Guidance from the Forecasting and Software Estimation Committee

uTip – Estimating Software Cost and Schedule in the AI Era

Using Function Points, SNAP, and AI-Aware Productivity Baselines

Authors: Emanuele Richiusa, Milind Mahajani		
Reviewers		
Atul Vaze	Daniel French	Eduardo Orefice
Grammati-Eirini Kastori	Zacharias Samos	Antonio Ferre

uTips (Usage Tips) provide insight into potential uses of IFPUG standards to support an organization's business needs. They do not replace standards or detailed procedural guidance. They help explain how IFPUG methods may be applied in realistic situations based on the application of IFPUG counting rules as defined in the Manuals of the IFPUG standards. This uTip focuses on how Function Points, Software Non-functional Assessment Process (SNAP), historical data, and productivity baselines can be used to improve software forecasting and estimation in AI-based environments.

Introduction

A common misconception is that AI makes software measurement less relevant. The opposite is true.

As AI changes how software is developed and introduces greater variability in delivery performance, objective, repeatable, and verifiable size measures and historical performance data become more important, not less.

Function Points remain applicable because they measure the functional user requirements delivered to the user, regardless of programming language, implementation technology, or the degree of automation used in development. SNAP remains relevant because it measures the non-functional user requirements, which may become more significant in AI-driven solutions due to additional architectural, operational, quality, and technical environment demands.

Terminology note: in this uTip, the term Function Points (FP) is used as a generic reference to functional size measured in accordance with either the IFPUG Function Points Counting Practices Manual (CPM), or the IFPUG Simple Function Points Counting Practices Manual (SPM), as applicable. Unless explicitly stated otherwise, references to FP and FP-based measures include both measurement approaches.

The practical question is therefore not whether software should still be measured, but how established IFPUG sizing methods and historical productivity data can be used effectively in AI-driven environments.

This uTip provides recommendations on how to:

- Use Function Points and SNAP as inputs to software estimation and forecasting
- Build historical repositories that support more reliable forecasts
- Define AI-aware project profiles and productivity baselines

- Explicitly account for AI-related effort and cost drivers that are not fully explained by functional size alone

This uTip is not intended to redefine counting practices or provide new counting rules. For counting rules, the relevant IFPUG manuals remain the reference.

Guiding principles

1. **Keep Function Points as the reference for functional size and scope.** Function Points continue to provide an objective, verifiable and repeatable basis for estimating and comparing user functionality across traditional and AI-enabled delivery models.
2. **Use SNAP to capture non-functional scope.** SNAP can provide an objective measure of non-functional user requirements and related technical complexity that may materially affect effort, cost and schedule.
3. **Do not force all AI work into a single sizing metric.** Some AI-related work is not by definition functional size alone. Estimation should remain transparent by identifying these cost and effort drivers explicitly, instead of hiding them in generic assumptions.
4. **Treat productivity as an empirical variable.** AI may reduce effort, but not in a uniform way. Productivity assumptions should be evidence-based, project-profile-specific, and updated as experience grows.

What should be estimated?

Functional scope – Use IFPUG Function Points to size user-recognizable functionality and externally observable behavior.

Non-functional scope – Use SNAP to measure relevant non-functional user requirements and related technical complexity.

Additional AI-related effort and cost drivers – Where relevant, explicitly estimate effort and costs associated with activities such as data creation and normalization, evaluation and refinement of AI outputs and knowledge assets, prompt and tool configuration, model tuning and production monitoring. The objective is to maintain visibility of significant effort and cost drivers that may not be adequately represented by functional or non-functional size alone.

Why historical repositories matter

Effective estimation requires historical baselines and project profiles that include size, schedule, effort, productivity, quality, staff and risk. This remains true in all software environments, but it becomes even more critical when AI affects delivery rates in uneven and evolving ways.

AI may significantly affect software productivity, but the magnitude and nature of these effects vary across projects and continue to evolve. A practical estimation repository should capture, at minimum, functional size, non-functional size, effort, duration, cost, defects, team size, project type, technology platform, sourcing model, software languages, organizational maturity level, development methodology used and lifecycle coverage.

In AI-driven environments, the repository should also capture selected AI-specific descriptors such as the level of AI assistance, use of coding assistants or agentic workflows, type of model or service used, effort spent on evaluation or prompt refinement, degree of human review, and whether the work mainly involved developing the AI capability itself or integrating it into a production-ready solution.

Historical data is only useful if it is normalized. Organizations should apply consistent rules regarding what effort is included, how size is measured, which lifecycle activities are counted, and how projects are classified.

Recommended Approach for Classifying Projects into Isoproductive Clusters

When measuring productivity, delivery rate, defect density or other project-level metrics, care should be taken to ensure that the metrics are calculated for comparable, or isoproductive, project clusters. Otherwise, comparisons across projects may be misleading and forecasts based on those comparisons may not be reliable.

Two complementary approaches can be used to classify projects into productivity clusters: a qualitative approach based on project characteristics and expert judgment, and a quantitative approach based on historical project data. The qualitative approach can be used initially to define candidate clusters, which can then be evaluated and refined using quantitative evidence.

Define project profiles and risk factors – Before starting a project, classify it into an expected productivity cluster such as low, average, or high-productivity, where these categories are defined according to the organization's historical data and forecasting objectives, based on relevant project-profile and risk factors. These may include requirements maturity and volatility, team capability, architectural uniqueness/complexity, security requirements, AI usage, governance maturity, dependence on external models or APIs, and review intensity. Other relevant factors may include process maturity, customer characteristics, technology risk, project size and complexity, and an overall allowance for significant unknowns.

Different combinations of project-profile characteristics may lead to the same productivity cluster. The factors used for classification should therefore be selected carefully, avoiding redundant or strongly correlated variables where this could distort the analysis.

Collect normalized historical data -- As projects are executed and completed, collect their actual productivity data, expressed as the relationship between size and effort. Depending on the measurement and forecasting purpose, productivity may be expressed as size per unit of effort (e.g., FP/hour) or as the reciprocal effort rate (e.g., hours/FP). Over time, the repository should accumulate productivity observations from multiple projects. Historical data should be normalized before being used for comparison or clustering. Consistent rules should be applied regarding what effort is included, how size is measured, which lifecycle activities are covered, and how projects are classified. The measurement standard used for functional size and related versions (e.g. IFPUG CPM 4.3.1 or SPM 2.2) should also be recorded and considered when comparing productivity data.

Identify candidate clusters -- Using the historical productivity observations, organizations can assess the variability within each candidate cluster. Where the data support such an analysis, the mean productivity, standard deviation, and coefficient of variation (CV = standard deviation / mean) can be used to characterize the observed distribution. The acceptable level of variation should be defined according to the organization's data and forecasting objectives rather than treated as a universal threshold. For example, an organization may initially use a target CV as a practical criterion for assessing whether a cluster is sufficiently homogeneous, and refine that criterion as more historical data become available. Outliers and unusual observations should be investigated to determine whether they are explained by identifiable special causes. Where appropriate, observations may be reassigned to another cluster or analyzed separately.

Validate the clusters using historical evidence -- For each candidate productivity cluster, analyze the observed productivity distribution and compare it with the distributions of other clusters. Appropriate statistical tests can be used to determine whether the observed differences between clusters are supported by the available data .

If two candidate clusters show no statistically meaningful difference in productivity, the data do not support treating them as separate productivity clusters and they should be considered for merging. If the observed differences are supported by the data, the clusters can be retained as distinct productivity profiles.

An **alternative quantitative approach** is to use an unsupervised clustering technique, such as *k-means*, to identify groups of projects with similar productivity characteristics. Depending on the available data and the purpose of the analysis, clustering may use productivity measures alone or combine them with relevant project-profile characteristics. The resulting clusters should then be interpreted and validated against the underlying project characteristics and forecasting objectives. This approach may be useful when sufficient historical data are available and the organization wants to reduce reliance on predefined qualitative categories. Comparisons should also account for the functional-size measurement standard used (e.g. CPM 4.3.1 or SPM 2.2).

As an **additional validation check**, organizations may examine the **relationship between size and effort** within each cluster, for example using correlation or regression analysis. A strong relationship can provide supporting evidence that the cluster is internally consistent. However, no single statistical threshold should be considered universally applicable; the appropriate criterion should depend on the data quality, sample size, and characteristics of the projects in the cluster.

Assign new projects to the most appropriate cluster – Before project execution, classify the new project into the most appropriate productivity cluster based on its project profile, technology, delivery context, AI usage, and other relevant characteristics. The corresponding productivity baseline can then be used to produce the initial estimate.

After the project is completed and actual productivity is known, assign the project to the cluster that best reflects its observed productivity and project characteristics. This new observation can then be used to validate or recalibrate the cluster .

Recalibrate the clusters periodically -- Productivity clusters should not be considered permanent. As new project data become available, organizations should periodically recalculate the cluster productivity statistics and verify whether the observed distributions remain stable and sufficiently homogeneous .

The organization should also investigate material changes, outliers, or special causes and reassess whether existing clusters remain appropriate. Where the data support it, new clusters may be created or existing clusters merged.

This continuous recalibration helps ensure that

- productivity data remain sufficiently stable for the intended forecasting use
- project profiles remain meaningfully associated with productivity behavior
- productivity-based forecasts can be progressively improved as additional evidence becomes available.

Context factors that shape AI-related productivity assumptions

Productivity assumptions for AI-assisted delivery should be calibrated against the actual delivery context. The evidence from the literature reviewed for this work suggests that at least seven core factors can materially influence the productivity gains that organizations can realistically expect.

- **Organizational scale and governance –** The size of the delivery organization influences how much AI gains at task level can translate into end-to-end gains.
- **Process maturity and delivery discipline** – The maturity of development and control practices affects whether AI gains remain local or become repeatable across the lifecycle. The effect of AI adoption may depend strongly on

the maturity of existing development and control practices; weak process foundations may limit the extent to which task-level productivity gains translate into end-to-end delivery gains.

- **AI adoption maturity –** There is a significant difference between isolated use of public AI tools and systematic enterprise adoption supported by governance, approved platforms, and repeatable practices. The more mature the adoption, the more stable the productivity baseline will be.
- **Project starting point: greenfield or brownfield** – Greenfield initiatives may provide more favorable conditions for AI-assisted delivery, while brownfield initiatives may introduce additional verification, legacy, dependency, and integration effort.
- **Task difficulty and novelty –** AI generally performs better on standard, repetitive, manually intensive or commonly performed tasks than on highly novel, research-like, or architecturally critical work. The more complex the task, the less safe it is to assume large productivity improvements.
- **Team experience and role mix –** The impact of AI is highly dependent upon the role and experience level of the users. Less experienced developers may benefit strongly on coding support, while more experienced professionals may gain more in upstream activities such as requirements clarification, design, review, and governance since more focused on "Validation" and quality of the output.
- **Technology stack and AI affinity** – The technology context matters. The technology context may affect the extent to which current AI tools can be effectively applied. Differences in tool support, available training data, integration capabilities, and legacy constraints may influence the resulting productivity gains.

These factors can be used to calibrate productivity assumptions, scenario ranges, and historical project clustering.

Additional estimation considerations

The following drivers are project-specific factors that may affect the relationship between size, effort, cost, schedule, and risk . These drivers can be used to improve forecasting assumptions and historical calibration.

- **Requirements quality and volatility –** Ambiguous, incomplete, or volatile requirements can reduce the effectiveness of AI assistance and increase rework, validation effort, and estimation uncertainty.
- **Data and knowledge readiness –** AI-enabled solutions often depend on the availability, quality, structure, and accessibility of data, knowledge bases, FAQs, context documents, or training examples.
- **Integration and dependency complexity –** Effort may increase significantly when AI capability must be embedded into existing systems, APIs, workflows, identity services, logging, monitoring, or enterprise platforms.
- **Review and validation effort –** The expected level of human review, verification, testing, explainability, and acceptance control can absorb part of the apparent AI productivity gain.
- **Quality expectations and defect tolerance –** Projects with high reliability, security, safety, auditability, or low-defect expectations may require additional controls and may experience lower net productivity gains.
- **AI tooling and platform readiness –** Approved tools, enterprise platforms, secure access to code and knowledge assets, and standard usage policies make AI adoption more repeatable and measurable.
- **Lifecycle effort mix –** The end-to-end impact depends on the relative weight of lifecycle phases. A coding-heavy project may benefit differently from a project dominated by integration, testing, governance, or deployment.
- **AI cost and consumption model –** Licensing, token consumption, model usage, compute cost, and specialist professional fees may affect the cost forecast even when effort is reduced.

Use of productivity baselines in estimation

Productivity baselines should be used as empirical references for forecasting, not as fixed universal rates. In AI-enabled environments, productivity may vary significantly across teams, technologies, organizational practices, AI adoption maturity, and project profiles. For this reason, baselines should be defined and used within comparable or isoproductive clusters, and should be expressed as ranges or probabilistic forecasts whenever possible.

A baseline can be considered mature enough for estimation when the underlying cluster is clearly defined, the data are normalized, and the observed variation appears stable enough to support forecasting. This means that the organization should understand which lifecycle activities are included in effort, how size has been measured, what project types belong to the cluster, and whether the data show special-cause variation that would make the baseline unreliable.

Statistical maturity of the baseline – Organizations should check whether the number of observations is sufficient, whether the mean and standard deviation are reasonably stable over time, whether the coefficient of variation is acceptable for the intended forecasting use, and whether outliers or special causes have been investigated.

Use of probability-based forecasts – When data are mature enough, estimates should be expressed with a selected confidence level or probability of achievement rather than as a single point. For example, an organization may decide to use the mean productivity for planning and a more conservative percentile for contractual agreements.

Periodic recalibration – AI technologies, tools, models, and practices evolve rapidly. Productivity baselines should therefore be recalibrated periodically and whenever material changes occur, such as adoption of a new AI platform, model upgrade, change in development process, change in team composition, or introduction of agentic workflows.

Initial reference values – When organizational data are not yet sufficient, initial productivity targets and reference ranges may be derived from market benchmarks, such as ISBSG datasets or other credible benchmark sources. These external values should be treated as temporary calibration references until sufficient internal data become available. They may be useful for early, high-level order-of-magnitude estimates, but should not be treated as organization-specific productivity baselines.

Versioning of baselines – Organizations should keep track of baseline versions and the assumptions behind them. This is particularly important in AI-enabled delivery, where productivity may improve or deteriorate as tools, models, practices, and governance mechanisms change.

In practice, the estimation process should start with the best available external or internal reference, classify the project into the most appropriate productivity cluster, apply context and risk adjustments, and then recalibrate the baseline as actual project evidence becomes available and reaches sufficient statistical relevance.

AI-specific productivity baselines

Organizations should not assume that one generic productivity rate remains valid across traditional development, AI-assisted development, and AI-intensive or agentic delivery.

A practical approach can be to maintain at least three categories of productivity baseline:

- traditional baseline: little or no AI support;
- AI-assisted baseline: AI supports coding, documentation, testing, or analysis while the lifecycle remains human-led;

- AI-intensive or agentic baseline: AI contributes materially across multiple phases and human effort shifts toward orchestration, review, exception handling, and governance.

These baselines may be represented through different related metrics, such as effort per Function Point, cost per Function Point, Function Points delivered per calendar month, and, where appropriate, effort per SNAP Point. The selected metric should be applied consistently within the relevant historical dataset and forecasting purpose. It is also useful to distinguish between effort for developing the AI capability itself and effort for integrating that capability into a production-ready solution, because integration often requires additional controls, production hardening, security, monitoring, and operational support. When comparing these metrics across projects, the measurement approach used for functional size should be consistent or explicitly accounted for.

Recommended estimation approach

Measure the functional scope – Use IFPUG Function Points to quantify the business functionality to be delivered.

Measure the non-functional scope – Use SNAP for non-functional user requirements and related technical complexity.

Identify additional effort drivers explicitly – Where AI-related work materially affects cost or schedule, keep it visible in the estimate rather than absorbing it into generic productivity assumptions.

Classify the project profile – Position the project by technology, business domain, sourcing model, architecture, and level of AI involvement.

Calibrate the productivity baseline using context factors – Adjust the baseline considering organizational scale, process maturity, AI maturity, greenfield/brownfield context, task difficulty, team experience, technology stack, and any additional estimation drivers that materially affect the project.

Adjust for risk and capability – Consider requirements maturity, team capability, governance, data readiness, integration complexity, quality expectations, AI cost model, and operational constraints.

Produce range estimates and re-forecast during execution – Prefer optimistic, most-likely, and pessimistic scenarios over single-point estimates /commitments.

Practical numeric example

The following example is illustrative. The productivity baselines are assumptions created to explain the estimation approach. They are not official benchmark values and should not be treated as industry standard data or averages.

Assume an enhancement project has been sized at 220 Function Points for the functional scope and 85 SNAP Points for the non-functional scope. Assume also that the organization maintains separate productivity baselines for traditional delivery, AI-assisted delivery, and different project profiles. For this illustrative example, the 220 FP should be understood as measured using either CPM or SPM; the same measurement approach should be used for the productivity baselines applied.

- Traditional functional effort rate = 3.8 hours/FP
- Traditional non-functional effort rate = 0.6 hours/SNAP Point
- AI-assisted functional effort rate (most likely) = 2.9 hours/FP
- AI-assisted non-functional effort rate (most likely) = 0.5 hours/SNAP Point
- Optimistic AI-assisted functional effort rate = 2.7 hours/FP

- Optimistic AI-assisted non-functional effort rate = 0.4 hours/SNAP Point
- Pessimistic AI-assisted functional effort rate = 3.3 hours/FP
- Pessimistic AI-assisted non-functional effort rate = 0.55 hours/SNAP Point

Scenario summary

Scenario	Functional effort	Non-functional effort	Total human effort	Human Cost @75$/h
Traditional	836.0 h	51.0 h	887.0 h	66,525 $
AI-assisted (most likely)	638.0 h	42.5 h	680.5 h	51,038 $
AI-assisted (optimistic)	594.0 h	34.0 h	628.0 h	47,100 $
AI-assisted (pessimistic)	726.0 h	46.8 h	772.8 h	57,956 $

The human cost shown in this example is calculated as total human effort multiplied by the assumed labor rate of $75/hour.

AI consumption costs are not included in these figures. In an actual AI-enabled project, AI-related costs should be estimated and tracked separately and combined with effort cost to obtain the total or augmented project cost.

AI consumption cost may not be linearly related to effort. Therefore, reducing effort does not necessarily result in a proportional reduction in total project cost.

Specifically, the total project cost should take into account the following components: Effort Cost + AI Consumption Cost , where Effort = Effort Hours × Labor Rate

At present, there may be insufficient empirical evidence to establish a reliable relationship between AI consumption cost and FP or SNAP Points. Organizations should therefore collect AI usage and cost data alongside size, effort, schedule, quality, and project-profile data.

Practical recommendations

- Keep Function Points as the primary basis for sizing functional scope.
- Use SNAP systematically when non-functional complexity is material.
- Keep significant AI-related effort and cost drivers visible in the estimate.
- Use project profiles and historical data to select appropriate productivity baselines.
- Assess the maturity and statistical stability of productivity baselines before using them for commitments.
- Recalibrate AI-aware productivity baselines periodically as tools, models, practices, and governance evolve.
- Prefer range estimates and reforecast during execution as actual evidence becomes available.

Summary

AI is changing software delivery, but it has not removed the need for objective measurement. On the contrary, increasing variability in delivery performance makes consistent sizing, historical data, and evidence-based productivity assumptions even more important .

Function points provide the basis for measuring functional scope, while SNAP can be used to capture relevant non-functional scope. AI-related effort and cost drivers should be identified explicitly rather than hidden within generic productivity assumptions.

Reliable forecasting in AI-enabled environments therefore depends on AI-aware historical repositories, comparable project profiles, calibrated productivity baselines, explicit risk and cost assumptions, and continuous reforecasting as project evidence becomes available.

Frequently asked questions (FAQ)

1. Should AI-generated code reduce the Function Point count?

No. Function Points measure delivered functional user requirements, not the amount of code written manually or with AI assistance. AI affects effort and productivity, not the definition of functional size.

2. Is SNAP useful in AI-driven projects?

SNAP can be particularly useful when non-functional requirements materially affect effort, cost, schedule, or risk. AI-driven solutions may introduce additional architectural, operational, security, data, monitoring, and quality requirements that should be considered explicitly.

3. How should model training, fine-tuning, prompt refinement, or data preparation be handled in estimation?

They should be treated explicitly as estimation drivers where relevant. The purpose of this uTip is not to define new counting rules for them, but recommends that they should not be ignored when they materially affect cost or schedule.

4. Should the same productivity rate be used for all projects after AI adoption?

No. AI productivity effects vary significantly depending on task complexity, developer capability, tool usage, governance, and delivery context. Separate productivity profiles are recommended.

5. Can AI help produce Function Point estimates?

Yes. AI can support and accelerate estimation activities, including extraction of likely Function Point counts from natural language requirements, but human methodological oversight remains necessary.

6. Should estimates be expressed as a single number or as a range?

A range is preferable, especially in AI-driven environments where the actual effect of automation and rework may differ from early assumptions.

7. Should additional context factors be used as formal multipliers?

Not necessarily. They should primarily be used to guide calibration, scenario analysis, repository design, and estimation discussion. Organizations may later turn them into local factors if supported by historical evidence.

8. When is a productivity baseline mature enough to use for estimation?

A baseline is more reliable when it belongs to a clearly defined project cluster, is based on normalized data, and the available statistical evidence supports its stability and relevance for the intended forecasting purpose. Until then, external benchmarks and conservative ranges may be used as temporary references.

9. Should AI consumption costs be included in the estimate?

AI consumption costs should be explicitly tracked as a separate cost component from human effort. Although current evidence is still insufficient to establish a measurable correlation between AI consumption and Function Points or SNAP Points, organizations are encouraged to collect AI usage data together with size, effort, cost, schedule, quality, and project profile information. Over time, this may support the development of empirical estimation models and indicators such as AI cost per FP/SNAP Point, tokens per FP/SNAP Point, and "augmented cost" (human + AI) per Function Point that can help explain the value produced per invested money.

Further reading

1. Function Point Counting Practices Manual (CPM and SPM IFPUG).
2. Software Non-Functional Assessment Process (SNAP), IFPUG Standards.
3. The IFPUG Guide to IT and Software Measurement.
4. Impact of AI-Assisted Development on Software Productivity and Delivery Speed (ISBSG, 2026).
5. Measuring the Immeasurable: An In-Depth Look at AI Metrics (MetricViews, Spring 2024).
6. Measuring Intelligence: Function Point Analysis in AI-Powered Chatbot Solutions (ISMA 2025 presentation).
7. Non-Functional Measurement – AI-Powered Chatbot (ISMA 2025 presentation).
8. Function Point Analysis for AI Agents: Measuring and Estimating AI Functionality (ISMA 2025 presentation).