability were assessed through VIF analysis and bootstrap resampling, respectively. The results suggest that the estimated weights are reasonably stable for the analyzed datasets and domains. In industrial applications, coefficient stability can therefore provide an additional criterion for assessing the reliability of a domain-specific Productivity Profile.

Threats to external validity concern the extent to which the results can be generalized beyond the analyzed project datasets. The three baseline domains were selected to evaluate the research hypotheses across clearly differentiated software contexts. To investigate the broader applicability of the regression-based formulation, the multicollinearity assessment was extended to two additional domains, showing that problematic predictor dependencies were not observed in these datasets. Furthermore, the structural discrimination analysis was conducted on projects belonging to one of the added domains. Evaluation on additional project populations could provide further empirical evidence across a broader range of industrial contexts.

The available industrial datasets do not include independent project outcomes such as schedule adherence, budget deviation, or ex-post expert assessments. Therefore, the relationship between structural consistency and such external project outcomes could not be evaluated in the present study and represents a relevant direction for future validation.

6. Conclusions

This work introduced the Structural Productivity Model, an adaptive analytical framework for software productivity assessment that extends traditional Function Point-based approaches through an explicit representation of project functional structure. In particular, the proposed model derives domain-specific Productivity Profiles that characterize the contribution of different functional component categories to observed productivity behavior.

The experimental evaluation conducted across three software application domains showed that distinct functional contexts are associated with distinguishable productivity profiles, supporting the hypothesis that productivity behavior is inherently domain-dependent. The results further demonstrated that structurally similar projects tend to exhibit comparable productivity characteristics, while structurally anomalous projects can be identified through the proposed structural indicators. In addition, the structural discrimination analysis showed that projects with similar aggregate Raw Productivity may exhibit different degrees of structural alignment with the reference baseline, providing empirical evidence that aggregate productivity alone does not preserve all the structural information captured by the proposed representation. Synthetic stress tests indicated that the model remains stable within the structural domain represented by the baseline and progressively loses interpretability when applied outside its calibration range.

Thus, the proposed approach provides an interpretable and adaptive framework for complementing aggregate and purely data-driven productivity estimation techniques. While recent AI-based methods often focus on maximizing prediction accuracy through increasingly complex learning models, the Structural Productivity Model explicitly relates productivity estimates to the functional composition of software projects through an interpretable structural representation. In this sense, the model can be viewed as a form of domain-adaptive productivity analytics that complements machine-learning approaches by making explicit the contribution of functional components to productivity estimates.

Indeed, future research will investigate the integration of the proposed structural representation with machine-learning techniques and its evaluation across broader and more heterogeneous software project repositories. The proposed representation may provide explicit domain-specific structural information that could be incorporated into informed machine learning frameworks, where prior knowledge is integrated with data-driven learning to improve interpretability and domain awareness [22].

Author Contributions: Conceptualization, C.A.; Methodology, C.A. and M.L.V.; Software, C.A.; Validation, C.A.; Data curation, C.A.; Writing—original draft, C.A. and M.L.V.; Writing—review and editing, C.A. and M.L.V.; Visualization, C.A.; Supervision, M.L.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data and experimental artifacts used in this study are publicly available in [7]. The underlying datasets originate from private projects unrelated to ENEA and were accessed only by the first author; these cannot be disclosed due to confidentiality and contractual restrictions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jones, C. *Applied Software Measurement: Global Analysis of Productivity and Quality*, 3rd ed.; McGraw-Hill: New York, NY, USA, 2008.
2. Duarte, C.H.C. Software Productivity in Practice: A Systematic Mapping Study. *Software* **2022**, *1*, 164–214. https://doi.org/10.3390/software1020008.
3. Albrecht, A.J. Measuring Application Development Productivity. In Proceedings of the Joint SHARE/GUIDE/IBM Application Development Symposium, Monterey, CA, USA, 14–17 October 1979; pp. 83–92.
4. International Function Point Users Group (IFPUG). *Function Point Counting Practices Manual (CPM), Release 4.3.1*; IFPUG: Westerville, OH, USA, 2010. Available online: https://ifpug.org/ifpug-standards/fpa (accessed on 18 April 2026).
5. Abran, A.; Desharnais, J.M.; Oligny, S.; St-Pierre, D.; Symons, C.R. *COSMIC-FFP Measurement Manual for ISO/IEC 19761*; COSMIC Implementation Guide Series; Common Software Measurement International Consortium (COSMIC): Montreal, QC, Canada, 2003. Available online: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:19761:ed-2:v1:en (accessed on 18 April 2026).
6. *ISO/IEC 24570:2018*; Software and Systems Engineering—NESMA Functional Size Measurement Method—Definitions and Counting Guidelines for the Application of Function Point Analysis. International Organization for Standardization (ISO): Geneva, Switzerland. Available online: https://www.iso.org/standard/72505.html (accessed on 21 August 2026).
7. Antonelli, C. Functional Structural Signature Model: Source Code, Datasets, and Experimental Evaluation Artifacts; GitHub Repository. 2026. Available online: https://github.com/ClaudioAntonelli/Functional-Structural-Signature-Model (accessed on 21 August 2026).
8. *ISO/IEC 20926:2009*; Software and Systems Engineering—Software Measurement—IFPUG Functional Size Measurement Method 2009. International Organization for Standardization (ISO): Geneva, Switzerland, 2009. Available online: https://www.iso.org/standard/51717.html (accessed on 18 April 2026).
9. Vo, V.H.; Ho, L.T.K.N.; Huynh, H.T. A Review of Software Effort Estimation by Using Functional Points Analysis. In *Computational Statistics and Mathematical Modeling Methods in Intelligent Systems*; Advances in Intelligent Systems and Computing; Springer: Cham, Switzerland, 2019; pp. 408–422. https://doi.org/10.1007/978-3-030-31362-3_40.
10. Quesada-López, C.; Jenkins, M. Function Point Structure and Applicability: A Replicated Study. *J. Object Technol.* **2016**, *15*, 1–26. https://doi.org/10.5381/jot.2016.15.3.a2.
11. Lavazza, L.; Morasca, S.; Robiolo, G. Towards a simplified definition of Function Points. *Inf. Softw. Technol.* **2013**, *55*, 1796–1809. https://doi.org/10.1016/j.infsof.2013.04.003.
12. Gencel, C.; Demirors, O. Functional size measurement revisited. *ACM Trans. Softw. Eng. Methodol.* **2008**, *17*, 3. https://doi.org/10.1145/1363102.1363106.
13. Bok, H.S.; Raman, K.S. Software Engineering Productivity Measurement using Function Points: A Case Study. *J. Inf. Technol.* **2000**, *15*, 79–90. https://doi.org/10.1177/026839620001500108.
14. Lavazza, L.; Locoro, A.; and Meli, R. Software Development and Maintenance Effort Estimation Using Function Points and Simpler Functional Measures. *Software* **2024**, *3*, 442–472. https://doi.org/10.3390/software3040022.
15. Matalkah, B.; El-Adaileh, N.; and Hajaya, M.K. A Hierarchical Software Quality Model for Estimating Function Point Analysis. *WSEAS Trans. Inf. Sci. Appl.* **2025**, *22*, 153–165. https://doi.org/10.37394/23209.2025.22.15.
16. Czarnacka-Chrobot, B. AI applications in software functional size measurement: A systematic literature review. *J. Syst. Softw.* **2026**, *232*, 112686. https://doi.org/10.1016/j.jss.2025.112686.
17. Wen, J.; Li, S.; Lin, Z.; Hu, Y.; Huang, C. Systematic Literature Review of Machine Learning Based Software Development Effort Estimation Models. *Inf. Softw. Technol.* **2012**, *54*, 41–59. https://doi.org/10.1016/j.infsof.2011.09.002.
18. Ali, A.; and Gravino, C. A systematic literature review of software effort prediction using machine learning methods. *J. Softw. Evol. Process* **2019**, *31*, 10. https://doi.org/10.1002/smr.2211.
19. Shepperd, M.; Schofield, C. A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models. *J. Syst. Softw.* **1997**, *39*, 281–289. https://doi.org/10.1016/S0164-1212(97)00055-1.
20. Cabral, J.T.H.A.; Oliveira, A.L.I.; da Silva, F.Q.B. Ensemble Effort Estimation: An Updated and Extended Systematic Literature Review. *J. Syst. Softw.* **2023**, *195*, 111542. https://doi.org/10.1016/j.jss.2022.111542.

21. Azzeh, M.; Nassif, A.B.; and Martín, C.L. Empirical analysis on productivity prediction and locality for use case points method. *Softw. Qual. J.* **2021**, *29*, 309–336. https://doi.org/10.1007/s11219-021-09547-0*).*
22. von Rueden, L.; Mayer, S.; Beckh, K.; Georgiev, B.; Giesselbach, S.; Heese, R.; Kirsch, B.; Pfrommer, J.; Pick, A.; Ramamurthy, R.; et al. Informed Machine Learning—A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 614–633. https://doi.org/10.1109/TKDE.2021.3079836.