

# Developer Thriving

## Four Sociocognitive Factors That Create Resilient Productivity on Software Teams

Catherine M. Hicks, Carol S. Lee , and Morgan Ramsey, Pluralsight

*// We present a research-based framework for measuring successful environments on software teams for long-term and sustainable sociocognitive problem-solving. Across 1,282 full-time developers in 12+ industries, we tested the factors of our framework and found it predictive of developers' self-reported productivity. //*



**TO CREATE NEW** technologies, developers must collaborate well on complex code in an iterative and

distributed manner. Developers and their teams also need to balance personal productivity, project constraints,

organizational context, and business impact alongside pushing the boundaries on what code can do in the world. Against this complexity, some estimates of the overall success rates of software projects claim that the *majority* of software projects deliver late, deliver out of scope of planned budgets, and fail to drive business impact.<sup>1</sup>

### Introduction

How can software teams thrive in the face of the unexpected and unplanned difficulties of software projects? This study contributes a theoretically grounded model for the workplace-specific *sociocognitive* drivers of developer thriving that promote productivity. We believe this model is an important tool for leaders and teams who seek to better define tractable and attainable targets for interventions and wish to include developers' well-being and needs as knowledge workers as they make decisions about team and organization design to yield productivity outcomes

### Background Research

Understanding what unlocks high-quality problem-solving for developers is key to maintaining software innovation. Nevertheless, software engineering as a field does not have a consensus on the measurement of developer productivity.<sup>2</sup> Absent a clear shared definition, working teams and engineering leadership frequently fall into one of the two measurement traps when attempting to define and subsequently increase developer productivity inside of their engineering functions, which lead to different behavioral outcomes:

1. Fixating on surface definitions of productivity and measuring and incentivizing the wrong things. Outcomes may include limited or punitive production measures,

- such as counting lines of code, and failures to account for tech debt.
2. Becoming paralyzed by complexity and measuring and incentivizing nothing. Outcomes may include poor organizational practices such as relying heavily on interpersonal coordination<sup>3</sup> or gut instinct in technical decision-making and evaluation of performance, which is subject to many potential biases.

While these two maladaptive scenarios for engineering organizations may seem opposing, they both spring from a foundational lack of clarity on what truly helps developers to achieve long-term success in sustaining productivity and, therefore, how to measure and incentivize it.

However, previous evidence on the factors that impact developers' achievement provide a starting path through these measurement traps. Software researchers have called out the need to develop new models of human-centered developer productivity by 1) investigating the sociocognitive factors that improve problem-solving during coding and software work overall, 2) doing research directly on the real-world experiences of modern software teams, and 3) avoiding major misconceptions in measuring productivity, such as defining developer productivity only as crude output measures such as lines of code or setting a single metric goal and using it as a threshold to evaluate all software work regardless of differing contexts and needs.<sup>4,5,6</sup>

The Satisfaction, Performance, Activity, Communication and Collaboration, and Efficiency and Flow (SPACE) framework, which characterized *developer productivity* in terms of satisfaction and well-being, performance, activity, communication, and efficiency, is one recent example of software research

that proposes a multivariate theory that includes psychological drivers of developer productivity.<sup>5</sup> The SPACE framework provides possible examples of systematically broadening "productivity" definitions with dimensions such as job satisfaction. Nevertheless, while the SPACE framework definition includes satisfaction and well-being as key pieces of productivity, it does not provide psychometric evidence for how to measure and evaluate developer satisfaction and well-being at scale. Psychometric evidence is defined as quantitative evidence about underlying latent variables that are believed to drive psychological processes. Because psychological constructs are not directly observed, reliable estimations of psychological constructs rely on developing measurements that use empirical approaches such as the use of validated items, reliability reporting, and adherence to previously validated constructs.<sup>7</sup> A psychometrically evidenced alternative to focusing on a general "satisfaction" construct is

developer thriving, the process of sustainable growth and development.<sup>8</sup>

### A Framework for Developer Thriving

The developer thriving framework presented in this study is adapted from previously validated models of thriving in psychosocial and workplace contexts and consists of four factors: developer agency, developer motivation and self-efficacy, developer learning culture, and developer support and sense of belonging (Table 1).<sup>8</sup>

The framework builds on the known connection between developer satisfaction and productivity to answer *what* drives satisfaction in the first place. Notably, the four factors in developer thriving explicitly call attention to the *environmental and structural affordances* that create the conditions for developer productivity. One central criticism of relying on happiness and satisfaction measures is that they are highly volatile and may capture immediate signals (e.g., mood) rather than true overtime productive

**Table 1. The four factors of developer thriving.**

The Behavioral Science Behind Developer Thriving	
Agency	A developer 1. is able to voice disagreement with team definitions of success 2. has a voice in how their contributions are measured.
Motivation and self-efficacy	A developer 1. is motivated when working on code at work 2. can see tangible progress most of the time 3. is working on the type of code work they want to work on 4. is confident that even when working in code is unexpectedly difficult, they will solve their problems.
Learning culture	A developer is 1. learning new skills as a developer 2. able to share the things they learn at work.
Support and belonging	A developer 1. is supported to grow, learn, and make mistakes by their team 2. agrees they are accepted for who they are by their team.

*See full details in the supplementary materials available at <https://doi.org/10.1109/MS.2024.3382957>.*

patterns.<sup>9</sup> In contrast, the four socio-cognitive factors of developer thriving each tie explicitly to behavioral cycles that maintain high performance over time using trait-based (rather than state-based) measures adapted from previously validated measures, thus yielding a more long-term and sustainable measure.<sup>7,8,10</sup>

### What Might Increase Developer Thriving? Visibility and Healthy Metrics Use

Along with developing new original measures for *what* key factors play into developer thriving, we

This impact on developer motivation was also a key theme underlined by both individual contributor developers in the pilot testing for this study, who described expecting and anticipating moments of recognition as key motivators, and by managers, who described a pivotal responsibility of making their team's work visible.

Increased measurement leading to positive outcomes echoes a significant body of research in the clinical and behavioral sciences, which indicates that we tend to forget or lack awareness of the amount of work we

to positive behavior change for both output and well-being.<sup>14</sup>

Based on this previous literature, we developed the following six key hypotheses:

- H1. HMU will be positively associated with perceived productivity.
- H2. Visibility and value of work will be positively associated with perceived productivity.
- H3. Developer thriving will be positively associated with perceived productivity.
- H4. Visibility and value of work will be positively associated with productivity *through* developer thriving (mediation).
- H5. HMU will be positively associated with perceived productivity *through* developer thriving (mediation).
- H6. HMU will be positively associated with perceived productivity *through* visibility and value of work (mediation).

The framework builds on the known connection between developer satisfaction and productivity to answer *what* drives satisfaction in the first place.

also wanted to ask what leaders and organizations can do to increase developer thriving. Based on the literature, we identified two potential factors: visibility and value of work and healthy metrics use (HMU).

The unique benefit of both expecting and planning for visibility, and getting feedback from a visibility cycle, is supported across scientific evidence on human well-being, health sciences, and organizational psychology. For example, research on behavioral change in healthcare settings highlights the value created from recognition and visibility as one of the strongest predictors of behavioral engagement, performance, and productivity of both individuals and team members.<sup>11,12</sup>

have done, leading us to devalue and minimize our progress. Tracking behavioral and psychological processes has been shown to mitigate this effect by providing us concrete evidence of our progress and accomplishments. Having this evidence not only increases mindful attention and awareness but also increases our sense of value and mastery over our work, increases empathy and self-compassion, boosts coping abilities and distress tolerance, empowers us to recognize and set boundaries, and drives behavioral engagement for both groups and individuals.<sup>13</sup> With developers specifically, research has found that self-reflection in a repeated cadence increased developers' awareness of their habits and led

## Methods

Our study consisted of a large-scale quantitative survey. This section discusses the quantitative survey measures (the “Survey Measures” section), survey sample recruitment and description (the “Survey Sample Recruitment and Description” section), and survey analysis approach (the “Survey Analysis Approach” section).

### Survey Measures

Throughout the quantitative measures on this survey, participants answered using a Likert-type scale. Unless otherwise indicated (see Table 2), the scale included a neutral midpoint; scores greater than 3 indicate agreement, while scores less than 3 indicate disagreement. For each multi-item measure, the items were averaged to

create a single composite score for each measure.

*Perceived productivity rating (PPR):* There is no standard measure for developer productivity,<sup>6</sup> and developers define productivity in multiple ways; software research has therefore frequently used self-report ratings of productivity.<sup>14</sup> To operationalize this complex concept, we also chose to ask developers to rate their own productivity over a recent period of time. This approach allows us to let developers summarize across their complex contexts, different industry paces of work, and working environments. In our study, the PPR is a self-report, single-item measure. In keeping with our aim of reducing within-survey response effects, this question was shown first to reduce biases that might arise from respondents' reflecting on questions about belonging, measurement, and software metrics.

*HMU:* HMU was operationalized as a two-item composite rating created for the purpose of this study. The first item asked participants to report their team's use of metrics. The second item asked participants to report if they believed their team used the "right" metrics for their team and agreed that "they measure the right things."

*Developer thriving scale (DTS):* The DTS is a ten-item measure created for the purpose of this study, abbreviated in order to be accessible to participants at scale in an applied research setting (see full details in the supplementary materials available at <https://doi.org/10.1109/MS.2024.3382957>). The measure draws from models of thriving in health and psychology to identify four primary factors: motivation and self-efficacy, support and belonging, learning culture, and agency. The items for each factor are adapted from empirically

validated psychological measures of these constructs. The measure had good internal consistency in our sample ( $\alpha = 0.86$ ).

*Visibility and value questionnaire (VVQ):* The VVQ is a three-item measure created for the purpose of this study. The measure draws from previous research indicating that recognizing and valuing employees' work predicts employee satisfaction and asks respondents to rate the

extent to which they believe their technical work is visible and valued by teammates and managers. The measure had good internal consistency in our sample ( $\alpha = 0.83$ ).

We pilot-tested the clarity of our survey by seeking feedback from five full-time software engineers within our organization. To reduce response biases within survey, we used a semi-randomized survey design. All participants answered key construct

**Table 2. Gender and sexual orientation.**

Demographic	Statistic (N = 1,282)
Gender (n = 683)	
Female	121 (17.72%)
Male	527 (77.16%)
Nonbinary/fluid/queer/gender queer	10 (1.46%)
Prefer not to answer	25 (3.66%)
Self-identify	0 (0%)
Transgender identity (n = 593)	
Yes	22 (3.71%)
Prefer not to respond	47 (.93%)
Sexual orientation (n = 537)	
Asexual/aromantic	14 (2.61%)
Bisexual	16 (2.98%)
Fluid	3 (0.56%)
Gay	11 (2.05%)
Lesbian	4 (0.74%)
Pansexual	3 (0.56%)
Queer	2 (0.37%)
Questioning/unsure	7 (1.30%)
Self-identify	3 (0.56%)
Straight/heterosexual	369 (68.72%)
Prefer not to respond	105 (19.55%)

measures *before* being asked to answer measures that may influence their responses. For example, to avoid stereotype threat, participants rated their productivity *before* being asked to answer any questions about demographic characteristics. Within the key construct measures (Table 1), the order of presentation was randomized to control for order effects.

### Survey Sample Recruitment and Description

We utilized snowball sampling and advertised our online Qualtrics survey publicly on various social media (e.g., X [formerly known as Twitter], Facebook, Mastodon, LinkedIn, and Reddit) from researchers' personal social media accounts and via direct e-mails to professional listservs of interest to developers. Our survey was also advertised inside the Pluralsight Skills platform, embedded in developer-relevant content pages such as internal learning programs, and inside the Pluralsight Flow platform,

shown as a banner advertisement to professional developer users. In all cases, this survey advertisement was optional and not connected to user data on either of these platforms. All participants provided consent and were informed of the Developer Success Lab's consent and participant privacy policies. Because our study design consisted of a survey study with a sample of adults who provided consent to participation and whose data were anonymized, it was determined to be exempt from the requirement to be reviewed by an Institutional Review Board (Category 2).

Our survey was open to all full-time individual contributor developers and software engineers responsible for technical code work in their role. We recruited a total of 1,409 individual

contributor developers. Of the 1,409 participants, 121 did not move past the first two questions of the survey, and 6 were removed for writing identity-based discriminatory responses in our open text demographic fields. Our final sample consisted of 1,282 participants. A summary of demographic and firmographic characteristics can be seen in Tables 2–6. As a token of appreciation for participation, our research team made a donation to an open source software nonprofit, chosen based on participant voting.

### Survey Analysis Approach

To test the normality of our variables, we obtained skew and kurtosis values of all variables. All variables were within normal ranges. As such, we performed a Pearson's correlation

**Table 4. Race.**

Demographic	Statistic (N = 1,282)
Race <sup>a</sup> (n = 576)	
Alaskan Native/Native American/Indigenous	5 (0.87%)
Black/African American	25 (4.34%)
East Asian	47 (8.16%)
Middle Eastern/North African (non-White)	12 (2.08%)
Middle Eastern/North African (White)	18 (3.13%)
Latinx/Hispanic (non-White)	19 (3.30%)
Latinx/Hispanic (White)	33 (5.73%)
Pacific Islander/Native Hawaiian	2 (0.35%)
South/ Southeast Asian	105 (18.23%)
White	245 (42.53%)
Multiracial	13 (2.26%)
Self-identify <sup>b</sup>	20 (3.47%)
Prefer not to respond	69 (11.98%)
Education (n = 614)	
Grade school	8 (1.30%)
Some high school	5 (0.81%)
High school diploma	29 (4.72%)
Some college	40 (6.51%)
Community or vocational	21 (3.42%)
Four-year college	283 (46.09%)
Graduate degree	228 (37.13%)
Years of coding experience	
Mean (SD)	14.4 (12.8)

<sup>a</sup>Participants could check all that apply.

<sup>b</sup>Examples include Indian or Black. We chose to be holistic, and for some respondents, this may have felt too limiting or broad.

to check the correlations among all variables (see Table 7).

To identify covarying demographic and firmographic variables, we conducted a series of correlations among our primary variables of interest and our continuous demographic and firmographic variables. To avoid potentially misleading signals resulting from the reduced sample size and highly skewed distributions across categorical identity questions, we did not examine the relations among our primary measures and our categorical demographic and firmographic variables. Years of coding experience were positively associated with the VVQ [ $r(701) = 0.20, p < 0.01$ ], DTS [ $r(472) = 0.21, p = 0.001$ ], and PPR [ $r(710) = 0.25, p < 0.001$ ]. Additionally, the percent of time spent writing code was positively associated with the HMU [ $r(719) = 0.09, p < 0.05$ ], PPR [ $r(730) = 0.20, p < 0.001$ ], DTS [ $r(494) = 0.27, p < 0.001$ ], and VVQ [ $r(722) = 0.16, p < 0.001$ ]. There were no other significant effects among our primary measures and the other firmographic and demographic variables ( $p = 0.07\text{--}0.84$ ). As such, we controlled for the effect of the percent of time spent writing code and years of coding experience in our analyses.

To test hypotheses 1–6, we conducted a serial mediation path analysis (see Figure 1 for model structure). This allowed us to simultaneously test if the HMU, VVQ, and DTS were positively associated with the PPR (hypotheses 1–3). This statistical model also allowed us to test if the VVQ was positively associated with productivity *through* or *because* of developer thriving (mediation; hypothesis 4) and if the HMU was positively associated with perceived productivity *through* or *because* of the DTS and the VVQ (mediation; hypotheses 5–6).

## Results

### Findings

From previous research,<sup>5</sup> a positive developer experience and increasing

developer satisfaction are suggested to be among the best ways to increase developer productivity. Our study aimed to provide more actionable and measurable factors in

**Table 5. Industry and organization characteristics.**

Firmographic	Statistic ( $N = 1,282$ )
Industry	
Education	32 (2.5%)
Energy	15 (1.2%)
Financial services	153 (11.9%)
Government	41 (3.2%)
Healthcare and pharmaceuticals	39 (3.0%)
Industrials and manufacturing	36 (2.8%)
Insurance	23 (1.8%)
Media/entertainment	13 (1.0%)
Nonprofit	5 (0.4%)
Retail/consumer/e-commerce	43 (3.4%)
Technology	249 (19.4%)
Telecommunications	24 (1.9%)
Other	64 (5.0%)
Missing	545 (42.5%)
Organization size	
1–4	26 (2.0%)
5–9	17 (1.3%)
10–19	29 (2.3%)
20–99	72 (5.6%)
100–499	100 (7.8%)
500–1,999	92 (7.2%)
2,000–4,999	61 (4.8%)
5,000–9,999	49 (3.8%)
10,000+	272 (21.2%)
Missing	564 (44.0%)

**Table 6. Team and role characteristics.**

Firmographic	Statistic ( <i>N</i> = 1,282)
Team size	
Mean (SD)	8.42 (7.06)
Cross-functional team	
Yes	494 (38.5%)
Maybe/not sure	144 (11.2%)
No	139 (10.8%)
Missing	505 (39.4%)
Percent of time spent writing code	
Mean (SD)	60.1 (24.1)
Engineering area	
Backend	468 (36.5%)
Frontend	285 (22.2%)
Full stack	305 (23.8%)
Mobile	71 (5.5%)
Database admin	125 (9.8%)
System admin	95 (7.4%)
DevOps	234 (18.3%)
Site reliability	80 (6.2%)
Other engineering area	94 (7.3%)
Missing	530 (41.3%)

**Table 7. Overall descriptives of key measures.**

Variable	Mean (SD)	Skewness	Kurtosis	HMU	VVQ	DTS	PPR
HMU, <i>n</i> = 958	1.23 (0.52)	0.03	-1.15	1.00	-	-	-
VVQ, <i>n</i> = 821	3.99 (0.91)	-0.86	.27	.33*	1.00	-	-
DTS, <i>n</i> = 562	4.26 (0.61)	-0.83	0.81	.34*	.73*	1.00	-
PPR, <i>n</i> = 1,280	3.45 (0.92)	-0.28	0.04	.26*	.41*	.43*	1.00

\**p* < 0.001.

developer experience than simple satisfaction and looked at whether the factors in developer *thriving* are predictive of productivity. Further, this study asked if implementing team-level tools and processes such as healthy metrics and increased visibility could improve developer thriving and productivity, even after controlling for factors like years of experience and time spent coding.

To test our hypotheses, we conducted a linear regression-based serial mediation path analysis with the HMU as our independent variable, VVQ as our first mediator, DSS as our second mediator, and PPR as our outcome variable (saturated model using only observed variables; CFI = 1; RMSEA = 0). Additionally, we entered the percent of time spent coding and years of coding experience as covariates for all variables, given their significant associations with our mediators and outcome variable.

With all variables in the model, we found that HMU, visibility and value of work, and developer thriving were all significantly associated with perceived productivity (hypotheses 1–3), with developer thriving having the strongest effect on perceived productivity. Notably, thriving also mediated the relations between perceived productivity and both HMU and visibility and value of work, indicating that the other variables impact productivity partially because of thriving, identifying developer thriving as a key component of increasing productivity (Table 8 and Figure 1; hypotheses 4–5).

The model also indicated that healthy team metrics use and greater visibility and value of software work were both significantly associated with greater developer thriving, with visibility and value of work having the stronger effect on thriving. Visibility and value of work also mediated the relations

between HMU and both developer thriving and perceived productivity, highlighting visibility and value of work as a key lever for increasing both thriving and productivity (Table 8 and Figure 1; hypothesis 6).

Finally, HMU was associated with greater visibility and value of work, highlighting HMU as one factor for creating visibility and value of software work (Table 8 and Figure 1).

## Limitations

The results of our study should be considered in the context of several limitations. First, our study utilized a survey design. As such, there may be a response bias in who responded to our survey, resulting in limits to generalizability. There may also be the potential for a social desirability bias in reporting, although this was mitigated through the use of anonymized data collection methods.

Second, although our study represents a step toward empirically measuring developer thriving through our use of previously validated items, future research could build upon this work by conducting a full factor analysis to assess the performance of each factor of developer thriving in software contexts with a larger global population.

As is typical of most survey studies, our study utilized snowball sampling. Although this is an effective and accepted sampling method, it carries a risk of sampling bias that could limit the generalizability (external validity) of our findings.

Additionally, we operationalized our constructs using previously validated trait-based measures, which aim to capture average or overall characteristics and experiences across contexts. Although trait-based measures are more longitudinally stable than state-based measures,<sup>10</sup> they can overlook context-specific experiences. As such,

while our findings can be generalized to developers' experiences *overall*, they are not universally applicable to every situation or context.

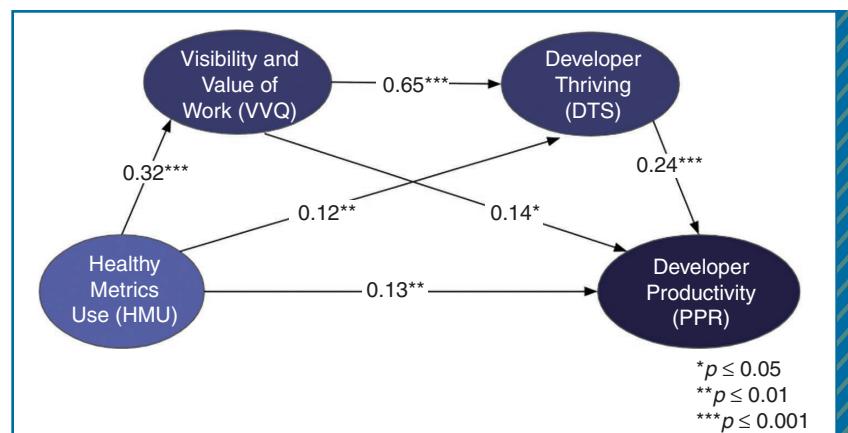
Finally, given that our data do not employ temporal precedence among

measures, our analyses cannot yield causal conclusions but must instead be considered as analyses of associations temporally placed based on the existing literature. Under this stipulation, our analyses can indicate that

**Table 8. The serial mediation model.**

Antecedent	Consequent					
	VVQ (m1)		DTS (m2)		PPR (y)	
	$\beta$	p	$\beta$	p	$\beta$	p
Direct effects						
% time code (cov1)	0.12	<.01	0.13	<.001	0.05	0.21
Years code (cov2)	0.17	<.001	0.05	0.15	0.16	<.001
HMU (x)	0.32	<.001	0.12	<.01	0.13	<.01
VVQ (m1)	—	—	0.65	<.001	0.14	<.05
DTS (m2)	—	—	—	—	0.24	<.001
Indirect effects						
HMU (x) via VVQ (m1)	—	—	0.21	<.001	0.04	<.05
HMU (x) via DTS (m2)	—	—	—	—	0.03	.001
VVQ (m1) via DTS (m2)	—	—	—	—	0.16	<.001
HMU (x) via VVQ (m1) and DTS (m2)	—	—	—	—	0.05	0.001

% time code = percent of time spent coding; years code = years of coding experience; cov1 = covariate 1; cov2 = covariate 2; x = predictor variable; m1 = mediator 1; m2 = mediator 2; y = outcome variable.



**FIGURE 1.** The developer thriving serial mediation model results. Standardized regression coefficients are represented.

there are significant relations among our variables but cannot statistically establish directionality among them. Although the proposed directionality of our model fits with current theoretical models of productivity, temporal precedence is necessary to make inferences about directionality.

cycles work to reinforce developers' sense of progress and problem-solving *even and especially* when developers encounter difficulty, friction, and failure. Across intervention science in human behavior, positive metacognitive beliefs, perceptions, and environmental factors have been found to drive

developer productivity in a human-centered way by improving developer thriving, for example, by considering factors such as enough learning time, strong supportive cultures, the opportunity to give feedback, and recognition for effort work and difficult problem-solving. Additionally, organizations can further unlock the benefits of thriving by making developers' work both visible and valued, for example, by using shared, accurate measures of software work, paying special attention to teams or types of engineering work that do not get shared broadly, and investing in systems that explicitly recognize and reward teams for the technical progress they make, particularly work that was unexpectedly challenging, required new skills, or fixed long-standing problems. 

**Developers need thriving and all its elements inside their immediate problem-solving environment, but they also need to believe that their *individual* productivity will go beyond their teams.**

Our findings are consistent with previous software research that highlights satisfaction as the strongest predictor of developer productivity.<sup>5</sup> However, our study provides evidence for a more rigorous, psychometrically tested measure that moves beyond developer satisfaction: the developer thriving framework, a multidimensional and longitudinally stable measure of the factors driving satisfaction in the first place. Additionally, our framework further expands on the connection between satisfaction and productivity to highlight visibility as the key to not only directly increasing developer thriving but also boosting the effect of thriving on developer productivity.

It is likely that the sociocognitive elements that create developer thriving are impactful because they create "virtuous cycles": positive beliefs, perceptions, and expectations about code work and problem solving. These

longitudinal behavior change, leading to long-term achievement.<sup>15</sup> Organizations can either enhance or subvert these important cycles: when teams and organizations put effort into creating a positive problem-solving culture, it sustains long-term achievement, iterative improvement, and reflective, collaborative problem-solving.

That is, developers need thriving and all its elements inside their immediate problem-solving environment, but they also need to believe that their *individual* productivity will go beyond their teams. Our visibility and value and HMU constructs are a step toward naming and measuring the missing pieces that helps explain an important connection between individual developer productivity and thriving and how the organization's measurement, valuing, and recognition of developers flows back down to software teams.

Taken together, the findings suggest that organizations can improve

### Acknowledgment

The authors thank each of our participants for their participation and support. Catherine M. Hicks and Carol S. Lee contributed equally to this work. This work involved human subjects or animals in its research. The authors confirm that all human/animal subject research procedures and protocols are exempt from review board approval. This article has supplementary downloadable material available at <https://doi.org/10.1109/MS.2024.3382957>.

### References

1. J. Verner, J. Sampson, and N. Cerpa, "What factors lead to software project failure?" in *Proc. 2nd Int. Conf. Res. Challenges Inf. Sci.*, Marrakech, Morocco, 2008, pp. 71–80, doi: 10.1109/RCIS.2008.4632095.
2. C. Jaspan and C. Sadowski, "No single metric captures productivity," in *Rethinking Productivity in Software Engineering*, C. Sadowski and

- T. Zimmermann, Eds., New York, NY, USA: Apress Open, 2019, pp. 13–20.
3. M. Cataldo and J. D. Herbsleb, “Coordination breakdowns and their impact on development productivity and software failures,” *IEEE Trans. Softw. Eng.*, vol. 39, no. 3, pp. 343–360, Mar. 2013, doi: 10.1109/TSE.2012.32.
  4. E. Bouwers, A. van Deursen, and J. Visser, “Software metrics: Pitfalls and best practices,” in *Proc. 35th Int. Conf. Softw. Eng. (ICSE)*, 2013, pp. 1491–1492, doi: 10.1109/ICSE.2013.6606755.
  5. M. A. Storey, T. Zimmermann, C. Bird, J. Czerwonka, B. Murphy, and E. Kalliamvakou, “Towards a theory of software developer job satisfaction and perceived productivity,” *IEEE Trans. Softw. Eng.*, vol. 47, no. 10, pp. 2125–2142, Oct. 2021, doi: 10.1109/TSE.2019.2944354.
  6. C. Sadowski and T. Zimmermann, *Rethinking Productivity in Software Engineering*. New York, NY, USA: Apress Open, 2019.
  7. D. Graziotin, P. Lenberg, R. Feldt, and S. Wagner, “Psychometrics in behavioral software engineering: A methodological introduction with guidelines,” *ACM Trans. Softw. Eng. Methodol.*, vol. 31, no. 1, pp. 7:1–7:36, Sep. 2021, doi: 10.1145/3469888.
  8. D. J. Brown, R. Arnold, D. Fletcher, and M. Standage, “Human thriving,” *Eur. Psychol.*, vol. 22, no. 3, pp. 167–179, Sep. 2017, doi: 10.1027/1016-9040/a000294.
  9. C. França, H. Sharp, and F. Q. B. da Silva, “Motivated software engineers are engaged and focused, while satisfied ones are happy,” in *Proc. 8th ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas. (ESEM)*, 2014, pp. 1–8, doi: 10.1145/2652542.2652545.
  10. C. Geiser, T. Gotz, F. Preckel, and P. A. Freund, “States and traits: Theories, models, and assessment,” *Eur. J. Psychol. Assessment*, vol. 33, no. 4, pp. 219–223, Jul. 2017, doi: 10.1027/1015-5759/a000413.

## ABOUT THE AUTHORS



**CATHERINE M. HICKS** is the vice president of research insights and the director of the Developer Success Lab, Pluralsight, Draper, UT 84020 USA, and a research affiliate at the University of California, San Diego, La Jolla, CA 92093 USA. Her research interests include the psychology of software teams, learning science, and motivation. Hicks received her Ph.D. in quantitative experimental psychology from the University of California, San Diego. Contact her at cat-hicks@pluralsight.com.



**CAROL S. LEE** is a principal research scientist at the Developer Success Lab, Pluralsight, Draper, UT 84020 USA. Her research interests include understanding, assessing, and creating interventions for the mental health and psychological experiences of developers, such as thriving, stress, and anxiety. Lee received her Ph.D. in clinical psychology at the University of Massachusetts Boston. She is an IBHRI research fellow and a member of APA and ABCT. Contact her at carol-lee@pluralsight.com.



**MORGAN C. RAMSEY** is a principal user experience researcher at Pluralsight, Draper, UT 84020 USA. Her research interests include inclusive design, ethnography, and UX to elevate underrepresented users' narratives. Ramsey received her B.A. in public policy at Duke University. Contact her at morgan-ramsey@pluralsight.com.



11. “Workplace mental health & well-being,” Office of the U.S. Surgeon General. Accessed: Jan. 1, 2023. [Online]. Available: <https://www.hhs.gov/surgeongeneral/priorities/workplace-well-being/index.html>
12. L. Dawson, B. Mullan, and K. Sainsbury, “Using the theory of planned behaviour to measure motivation for recovery in anorexia nervosa,” *Appetite*, vol. 84, pp. 309–315, Jan. 2015, doi: 10.1016/j.appet.2014.10.028.
13. J. S. Cohen, J. M. Edmunds, D. M. Brodman, C. L. Benjamin, and P. C. Kendall, “Using self-monitoring: Implementation of collaborative empiricism in cognitive-behavioral therapy,” *Cogn. Behav. Pract.*, vol. 20, no. 4, pp. 419–428, Nov. 2013, doi: 10.1016/j.cbpra.2012.06.002.
14. A. N. Meyer, G. C. Murphy, T. Zimmermann, and T. Fritz, “Enabling good work habits in software developers through reflective goal-setting,” *IEEE Trans. Softw. Eng.*, vol. 47, no. 9, pp. 1872–1885, Sep. 2019, doi: 10.1109/TSE.2019.2938525.
15. D. Yeager, G. Walton, and G. L. Cohen, “Addressing achievement gaps with psychological interventions,” *Phi Delta Kappan*, vol. 94, no. 5, pp. 62–65, Feb. 2013, doi: 10.1177/003172171309400514.