Advancing Organizational Science with Computational Process Theories

Goran Kuljanin[1], Michael T. Braun[1], James A. Grand[2], Jeffrey D. Olenick[3], Georgia T. Chao[4], and Steve W. J. Kozlowski[4]

[1] Department of Management and Entrepreneurship, DePaul University

[2] Department of Psychology, University of Maryland

[3] Department of Psychology, University of Georgia

[4] Department of Psychology, University of South Florida

Cite As:

Kuljanin, G., Braun, M. T., Grand, J. A., Olenick, J. D., Chao, G. T., & Kozlowski, S. W. J. (in press). Advancing organizational science with computational process theories. *The Leadership Quarterly*.

Author Note

This work received support by a research grant (W911NF2210005; S.W.J. Kozlowski, PI; G.T. Chao, Co-PI, J. A. Grand, Co-I; M. T. Braun, Co-I; G. Kuljanin, Co-I) from the U.S. Army Research Institute for the Behavioral and Social Sciences. The views, opinions, and/or findings contained in this manuscript reflect solely those of the authors, and one should not construe them as an official Department of Defense position, policy, or decision, unless so designated by other documentation.

We thank the developers of *R* (Version 4.3.0; R Core Team, 2023), the *R*-packages *dplyr* (Version 1.1.3; Wickham et al., 2023), *DT* (Version 0.30; Xie et al., 2023), *flextable* (Version 0.9.1; Gohel & Skintzos, 2023), *forcats* (Version 1.0.0; Wickham, 2023), *fs* (Version 1.6.1; Hester et al., 2023), *furrr* (Version 0.3.1; Bengtsson, 2021; Vaughan & Dancho, 2022), *ggplot2* (Version 3.4.4; Wickham, 2016), *htmltools* (Version 0.5.7; Cheng et al., 2023), *knitr* (Version 1.45; Xie, 2015), *lavaan* (Version 0.6.16; Rosseel, 2012), *magrittr* (Version 2.0.3; Bache & Wickham, 2022), *officer* (Version 0.6.2; Gohel, 2023), *papaja* (Version 0.1.1; Aust & Barth, 2022), *purrr* (Version 1.0.2; Wickham & Henry, 2023), *rlang* (Version 1.1.1; Henry & Wickham, 2023), *rmarkdown* (Version 2.21; Xie et al., 2018, 2020), *scales* (Version 1.2.1; Wickham & Seidel, 2022), *tibble* (Version 3.2.1; Müller & Wickham, 2023), *tictoc* (Version 1.2; Izrailev, 2023), and *tinylabels* (Version 0.2.3; Barth, 2022), *RStudio* (Posit Team, 2023), and *Mermaid.js* (Sveidqvist & Jain, 2021) in the production of this manuscript and supplemental materials.

Please address correspondence concerning this manuscript to Goran Kuljanin, 1 E. Jackson Blvd., Department of Management and Entrepreneurship, DePaul University, Chicago, IL, 60604. E-mail: g.kuljanin@depaul.edu.

Abstract

Organizational scholars commonly refer to organizations as complex systems unfolding as a function of work processes. Consequently, the direct study of work processes necessitates our attention. However, organizational scholars tend not to study work processes directly. Instead, organizational scholars commonly develop theories about relationships among psychological construct phenomena that indirectly reference people's affective, behavioral, cognitive, and/or social processes as underlying explanations. Specifically, construct-oriented theories summarize processes in operation across actors, time, and contexts, and thus, provide limited insights into how focal phenomena manifest directly as a function of process operations. Construct theories remain one-step removed from articulating sequences of actions and two-steps removed from describing generative mechanisms responsible for observed actions. By "missing the action," construct theories offer incomplete explanatory accounts and imprecise interventions. We assert that researchers in organizational science can make progress towards addressing these concerns by directing greater attention to developing computational process theories. We begin by presenting a framework for differentiating theories based on their focus (constructs versus processes) and modality (narrative versus computational). We use the framework to contrast narrative construct theories to computational process theories. We then describe key design principles for developing computational process theories and explain those principles using a leadership example. We use simulated data, from the computational process model we develop, to explicitly demonstrate the differences between construct and process thinking. We then discuss how computational process theories advance theory development. We conclude with a discussion of the long-term benefits of computational process theories for organizational science.

Keywords: computational process theories, generative mechanisms, process thinking, process explanations, organizational science

Advancing Organizational Science With Computational Process Theories

The foundational development of organizational theories in the 1950s and 1960s regarded organizations as complex systems driven by process dynamics that lead to the emergence of phenomena across system levels (e.g., Katz & Kahn, 1966; Likert, 1961), a conception with roots dating back to at least the 1930s (Roethlisberger & Dickson, 1939). However, use of this conception in theory and research served merely as a metaphor. The advent of multilevel theory, at the turn of the twenty-first century, intended to push the “organizations as complex systems” concept beyond mere metaphor to directly shape the development of organizational theories and empirical research (Kozlowski & Klein, 2000). Multilevel theories focused on two primary pathways that drive organizational system behavior: (1) top-down contextual influences (e.g., team norms) that shape and constrain lower-level entities (e.g., team members); (2) bottom-up influences that consist of lower-level entities enacting processes (e.g., communication) with phenomena manifesting at the meso and higher levels of the system (e.g., shared mental models). Although multilevel theories and empirical research in organizational science now prove normative, most of that work focuses on top-down, cross-level construct relationships utilizing static, cross-sectional research designs (Kozlowski & Chao, 2012). Thus, the theoretical promise of detailing processes and their dynamics as explanations for multilevel phenomena in complex organizational systems largely remains unrealized (Cronin et al., 2011; Kozlowski, 2015).

Virtually all theories in organizational science – across the micro, meso, and macro levels – evoke processes as explanations for relationships that link psychological constructs of interest (Kozlowski, 2022). However, organizational scientists tend not to examine processes directly (Kozlowski, 2015). Rather, scholars in the organizational sciences commonly represent processes as mediating variables that serve as transmitters of influence from inputs to outputs (see Wright, 1920, and Hyman, 1955, for the origins of such thinking). The mediators evoked in empirical studies typically summarize the perceptions of thoughts, feelings, behaviors, and social relations of individuals across actors, time, and contexts; what Marks et al. (2001) describe as “emergent states.” This summation misses the details of which and how the affective, behavioral, cognitive, and social processes enacted by actors generate those chained construct relations. Consequently, processes – the timing and sequencing of actions enacted by actors, and critically, the underlying generative mechanisms responsible for such actions (Simon, 1992) – remain unexplored.

Organizational scholars recognize two primary approaches toward theory development in organizational science: construct (or variance) and process (or actor) theories (Macy & Willer, 2002; Mohr, 1982; Pentland, 1999). Construct theories, which focus on prediction, dominate in organizational science. Over the last century, the application of the construct approach by organizational scholars generated a substantial and informative scientific foundation. We may usefully translate many of these findings into policies and practices that improve selection (e.g., Hunter & Hunter, 1984), training (e.g., Arthur et al., 2003), leadership (e.g., Paustian-Underdahl et al., 2014), and team effectiveness (e.g., LePine et al., 2008), to name but a few areas of organizational science. Yet, prediction, per se, does not equate to a theoretical explanation (Coveney et al., 2016; Mohr, 1982; Pentland, 1999). Process theories focus on explaining the generative mechanisms responsible for processes as action sequences that ultimately underpin construct relationships and the predictions we derive from such relationships. We assert that advancing the theoretical foundation for organizational science necessitates an improved capacity to develop process theories that can explain behavior in complex, dynamical organizational systems. To achieve such an aim, we intend for this manuscript to facilitate a transition in developing theory from one focused on constructs to one focused on processes. This transition in developing theory primarily involves the consideration of generative mechanisms that drive sequences of actions undertaken by organizational actors. Such process thinking would improve the quality of theoretical explanations in organizational science.

Organizational scholars recognize the need for theories of organizational science (Antonakis, 2017; Fischer et al., 2017; Kozlowski et al., 2013, 2016) to detail the operations of processes in action. Indeed, if people make the place (Schneider, 1987), then we need theories to understand people, not merely as bundles of constructs, but also, as actors. In other words, we need theories that directly speak to how people make the place. We need to advance theory development from explaining construct covariances and mediational pathways to also explicating the generative mechanisms that produce action sequences and emergent phenomena of interest (Braun et al., 2022; Kozlowski, 2022; Kozlowski et al., 2013; Smaldino et al., 2015). Thus, we need to advance theory development from a primary focus on *construct* thinking to one that also embraces *process* thinking, or, if we wish to make a pleasant rhyme of it, we need to expand the focus of theory development from factors to actors (Macy & Willer, 2002; Strauss & Grand, 2022).

To understand processes and better explain known causal construct relationships, we need to develop theories that describe how processes function (Mohr, 1982; Pentland, 1999). Process thinking involves specifying the generative mechanisms underlying processes by considering the characteristics and states of actors and their environments, the possible processes and actions that actors may enact, how environments evolve and shape actors, how actors respond to and update their environments, and how actors act and interact with each other (Epstein, 1999; Sayama, 2015). Until recently, organizational researchers tended to confine process thinking to narrative theorizing and qualitative research (Pentland, 1999). Yet, we cannot easily articulate, nor play out in the narratives formulated by our minds, the inherent complexities stemming from the unfolding dynamics of processes in operation (Cronin et al., 2009). Consequently, we need an alternative approach for developing process theories to help track the myriad implications of these complexities. Computational modeling affords the capability to build and establish the plausibility of process theories (Grand et al., 2016; Kozlowski et al., 2013; Smaldino et al., 2015).

Although still not commonly applied as a tool for developing theory in organizational science (Kozlowski, 2015), organizational scholars consistently tout the value of computational modeling for advancing a better understanding of complex systems (see Appendix A for a concise overview of computational modeling work in organizational science). We may write a computational model to express our thoughts formally in the form of mathematical and statistical equations, logical statements, control loops, and/or functions (Ilgen & Hulin, 2000). Once formally stated, we can simulate these models, with a sufficiently capable programming software of choice (e.g., *R* – R Core Team, 2023; *Python* – Van Rossum & Drake, 2009; *Julia* – Bezanson et al., 2012; *NetLogo* – Wilensky, 1999), to explore the implications of process theories (as an alternative to programming software, see Sayama, 2015, for the use of pure mathematics or Schelling, 1971, for the use of physical/tangible items [e.g., coins] to enact rules as methods for computational simulation). Thus, computational models help scientists represent their thinking with greater transparency, specificity, and reproducibility, which helps them advance the development of their theories (Eronen & Bringmann, 2021; Guest & Martin, 2021; Simon, 1957). Organizational theorists may use computational models to more explicitly propose how inputs (e.g., knowledge, skills, abilities, personalities) convert to outputs (e.g., productivity, commitment, engagement) via affective (e.g., regulating emotions), behavioral (e.g., pursuing goals), cognitive (e.g., decision-making and learning), and social (e.g., collaborating) processes that cut across multiple organizational levels to foresee what patterns emerge from such interactions (Ballard et al., 2021; Grand et al., 2016). To supplement our imaginations for process theorizing, we need computational models to formalize how processes of human systems explicitly operate.

We aim to facilitate a transition in thought for organizational scientists from the dominant paradigm of narrative construct thinking to computational process thinking. Such a transition in thought proves particularly important for advancing theory that can explain complex, dynamic organizational system phenomena. First, we situate organizational scientists with respect to the differences between developing computational process theories focused on explaining generative mechanisms and narrative construct theories focused on proposing construct relationships. Second, we provide direct guidance for developing computational process theories of organizational phenomena using a leadership example. With simulated data from a developed computational process model of the leadership example, we explicitly demonstrate the differences between construct and process thinking. Third, we detail how computational process theories advance theory development even before any new empirical data collection. In doing so, we highlight "default" scientific advantages that "come along for the ride" when we develop computational process theories. Lastly, we discuss how the proliferation of computational process theories may benefit organizational science moving forward. We hope organizational scientists find that our exposition facilitates a transition in theory development from narrative construct theories to computational process theories of organizational phenomena.

A Typology for Theory Development

As a means of situating our discourse, we propose a typology to characterize theory development along two dimensions. The *focus of a theory* reflects the extent to which a theory develops an account of construct levels, variances, covariances, and relationships versus a focus on explaining the generative mechanisms underlying processes responsible for emergent phenomena of interest (Anghel et al., 2004; Braun et al., 2022; Klein et al., 2006; Kozlowski, 2022; Mohr, 1982; Pentland, 1999; Strauss & Grand, 2022). Construct theories primarily direct attention to advancing predictions about patterns of covariance among constructs, whereas process theories focus on explaining how actors think, feel, behave, and influence each other to generate emergent organizational phenomena. The *modality of a theory* reflects the extent to which a theory represents core concepts and mechanisms primarily using narrative discourse versus computational formulation (Csaszar, 2020; Park et al., 2015; Vancouver et al., 2020). Narrative theories rely primarily on natural language to define, characterize, and communicate its central tenets. In contrast, computational theories employ an explanatory narrative and supplement it with mathematics, logic, and/or computer code to specify their assertions explicitly. As illustrated in Table 1, when we cross these two dimensions, they yield four types of theories: (1) narrative construct theories, (2) narrative process theories, (3) computational construct theories, and (4) computational process theories. Using the theory typology, we characterize the overwhelming majority of organizational theories as narrative construct theories. Although narrative construct theories serve as a foundational approach in organizational science, we argue that the science would significantly benefit from developing more computational

Table 1

A Typology for Theory Development

	Theory Modality	
Theory Focus	Narrative	Computational
Constructs	**Definition**: Uses narrative rationale to predict how constructs covary **Example**: DeRue et al. (2010) **Note**: Most common form of theorizing in organizational science	**Definition**: Uses mathematics and logic to highlight consequences of construct relationships, especially consequences over time **Example**: Park et al. (2015) **Note**: Commonly comes in the form of what-if scenario analyses
Processes	**Definition**: Uses a narrative rationale to describe how actors think, feel, behave, and socialize with each other in pursuit of pertinent goals **Example**: Klein et al. (2006) **Note**: Qualitative research investigating work processes	**Definition**: Uses mathematics, logic, and computer programs to describe how actors think, feel, behave, and socialize with each other in pursuit of pertinent goals **Example**: Anghel et al. (2004) **Note**: Explicit, formal specification of operational processes; we argue the benefits of developing these types of theories for organizational science

process theories.

Theory Focus: Construct Versus Process Theories

We distinguish construct theories from process theories (Braun et al., 2022; Kozlowski, 2022; Mohr, 1982; Niederman & March, 2018). We suspect the distinction between construct and process theories may prove challenging given the omnipresent references to processes in construct mediation models throughout the organizational sciences. To facilitate an understanding of this critical distinction, Table 2 presents the fundamental ways in which construct and process theories differ across four key criteria: (1) focus of explanation, (2) basis for explanation, (3) empirical measurements, and (4) empirical analytics.

Focus of Explanation

Construct theories focus on describing how a set of two or more constructs relate to one

Table 2

Comparing Construct Theories to Process Theories

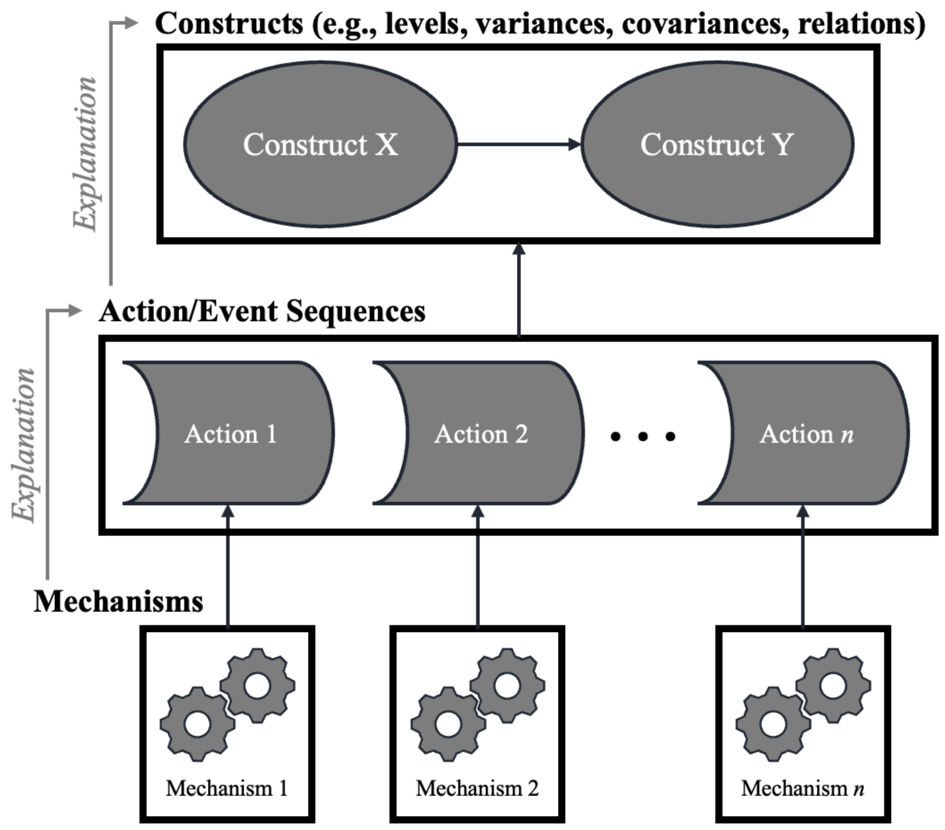
Comparison	Construct Theories	Process Theories
Focus of Explanation	How changes in the levels of one (input) construct relate to changes in the levels of another (output) construct	How actors enact actions and environments update via generative mechanisms
Basis for Explanation	Mediator constructs that transmit impact from input to output constructs	Generative mechanisms that serve as rules for the process actions of actors and evolution of environmental events
Empirical Measurements	Observations across different actors (i.e., cross-sectional data) and/or within actors across time periods (i.e., longitudinal data)	Observations of actor actions and/or interactions and environmental events within (e.g., actors taking actions during performance episodes) and in-between focal time periods (e.g., actors learning in-between performance episodes)
Empirical Analytics	Statistical models (e.g., structural equations modeling) that assess construct relationships cross-sectionally or longitudinally	Computational or statistical models that assess generative mechanisms and the flow of actor actions and environmental events within and in-between focal periods of time

another in a chain of (often implied) causal relationships. In doing so, construct theories primarily direct attention towards describing how the levels of one (input) construct (causally, with appropriate experimental or statistical control) associate with the levels of another (output) construct, in the population. Such accounts typically result in the development of propositions regarding the relationships predicted to exist among constructs (e.g., positive/negative linear construct relationships) (Antonakis et al., 2010; Antonakis, 2017). Besides predicting construct relationships, construct theories – at least implicitly – propose constructs as causal drivers of actions/events (e.g., individuals with higher levels of autocratic leadership enact more directive behaviors). Yet, constructs, including those constructs presented as mediational mechanisms responsible for transmitting the effects of inputs to outputs (Ilgen et al., 2005), inherently summarize specific process sequences, patterns, or instances of actions enacted by actors (Braun et al., 2022). This construal implicitly treats all specifics regarding actions and actors that result in the same level of a construct as equifinal and interchangeable. Thus, construct theories frequently do not seek to explain how variance in a construct translates into specific actions/events that produce phenomena of interest. Instead, the descriptive focus of construct theories primarily concerns itself with elaborating and justifying nomological networks of constructs that theorists subsequently use to generate predictions about how differences in the levels of constructs should relate to differences in the levels of other constructs.

To help us discuss the focus of explanation for process theories, we refer to Figure 1. Process theories focus on explaining how observed phenomena, such as construct relations (top of Figure 1), stem from process actions/events (middle of Figure 1) that actors/environments output as a function of active generative mechanisms (bottom of Figure 1) operating during

Figure 1

Layers of Explanation Offered by Process Theories



and/or in-between focal time periods and/or contexts. In other words, process theories primarily direct attention towards explicating the specific actions/events involved, and the order in which those actions/events manifest, as a function of generative mechanisms (e.g., Anderson, 1990; Borsboom et al., 2021; Mohr, 1982; Pentland, 1999; Simon, 1992). Such accounts typically result in the development of assumptions and/or propositions regarding a series of actions/events (i.e., a process) that, as they transpire, cause construct states to emerge. Of note, we might characterize emergent construct states as distributional properties (i.e., different levels of team performance emerge depending on how a proposed set of actions/events unfold) or construct-to-construct relationships (i.e., how leaders enact a proposed series of actions/events engenders a correlation between the levels of leadership direction and team performance). Importantly, we do not imply that process explanations require multilevel theory. We may define leader direction as a behavioral construct that involves dictating instructions to subordinates to various degrees. As opposed to focusing on levels of leader direction, a process explanation of leader direction may consider exactly how, when, and to whom leaders dictate instructions at the same behavioral level. In other words, we define the construct and provide the process explanation, which details actions sequences and generative mechanisms, at a single, behavioral level of theory as opposed to traversing multiple levels of theory (Anderson, 1990). Of course, process explanations may refer to other levels of theory, such as when considering team and leadership processes (e.g., discussing the processes that actors enact to explain dyadic relationships).

To summarize, the explanatory focus of process theories primarily concerns itself with elaborating action/event sequences and their underlying generative mechanisms that explain how phenomena emerge at the same or different levels of theory. In other words, process theories explain predicted or observed group mean differences and construct relationships by detailing the action sequences that actors enact and the events that unfold in actors' environments. Process theories explain those action sequences and environmental events by further detailing the generative mechanisms that produce them. In this way, process theories offer layers of explanation to account for emergent phenomena of interest such as construct relationships.

Theories that detail action/event sequences and generative mechanisms responsible for them advance our understanding of organizational phenomena.

Basis for Explanation

In many respects, differences in how construct versus process theories construe their basis for explanation demarcate important distinctions in the underlying foundations and specifics of causal explanation embraced by both types of theories. Construct theories most commonly reference other constructs – typically in the form of mediators – as the "processes" that account for observed relationships. Such an idea implies that the level of an input construct (causally, with appropriate experimental or statistical control) influences the level of a mediating construct, which, in turn, (causally, with appropriate experimental or statistical control) influences the level of an output construct. For instance, the levels of leadership direction may influence the levels of any team member's motivation, which, in turn, may influence the levels of team performance. In this case, team member motivation serves as the mediating mechanism that transmits the impact of leadership direction onto team performance. Consequently, construct theories often direct substantial attention toward distinguishing constructs that serve in the roles of antecedents, mediators, and consequents, while justifying the presence/relevance of specific mediators because such constructs serve as purported causal mechanisms (i.e., inputs impact outputs via mediators).[1]

In contrast, process theories rely on generative mechanisms as a basis for explanation. Generative mechanisms serve as a system of functions, conditional rules, and operational

[1] From a philosophy of science perspective, this approach cleanly aligns with the logic of justifying causality through construct-focused counterfactual reasoning. Assuming no omitted causes, or at least the appropriate experimental or statistical control/elimination of all other causes (see Antonakis et al., 2010), the typical formulation of construct-focused counterfactual reasoning comes in the form of: "Construct X causes construct Y because in the absence of X, Y does not exist." However, we note that construct-focused counterfactual reasoning only strictly supports causal conclusions/predictions relevant to relationships between levels of constructs. The proposition that X should covary with Y does not provide support for a process explanation for why and how specific states/levels of X result in specific states/levels of Y (for further discussion, see Borsboom et al., 2003, and Borsboom et al., 2004). To study processes, we need process-focused counterfactual reasoning.

principles responsible for generating observable states, actions, or events (Simon, 1992). Stated differently, process theories represent generative mechanisms as statements that convey which, when, where, and how actors act, interact, and change, and how and when environments evolve. For instance, a process theory may propose that a critical action/event carried out by a leader during the process of facilitating team performance consists of providing feedback to members. A process theory would proceed to develop one or more generative mechanisms (i.e., functions or rules) that specify how this action/event occurs (e.g., leaders provide feedback to any member whose work activities fail to follow desired work procedures). In this way, generative rules serve as the proposed causal drivers for what we observe (i.e., actions/events) and the subsequent implications of those actions/events (e.g., construct relationships) (Braun et al., 2022; Pentland, 1999; Sayama, 2015).

Empirical Measurements and Analytics

Differences in the focus and basis of explanation emphasized by construct and process theories naturally lend themselves to differences in the efforts to evaluate and examine the implications of both types of theories. With respect to empirical measurements, the examination of construct theories primarily focuses on measuring constructs between actors (i.e., cross-sectional data) and/or within actors capturing snapshots across multiple time periods (i.e., longitudinal data). Organizational researchers use within- and between-actor construct data to evaluate theoretical propositions with statistical models (e.g., structural equations modeling) that relate variability in measured constructs to each other (e.g., estimating a group mean difference or a regression coefficient between two constructs; calculating an effect size or shared variance) with causal claims requiring sufficient methodological controls and appropriate data collection designs (Antonakis et al., 2010).

In contrast, data aligned with process theories focus on measuring actors' actions and/or interactions within and/or in-between focal time periods and/or contexts. Critically, we note that longitudinal construct data does not equate to process data, as the former does not capture processes in action. For instance, the measurement of team performance across several performance episodes may establish a negative, positive, or stable trend in team performance. Yet, those construct measurements of team performance do not speak to the executory actions of team members during performance episodes nor the preparatory actions of team members in-between focal performance episodes (Marks et al., 2001). Hence, only process measurement may speak to processes in action. Organizational researchers may use process data to evaluate propositions from a process theory in several ways. For example, they can use process data to evaluate the effect of proposed generative mechanisms on emergent construct states of interest via the use of familiar analytics (e.g., descriptive statistics, data visualizations, statistical models), analyze the flow or sequencing of proposed activities via process analytics (e.g., process mining, relational event modeling, network flows), or examine the generative mechanisms of actors and environments via computational models representative of process theories. Organizational researchers may also use process data to evaluate process causality, demonstrating that specific generative mechanisms prove responsible for observed action sequences (Braun et al., 2022).

Theory Modality: Narrative Versus Computational Theories

We distinguish narrative theories from computational theories (Adner et al., 2009; Csaszar, 2020; Vancouver et al., 2020). Narrative theories provide argumentation for hypotheses and explanations of phenomena in written natural language. Certainly, when compared to merely discussing ideas orally, written ideas provide an advancement with respect to theory development. Any reader may examine the exact same narrative record to understand it and/or develop the argumentation further. Yet, readers often come to different conclusions about the meaning of a particular natural language record even when considering the more standardized writing in academic journals. Thus, they might read the same words but come away with different interpretations. If they hold different interpretations, then they cannot reproduce theoretical implications in a consistent manner. The differences in interpretations, in turn, create problems for theory development, not to mention empirical replication.

Computational theories augment a narrative theoretical explanation with specifications for hypotheses and explanations of phenomena using formal languages such as mathematics, logic, and computer programs (Davis et al., 2007). In other words, computational theories express our thoughts in the form of mathematical and/or statistical equations, control loops, logical statements, and/or programmed functions. The specification of computations resolves the interpretive ambiguity inherent in narrative theories because computations more clearly communicate the intentions of theorists. Due to the explicit expression of our thoughts, we can directly simulate computational theories to evaluate how multiple processes unfold, evolve, and interact over time and/or contexts to tangibly show their implications. Indeed, without formal computational aids, it proves quite difficult for us to reason accurately about what we expect to happen as multiple processes, possibly spanning several system levels, unfold and interact over time (Cronin et al., 2009; Farrell & Lewandowsky, 2010; Vancouver et al., 2020). Furthermore, computational theories offer an internal validation check, referred to as generative sufficiency, on the plausibility of the theoretical logic (Epstein, 1999). Simulations of computational theories can ascertain whether programmed generative mechanisms produce action sequences consistent with theoretical logic. Subsequently, we can summarize intermediate or ending states of processes in operation to evaluate construct implications of process theories. Thus, computational theories facilitate theory development, not to mention empirical replication.

Narrative Construct Theories Versus Computational Process Theories

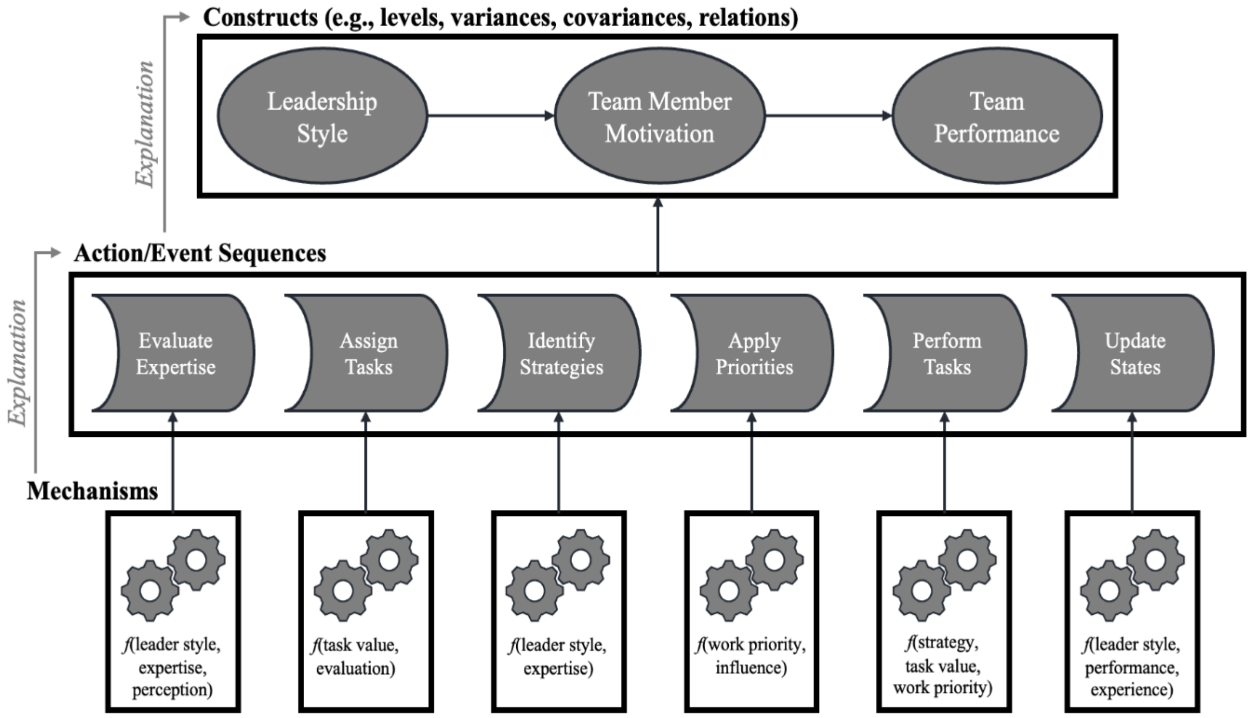
We argue that organizational science would greatly benefit from the development of computational process theories (Braun et al., 2022; Hazy, 2007; Ilgen & Hulin, 2000; Kozlowski et al., 2013; Prietula et al., 1998; Sayama, 2015; Smaldino et al., 2015). To highlight those benefits, we compare computational process theories with narrative construct theories. Narrative construct theories represent the overwhelmingly dominant approach to organizational science and serve as our referent point for the promotion of computational process theories. As an overview of our comparison, we distinguish how the two types of theories differ along the two dimensions of our theory typology. Computational theories, as opposed to narrative theories, provide the advantage of explicitly articulating ideas and a means for simulating the dynamic implications of those ideas directly. Process theories, as opposed to construct theories, provide the advantage of describing how actors, operating on their own and in concert, execute actions to satisfy their needs and accomplish their goals.

To further explicate the differences between the two types of theories, emphasize benefits afforded by computational process theorizing, and demonstrate how to develop a computational process theory, we develop an example focused on leadership styles, team member motivation, and team performance (e.g., DeRue et al., 2010; Lewin et al., 1939; Lewin & Lippitt, 1938). Although developing computational process theories offers benefits for any area of organizational research, leadership proves an excellent exemplar for formulating computational process theories given that it constitutes a process consisting of actions and interactions among multiple organizational actors shaped and constrained by organizational contexts (Carter et al., 2015). As inspiration for our example, we consider seminal studies (reported by Lewin et al., 1939, and Lewin and Lippitt, 1938), which experimentally examined the effect of different leadership styles – autocratic versus democratic (which we refer to as delegative) – on group behavior. With the help of Figure 2, we differentiate the insights and understanding afforded by a narrative construct theory versus a computational process theory in accounting for how leadership styles influence team member motivation, which, in turn, affects team performance, paying special attention to the novel benefits obtained through computational process theorizing.

A narrative construct theory would focus on explaining differences in team performance by referring to construct relationships. Consistent with our inspirational exemplar (Lewin et al., 1939; Lewin & Lippitt, 1938), a narrative construct theory might state that a particular leadership style, consisting of leaders who direct team members as opposed to leaders who allow team members to direct themselves, offers a greater capacity to motivate team members. This singular leadership capacity to motivate team members results in energizing team members who, in turn, use their motivation to perform their work effectively. Alternatively, a narrative construct theory might argue that delegative leadership excites team member motivation to a greater degree, and,

Figure 2

An Example of Developing a Computational Process Theory


Constructs (e.g., levels, variances, covariances, relations)
Explanation
Leadership Style
Team Member Motivation
Team Performance
Action/Event Sequences
Explanation
Evaluate Expertise
Assign Tasks
Identify Strategies
Apply Priorities
Perform Tasks
Update States
Mechanisms
f(leader style, expertise, perception)
f(task value, evaluation)
f(leader style, expertise)
f(work priority, influence)
f(strategy, task value, work priority)
f(leader style, performance, experience)

in turn, translates to more effective team performance. In any case, the narrative construct theory uses a construct mediation perspective to formulate its central thesis: leadership style → team member motivation → team performance. We depict these construct relations at the top of Figure 2. Importantly, narrative construct theories only implicitly reference actors by referring to actors' constructs. They do not directly advance claims or explanations regarding the actions that actors perform nor the mechanisms that generate (cause) those actions. In summary, narrative construct theories "miss the action" by summarizing it into a construct representation.

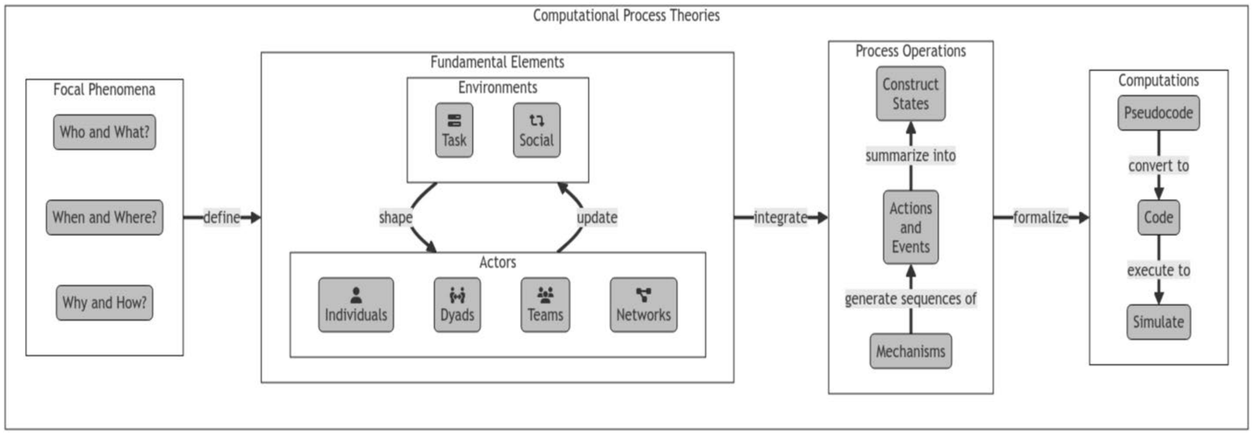
In contrast, a computational process theory would focus on explaining differences in team performance by formally specifying how leaders and team members act and interact with each other over time. To provide the explanation, a computational process theory might specify that team performance in this context emerges from a particular series of actions: first, actors (i.e., leaders or team members) evaluate each team member's expertise to perform tasks; second, actors allocate team members to tasks based on their evaluations and the importance of tasks; third, actors identify task strategies for team members to execute; fourth, team members apply their work priorities to decide if they will perform their assigned tasks; fifth, team members perform their assigned tasks; sixth, leaders and team members update their states based on their experiences; and the cycle repeats. We depict generative mechanisms and resulting actions at the bottom and middle of Figure 2, respectively. Figure 2 indicates that a computational process theory specifies generative mechanisms as an explanatory account of how actors execute each action step. For example, a computational process theory would specify exactly how leaders evaluate team members for task assignment via computational, logical, and/or quantitative expressions (e.g., a leader's evaluation of a team member happens as a function of actual expertise of the team member, represented using an initial score drawn from a normal distribution, and the leader's perceptual accuracy, represented using an initial validity score of zero; both the team member's expertise and the leader's perceptual accuracy dynamically update as task experience accrues). We may then use these process actions as an explanation for construct levels, variances, and covariances. In this way, a computational process theory explicitly describes how the construct relationships between leadership style, team member motivation, and team performance emerge via individual and dyadic generative mechanisms and the consequent actions and interactions that result. In this sense, computational process theories "live in the action" by explaining how actors who follow generative mechanisms operate.

A Guide for Developing Computational Process Theories of Organizational Phenomena

We now turn our attention to describing a guide for developing computational process theories. In Figure 3, we provide a general guide that emphasizes the most important considerations for developing computational process theories of organizational phenomena. We present these considerations as sufficiently general to encompass any choice of eventual computational framework such as social networks, systems of dynamical equations, or agent-based models (see also Kozlowski et al., 2016). Our guide walks readers through (1) defining focal phenomena of interest, (2) specifying actors and environments of focal phenomena, (3) elaborating process operations for actors and environments, and (4) formalizing theory with

Figure 3

A Guide for Developing Computational Process Theories



computations.

To demonstrate the application of our guide, we reference the running example of leadership styles, team member motivation, and team performance. The computational process theory we develop seeks to ascertain how two types of leadership style operate differently under a general team and task context, and the consequences those operational differences hold for team performance (Lewin et al., 1939; Lewin & Lippitt, 1938). Throughout this section, we emphasize the transition in thought from narrative construct thinking to computational process thinking. In other words, we highlight how we translate ideas we hold about construct relationships into ideas about process operations.

Before discussing these considerations, we wish to highlight two overarching considerations that influence the development of computational process theories residing beyond the guiding framework shown in Figure 3. The first consideration consists of balancing the tension between parsimony – keeping the critical components and core operationalizations that comprise a computational process theory as sparse as possible – and completeness – sufficiently incorporating known aspects relevant to focal phenomena. In brief, we wish to incorporate generative mechanisms necessary to produce focal phenomena of interest while remaining as parsimonious as possible. We aimed to achieve this balance with the running example.

A second, related consideration consists of establishing "where" a model intended to represent a computational process theory of interest resides along the continuum of intellective (de-contextualized, theoretical) to emulative (contextualized, applied) computational models (Carley, 2002). Intellective computational models focus on how core processes, theoretically, may result in emergent phenomena of interest in a broadly representative, yet de-contextualized environment (e.g., Samuelson et al.'s, 2019, model on the emergence of organizational gender stratification under various employee staffing and development circumstances; Scullen et al.'s, 2005, model on the consequences of forced distribution performance management systems for organizational performance). When building an intellective computational model, we concern ourselves less with specific parameter values of inputs, and instead, focus on the appropriateness of modeled processes that generate, and thereby, explain sequences and patterns of actions and emergent outcomes. Therefore, we generally investigate either a set of processes under a range of parameter values to see what patterns in actions and outcomes emerge, or we evaluate the differences that emerge in the patterns of actions and outcomes from various modeled processes.

Emulative computational models concern themselves with representing generative mechanisms, processes, and actors operating in a contextualized environment (e.g., Ballard et al.'s, 2021, model of self-regulation for an air traffic control task; Levitt's, 2012, model of project management for complex projects such as satellite launches). For emulative computational models, we generally aim to simultaneously model the specific mechanics of relevant processes of a particular environment and fit our model to representative data reflective of the contextualized environment. In such a way, an emulative computational model, compared to an intellective computational model, allows for more robust and accurate inferences, predictions, and recommendations with respect to the specific context under investigation. Yet, once we build a computational model, we may "slide" a model along the intellective-emulative continuum. Most naturally, we might first represent broad processes to evaluate theoretical possibilities, and then, we might contextualize a model for a specific environment (Carley, 2002).

With respect to the running example we discuss, we develop a computational model at the intellective end of the continuum. With the example model, we aim to capture the relevant generative mechanisms responsible for actions undertaken by leaders and team members as they engage in their work. Given such an aim, our modeling choices seek to ensure a logical consistency for the example computational model, which, in turn, provides us the opportunity to evaluate how modeled mechanisms generate patterns in actions (e.g., task assignment) and emergent construct relationships (e.g., team member motivation to team performance).

In the supplemental materials associated with this manuscript, we translate our computational process theory into an executable computational process model in the *R* software environment (R Core Team, 2023). The supplemental materials include an *R Markdown* script, best executed via *RStudio* (Posit Team, 2023), and the *R Markdown* script converted to a *HyperText Markup Language* (*HTML*) file, which readers may open in any web browser without needing *R* and *RStudio*. For readers who wish to execute the computational process model code, we recommend they first open the *HTML* file for instructions on how to execute the code in the *R Markdown* script. Both files provide the complete details of our computational process model.

Focal Phenomena

Developing computational process theories begins by defining the focal phenomena of interest and setting boundaries around them (i.e., what we include and exclude). For clarity, we define phenomena as naturally emergent patterns that scientists seek to explain (e.g., construct levels, variances, and covariances, multivariate relations, network connections, states, outcomes, events, experiences) (Borsboom et al., 2021). To define focal phenomena and set boundaries, we ask six interrogative questions grouped into three pairs: who and what, when and where, and why and how. Responses to these interrogative questions define bounded focal phenomena for formulating computational process theories.

Who and What

To define focal phenomena, we may naturally begin by considering *who* the focal phenomena involves and *what* they tend to do. The running example concerns itself with leaders managing team members performing tasks, which implies that we consider (1) the *who* of leaders and team members (i.e., our actors) and (2) the *what* of leadership and task performance (i.e., our processes). With respect to the *who*, we want to consider static actor characteristics and dynamic actor states. The example model considers actors who occupy static roles (i.e., leaders and team members) that do not change during the temporal period of interest. At the same time, the example model considers leader expertise to lead teams on tasks of interest and team member expertise to perform those tasks as dynamic states that generally improve with task experience. With respect to the *what*, we think about the actions leaders and team members enact as performance episodes unfold. The example model provides the possibility to consider two different operational leadership styles: autocratic and delegative. Under the autocratic leadership style, leaders assign members to tasks and choose how team members should perform their assigned tasks (i.e., the formal leader dictates task operations to team members). Under the delegative leadership style, team members both assign themselves to tasks and choose how to perform their assigned tasks (i.e., the formal leader delegates task operations to team members). In both cases, only team members go about actually performing tasks. In summary, by considering the *who* and *what* of the focal phenomena, we define the actors and their actions relevant for the focal phenomena.

When and Where

We may next consider the *when* and *where* of the focal phenomena. With respect to the *when*, we want to consider when actors act and environments update and the general time span under consideration. The example model consists of actors taking actions during performance episodes with several sequential performance episodes contributing to the accomplishment of a team project. With respect to the *where*, we want to consider how task and social environments shape and constrain actors and their actions. The example model consists of several tasks which team members perform. Each task differs with respect to how much it can contribute to project accomplishment in a particular performance episode. Furthermore, each task consists of different ways team members might perform it, and those different ways of performing tasks result in different performance rewards upon task completion. Both task properties constrain overall task accomplishment and shape actor functioning. Team members also do not automatically perform assigned tasks. Each team member comes with other work obligations, outside of the focal team. These other obligations may prohibit any team member from performing the focal team's work. Yet, leaders, under autocratic leadership, can motivate team members (or team members, under delegative leadership, can motivate each other) to increase their work priorities for their focal teams, which highlights how the social environment impacts work processes. In summary, by considering the *when* and *where* of the focal phenomena, we define relevant temporal dynamics and contextual elements for the focal phenomena.

Why and How

Lastly, we may consider the *why* and *how* of the focal phenomena. When we consider the questions of *why* and *how*, we consider possible process interventions for changing unfavorable outcomes (e.g., poor performance). With respect to the *why*, we consider the reasons behind actions we observe. In the example model, leaders and team members wish to maximize their task performance and attempt to make decisions accordingly. With respect to the *how*, we want to know the specific ways in which actors execute actions. In the example model, under autocratic leadership, leaders perceive team member task expertise, and they rank-order team members based on these perceptions. Furthermore, leaders possess awareness of the reward value of performing each task. Leaders assign team members to tasks by matching their perceptions of team member expertise to the reward values of tasks. Under delegative leadership, team members perceive the expertise of each other to perform tasks with team members coming to terms on a rank-order based on their averaged expertise perceptions. They then assign themselves to tasks by matching their rank-ordered averaged expertise perceptions to the reward values of tasks. This discussion highlights how the same process (e.g., leadership) may operate differently (e.g., autocratic versus delegative). In summary, by considering the *why* and *how* of the focal phenomena, we define the plausible set of generative mechanisms responsible for actions behind focal phenomena.

Fundamental Elements

The discussion of focal phenomena makes clear that we define two fundamental elements as we develop computational process theories: actors and environments. For computational process theories of organizational phenomena, actors may consist of individuals, dyads, teams, organizations, and/or networks. At the same time, actors operate in environments, which may consist of work tasks, social and team norms, organizational policies, and/or available resources. Environments shape, constrain, and/or facilitate what actors do, and actors, in turn, update their environments, especially the environments internal to organizations (e.g., work tasks, team norms).

Actors

Conceptualizing actors typically entails describing a core set of characteristics and/or states that actors possess. Although real actors possess many attributes, we usually only represent a targeted number of these qualities in computational process theories. As with any theory development effort, determining which qualities to incorporate into any single computational process theory constitutes an important part of the theory development process. When identifying the attributes of actors within computational process theories, arguably the most important criterion consists of establishing their relevance with respect to how actors function and/or the focal phenomena emerge. In this way, we maintain the overarching principle of balancing parsimony with completeness. For example, leaders and team members possess a variety of characteristics and states, including demographics, knowledge, skills, abilities, personality, positive and negative affect, cohesion, work satisfaction, and goals, to name but a few (e.g., Tuncdogan et al., 2017). However, attempting to represent how, when, and why all these characteristics and states affect how actors operate may prove unnecessary and/or overly complicate any given theory. The example model focuses on just a few actor characteristics and states: role (leader versus team member), leadership and task expertise, perceptual accuracy of task expertise, social influence, work priority, and performance. These characteristics and states help us explore how leadership styles impact team performance in a bounded theoretical space.

Environments

Conceptualizing environments in computational process theories typically entails describing the contexts of actors. Organizational phenomena generally involve navigating task and social environments, although specific theories may vary in their primary focus and the extent to which they incorporate different environmental contexts. Task environments shape what actors do with respect to performing their roles or jobs. Social environments shape how actors might interact, communicate, and collaborate with each other. Similarly to defining actors, when we define environments, the most important criterion to consider consists of their relevance to how actors function and/or the focal phenomena emerge. With respect to the task environment, the example model consists of several tasks that differ with respect to performance value, and only one team member may perform a task in any given performance episode. Those task properties shape how leaders (under autocratic leadership) or teammates (under delegative leadership) assign team members to tasks. With respect to the social environment, team members possess different priorities for performing tasks and only perform assigned tasks in a performance episode as a function of work priorities, which may update across performance episodes as a function of leader (under autocratic leadership) or team member (under delegative leadership) influence. In this way, task and social environments shape what actors do in the example model.

Yet, actors may also update their task and social environments, and, in turn, respond to those updated environments reciprocally (Brass & Borgatti, 2019; March, 1991; Sayama, 2015). With respect to the task environment, when actors work on a task in one period, such work contributes to project accomplishment (Van Der Vegt et al., 1998). In response, actors who experience the task environment change, update their subsequent actions. In the example model, leaders or team members update their perceptual accuracies for judging task expertise based on observed task contributions. With respect to the social environment, under autocratic leadership, leaders develop their social influence as a function of how well team members perform their tasks. In turn, leaders use their social influence to motivate team members’ work priorities for subsequent performance episodes. Under delegative leadership, team members develop social influence as a function of how they perform their tasks, which, in turn, they use to motivate the work priorities of teammates. In both cases, the social environment changes as team members perform tasks, and, in turn, the decisions and actions of leaders and team members change as they respond to their updated social environment.

Process Operations

When we develop computational process theories, we give great attention to detailing how processes operate. Process operations bring together three aspects of interest for computational process theories. We describe the operations of specific mechanisms that generate actions and events, which we can summarize into construct states to assess construct relations.

Mechanisms

Mechanisms entail the rules that actors and environments follow in computational process theories. Rules define (1) how actors act, (2) how actors interact, (3) how environments evolve on their own and/or as a function of actors, and/or (4) how actors update as a function of each other and/or evolving environments (Sayama, 2015). Together, the mechanisms of actors and environments serve as the engines of action and event sequences in computational process theories. The example model consists of several mechanisms that operate to generate actions and events during each performance episode. Under autocratic leadership, we develop rules for how leaders (1) evaluate team member expertise, (2) assign tasks to team members, (3) choose task actions for team members, and (4) update their expertise, perceptual accuracies, and social influence. Under delegative leadership, we develop the same rules except team members operate instead of leaders. Under both leadership styles, we develop rules for how team members (1) prioritize their work, (2) receive rewards for task performance, and (3) update their expertise. To make our discussion tractable, we discuss the computational details of evaluating expertise and task assignment here, and we provide the computational details of other processes and their mechanisms in Appendix B and the supplemental materials.

To eventually assign team members to tasks, assessors (either leaders or teammates) must first evaluate the expertise of team members to perform tasks. The evaluation process consists of establishing a validity correlation between team member expertise to perform tasks and an evaluator's assessment of how well a team member can perform tasks. To establish such a validity correlation, we implement the evaluation process via the following equation (see Scullen et al., 2005, for the use of the same equation to establish a validity correlation in a performance management context):

$$EP = EPA * ME + \sqrt{(1 - EPA^2)} * Error. \tag{1}$$

In Equation 1, *EP* represents an evaluator's perception of a team member expertise's to perform tasks, *ME* represents a team member's true expertise to perform tasks, *EPA* represents an evaluator's perceptual accuracy for evaluating a team member's expertise to perform tasks (i.e., the validity correlation), and *Error* represents a term created by sampling from the standard normal distribution to reflect the lack of validity between perceptions of expertise and true expertise. This evaluation process happens per evaluator and team member pair in each performance episode. Furthermore, as team members perform tasks, evaluators update their perceptual accuracy to judge team members, and team members update their expertise to perform tasks. Such updating represents learning from work experience. In other words, the components of the evaluation process update dynamically across performance episodes. From this description, we can see how we set into motion the action sequences that unfold as we simulate the model.

The task assignment process follows the evaluation process. Once evaluators make their perceptual assessments of team members, they rank team members by their perceptions from most to least capable. For autocratic leadership, leaders rank team members based on their singular perceptions; for delegative leadership, team members rank themselves based on their averaged perceptions of each other (see Broomell and Davis-Stober, 2023, and Page, 2018, for a discussion of the "wisdom of the crowd" as a basis for the delegative averaging procedure). In either case, upon completing the ranking of team members, relevant actors (i.e., leaders or team members) match the team member perceived as most capable to the most valuable task, the team member perceived as second most capable to the second most valuable task, and so on. The matching of team members to tasks changes each performance episode as a function of the evaluation process. In turn, the task assignment process contributes to subsequent processes. From this description, we see how different processes impact each other as each process generates its actions. We provide the details of other processes in Appendix B and the supplemental materials.

Actions and Events

Actors enact actions and environments evolve as a function of corresponding generative mechanisms. For organizational actors, affective actions define how actors express their emotions and feelings in their work environment (e.g., Maruping et al., 2015); behavioral actions define how actors behave in their work environment (e.g., Tuncdogan et al., 2017); cognitive actions define how actors decode, evaluate, and encode their work environment (e.g., Grand et al., 2016); and social actions define how actors initiate interactions with one another in their work environment (e.g., Kellen et al., 2021). Similarly, environments evolve according to generative mechanisms that may define how social and task environments update. In our example model, actors evaluate team members, assign team members to tasks, decide task action strategies, apply their work priorities, and perform tasks. This action sequence represents actions stemming from generative mechanisms, and an action generated from one mechanism initiates a subsequent mechanism to generate the subsequent action. Furthermore, the choice and act of performing leads to a performance reward event, which, in turn, leads to actors updating their expertise, perceptual accuracies, priorities, and social influence as a function of their own mechanisms. These updates feedback into mechanisms to generate actions in the next performance episode. In summary, the sequences of actions and events reflect the operations of generative mechanisms.

Construct States

As actors act, interact, and respond to their environments, their construct states update. For instance, an actor who performs a task may do so effectively or ineffectively (i.e., the state of the task performance construct updates). Furthermore, as processes unfold in the form of series of actions, we may summarize actions into intermediate (e.g., month to month sales) and cumulative (e.g., sales for the year) construct states. Indeed, survey-based construct measurement generally asks respondents to summarize past actions, which serve as a basis for data analyses that evaluate narrative construct theories.

Intermediate and cumulative states commonly serve as checkpoints that actors pay particular attention to as they accumulate, and thus, construct states can serve as inputs to generative mechanisms for subsequent periods (Ilgen et al., 2005; Kozlowski et al., 1999; Marks et al., 2001). For instance, team leaders want their teams to perform better than other teams, organizational leaders want employees engaged in their work, and workers want to experience high levels of job satisfaction. These desires, expressed as construct states (i.e., high levels of employee performance, engagement, and satisfaction), impact how actors will act in subsequent periods. Importantly, construct states emerge from generative mechanisms (Marks et al., 2001).

In the example model, one focal construct pertains to project accomplishment across tasks. Indeed, at the conclusion of each performance episode, we can calculate the performance of each team member, team, and organization. Furthermore, we can track the trajectories of team member, team, and organizational performance across performance episodes. If actors feel their performance trajectories do not meet desired goals, then they can react by considering their work processes. Just like actions, actors assess their construct states via process rules, and, in turn, update subsequent work processes. Importantly, these construct measurements do not inform us about actual underlying generative mechanisms. Instead, they serve as bellwether assessments of process happenings. As stated, everything that happens, happens through process rules.

Computations

Once we define the elements of a computational process theory, then we may formalize those elements with computations. To facilitate the transition to a computational formalization, we begin by writing pseudocode. For clarity, we use the term pseudocode to refer to a higher-level representation of a computational process theory, which may include a listing of computational steps, diagrams, equations, etc., with varying specificity. After we write pseudocode, we translate it into formal, computational code, which we ultimately simulate to understand processes in action.

Pseudocode

One form of pseudocode consists of a step-by-step listing of process operations and resulting actions and events that unfold as actors act and environments evolve. We write pseudocode to provide a mapping (or translation) from a process theory to computations representative of the mechanisms, actions, and constructs of the theory. When we write pseudocode, we can write it with different levels of specificity ranging from a description of high-level action steps to computational details of implementing each action step. For organizational researchers beginning to take steps toward developing computational process theories, we recommend writing pseudocode that captures high-level action steps first, and then, refining the high-level pseudocode into more detailed pseudocode as theorists implement their process theories in computer code. Overall, pseudocode provides a starting foundation for formulating and formalizing process theories.

We provide an example of a high-level pseudocode in Table 3 for the example computational process model, and, in Appendix B, we provide a detailed pseudocode that describes the computations for all processes. Pseudocode generally provides a description of how to initialize a model, and then, how processes operate. To initialize a model, we define (1) actors of interest including their characteristics and states, (2) the task, social, and/or other relevant environments, and (3) time frames and/or contexts. The example model initializes leaders and team members as connected actors who work in teams of an organization, tasks with varying value that team members perform during performance episodes, and parameters that control aspects of processes (Steps 1-3). After initialization, we define processes that operate during each performance episode. The example model defines the operational rules for the following

Table 3

Pseudocode for a Computational Process Model of Leadership Styles and Team Performance

Step	Description
Initialization	
1	For both leadership styles, initialize: (1) the number of leaders, team members, tasks, task actions, and performance episodes, (2) the maximum task reward value, (3) process parameters pertaining to actor social influence, expertise, and perceptual accuracies, and (4) a process parameter for team member work priorities.
2a	For autocratic leadership, (1) establish a connection between leaders and team members in an organization, (2) initialize leader and team member expertise by drawing from a relevant statistical distribution, (3) initialize team member work priorities by drawing from a relevant statistical distribution, (5) set leader perceptual accuracies and social influence to zero, (5) establish a connection between teams and work tasks, and (6) initialize task reward values via a probability function that samples higher reward tasks less frequently.
2b	For delegative leadership, (1) establish a connection between leaders and team members in an organization, (2) initialize team member expertise and work priority by drawing from relevant statistical distributions, (3) establish connections between team members on the same team, (4) set team member perceptual accuracies and social influence to zero, (5) establish a connection between teams and work tasks, and (6) initialize task reward values via a probability function that samples higher reward tasks less frequently.
3	For both leadership styles, rank tasks for each team from most to least valued.
Processes	
4a	For autocratic leadership, leaders perceptually evaluate team member expertise to perform tasks as a function of team member actual expertise and their own perceptual accuracy according to a relevant validity equation.
4b	For delegative leadership, team members evaluate each other's expertise as a function of each target team member's actual expertise and each evaluator's perceptual accuracy according to a relevant validity equation.
5a	For autocratic leadership, leaders rank team members from most to least capable by their perceptual evaluations.
5b	For delegative leadership, team members rank themselves from most to least capable by their averaged perceptual evaluations.

Step	Description
6	For both leadership styles, actors match team members to tasks in matching rank-order so that the most capable team member matches with the most valued task, the second most capable team member matches with the second most valued task, and so on.
7a	For autocratic leadership, leaders decide what action strategies team members should employ to perform their assigned tasks via an ordinal probability function defined by leader expertise to choose the first, second, etc., most effective action for each task.
7b	For delegative leadership, team members decide what action strategies they should employ to perform their assigned tasks via an ordinal probability function defined by team member expertise to choose the first, second, etc., most effective action for each task.
8	For both leadership styles, team members determine if they will perform their assigned tasks as a function of their work priorities relative to a threshold value drawn from a relevant statistical distribution.
9	For both leadership styles, team members who chose to perform their tasks do so via the selected action strategy and accumulate a performance reward as a function of the proportional quality of their action strategy relative to task value. Team members who chose not to perform their assigned tasks, in turn, do not accumulate performance rewards.
10a	For autocratic leadership, (1) team members update their expertise by a small amount drawn from a relevant statistical distribution if they performed a task, (2) leaders gain social influence over team members as a function of the proportional degree of performance reward team members received plus a small amount drawn from a relevant statistical distribution, (3) team members update their work priorities as a function of a draw from a relevant statistical distribution plus the updated leader influence, (4) leaders update their perceptual accuracy of a team member by a small amount drawn from a relevant statistical distribution if the team member performed a task, and (5) leaders update their expertise as a function of a draw from a relevant statistical distribution multiplied by the proportion of team members who performed their assigned tasks.
10b	For delegative leadership, (1) team members update their expertise by a small amount drawn from a relevant statistical distribution if they performed a task, (2) team members gain social influence over teammates as a function of the proportional reward teammates received for performing their tasks plus a small amount drawn from a relevant statistical distribution, (3) team members update their work priorities as a function of a draw from a relevant statistical distribution plus the updated average influence of teammates, (4) team members update their perceptual accuracies of teammates by a small amount drawn from a relevant statistical distribution if their teammates performed a task.
11	For both leadership styles, processes continue until the last performance episode.

processes: perceptual evaluation of team members (Steps 4a and 4b), ranking of team members by perceptual evaluations (Steps 5a and 5b), assigning tasks to team members (Step 6), identifying task action strategies (Steps 7a and 7b), team members deciding if they will perform their assigned tasks (Step 8), accumulating performance rewards upon task execution (Step 9), and actors updating the status of their expertise, perceptual accuracies, social influence, and work priorities (Steps 10a and 10b). Processes operate until the last performance episode (Step 11).

Code

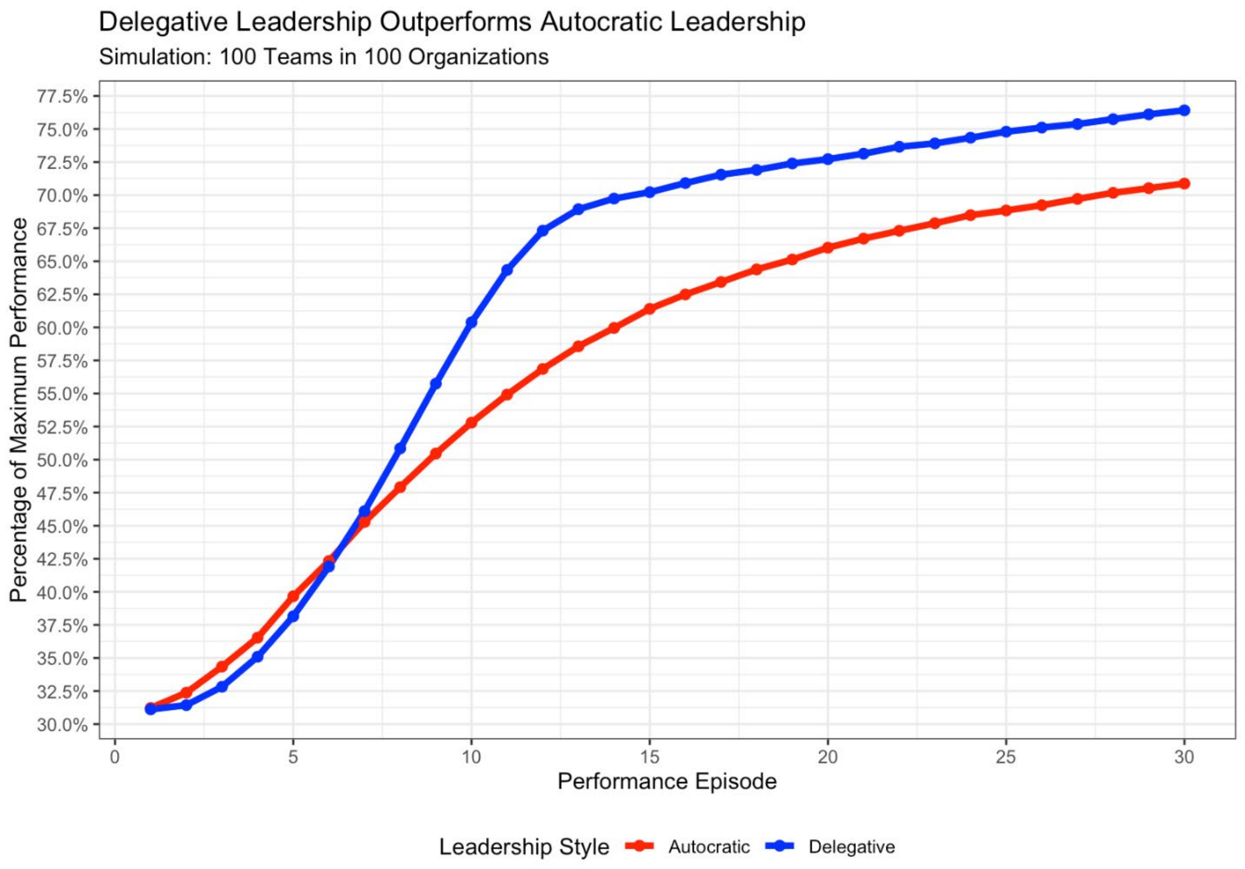
Once we write pseudocode, we may translate it into computer code. In contrast to writing pseudocode, writing executable computer code requires knowledge of programming and software applications designed for developing computational process models (e.g., *R*, *Python*, *Julia*, *NetLogo*). The supplemental materials provide the complete computational code written in the *R* programming language for the example computational process model we present (see Grand et al., in press, for a tutorial on programming fundamentals with *R*). To create the computational code, we converted each step of the detailed pseudocode presented in Appendix B into tractable instructions using equations, functions, logical statements, and control loops. We wish to emphasize that even if researchers do not develop a computer-executable version of their process theories, writing pseudocode still proves useful in the sense of improving the specificity and quality of theory development as it directs attention to the process operations proposed to generate focal phenomena. Such pseudocode facilitates the eventual writing of complete computational process models. Thus, we encourage organizational theorists to minimally consider writing pseudocode to represent their process theories to facilitate the eventual conversion of theory into computations.

Simulate

Once we instantiate a computational process model into code, we can simulate it to advance our understanding of the focal phenomena under consideration. In this section, we provide simulation results from the example model executed under the parameter values specified in Appendix B. We begin by showing the focal result of interest. In Figure 4, we see that teams operating under delegative leadership eventually outperform teams operating under autocratic leadership. Next, we utilize the simulated data to highlight explanations offered by a narrative construct theory versus a computational process theory in accounting for the emerged mean difference favoring delegative teams. In doing so, we illustrate the links among the

Figure 4

Comparing Two Leadership Styles on Team Performance

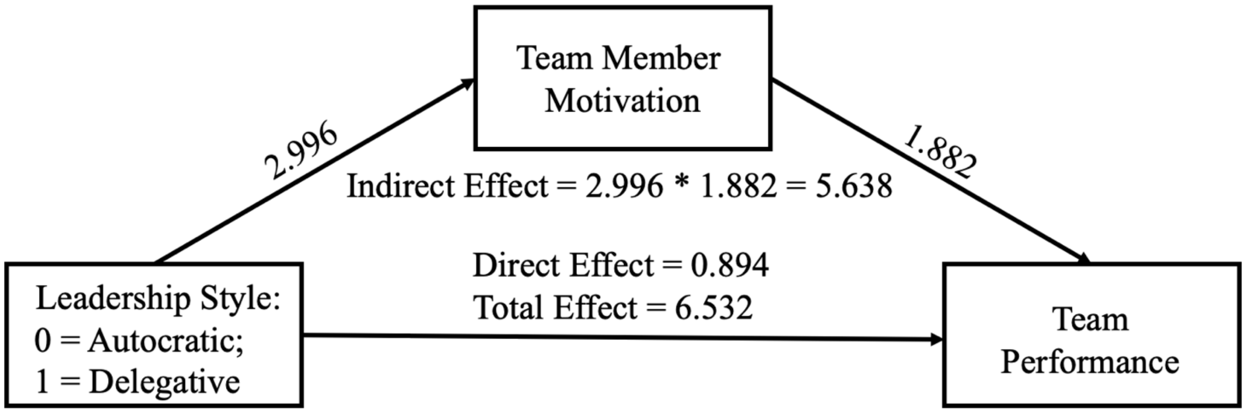


components of Figure 2 – generative mechanisms, action sequences, construct relations – with the simulated data.

We assume a narrative construct theory leads us to the construct mediation model, presented at the top of Figure 2, in which team member motivation transmits the effect of leadership style to team performance. Hence, team member motivation serves as a mediator and represents the explanatory mechanism for our narrative construct theory. We define motivation as individuals directing their efforts toward accomplishing goals (Locke & Latham, 1990). In the example model, we use the priorities team members hold for performing their assigned tasks as the representation for their motivation levels. In Figure 5, we present the construct mediation model as a path diagram fitted to some of the data generated by our computational process model. The data we use consists of sampling, from a single performance episode in the later

Figure 5

Construct Path Diagram



stages of unfolding process operations (i.e., performance episode number 20), 100 teams operating under autocratic leadership and another 100 teams operating under delegative leadership akin to the data we might collect in an actual empirical study (i.e., cross-sectional data on teams operating under the two leadership styles). We rescaled the range of team member motivation and team performance, from zero to one to zero to 100, for easier interpretability of the path coefficients. From this data, we compute that the indirect effect from leadership style to team performance via team member motivation accounts for 86.3% of the total effect from leadership style to team performance (i.e., 5.638 / 6.532 = 0.863). As an important aside, we note that estimating the same mediation model on empirical data would require sufficiently addressing endogeneity concerns (i.e., the possibility of omitted causes), such as using the instrumental variables approach (Antonakis et al., 2010; Bastardoz et al., 2023; Kline, 2015; Smith, 2012), which we do not need to worry about in this case given our knowledge of the generating mechanisms of the computational data. The results presented in the path diagram suggest team member motivation provides essentially a "full accounting" of the effect from leadership styles to team performance, giving us a sense of initial confidence that we can explain the focal result of interest (i.e., the discrepancy in team performance favoring delegative teams).

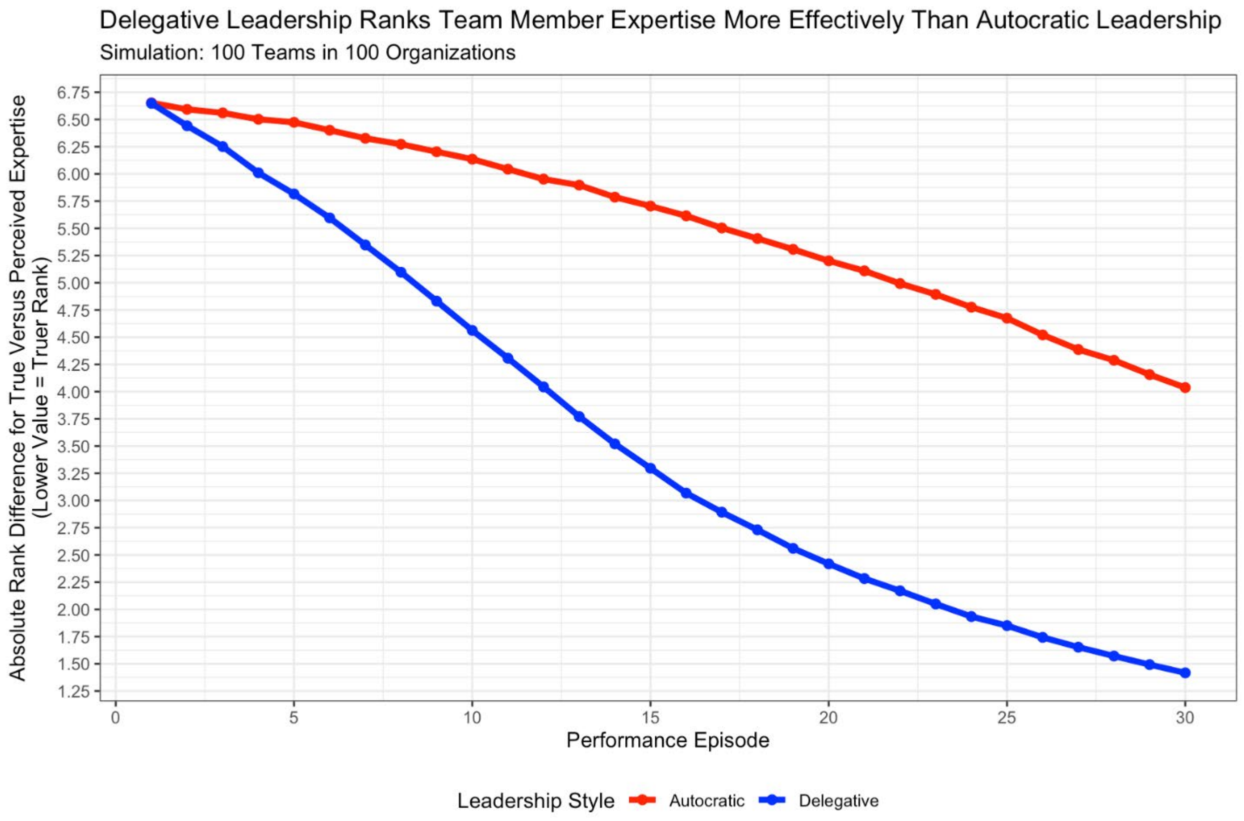
Yet, we remain uninformed with respect to two issues. First, if we suggest to autocratic leaders that they should work on increasing team member motivation to eventually improve team performance, the path diagram representing our narrative construct theory does not inform us exactly how autocratic leaders can increase team member motivation. We remain unaware of possible generative mechanisms in operation that ultimately drive team member motivation. Second, the strong indirect effect may mislead us into thinking that only generative mechanisms associated with team member motivation prove important. Thus, on the one hand, we derived a true causal construct mediation model, but, on the other hand, we remain unaware of processes and generative mechanisms in operation, which prevents us from instantiating an efficient and effective intervention to address team performance shortcomings.

For a deeper understanding of the differences in team performance between the two leadership styles, we examine the operations of one of the processes in our example model: evaluating team member expertise. For teams to optimize their performance, they need evaluators to correctly evaluate team member expertise. Otherwise, the subsequent process of assigning team members to tasks will mismatch the expertise of team members with the value of tasks, which, in turn, holds downstream consequences for subsequent processes such as team members optimizing performance rewards. To examine the evaluation process, we access the simulated process data. We first highlight a difference between the two leadership styles in the evaluative action, and then, we highlight a difference in the mechanism responsible for the evaluative action.

For any given performance episode, we know the true expertise and the perceptual evaluation of expertise for all team members. Thus, we can rank team members separately by their true and perceived expertise, and then, compute the absolute difference in those two ranks. The absolute difference between those two ranks represents the extent to which evaluators correctly ranked team members on their true expertise with a lower absolute difference in ranks indicating more accurate rankings. In Figure 6, we see that, across performance episodes, teams operating under delegative leadership gain a widening advantage with respect to more accurately ranking team members on their expertise than teams operating under autocratic leadership. By showing differences in the evaluative action, we derived one of the actions highlighted in the

Figure 6

Comparing Two Leadership Styles on Team Member Expertise Ranks



middle of Figure 2. Our construct mediation model indicated that team member motivation seems to provide a rather full accounting of the effect from leadership style to team performance. Yet, the differences in the evaluative action for the two leadership styles demonstrates the critical importance of the evaluation process to the eventual emergence of differences in team performance.

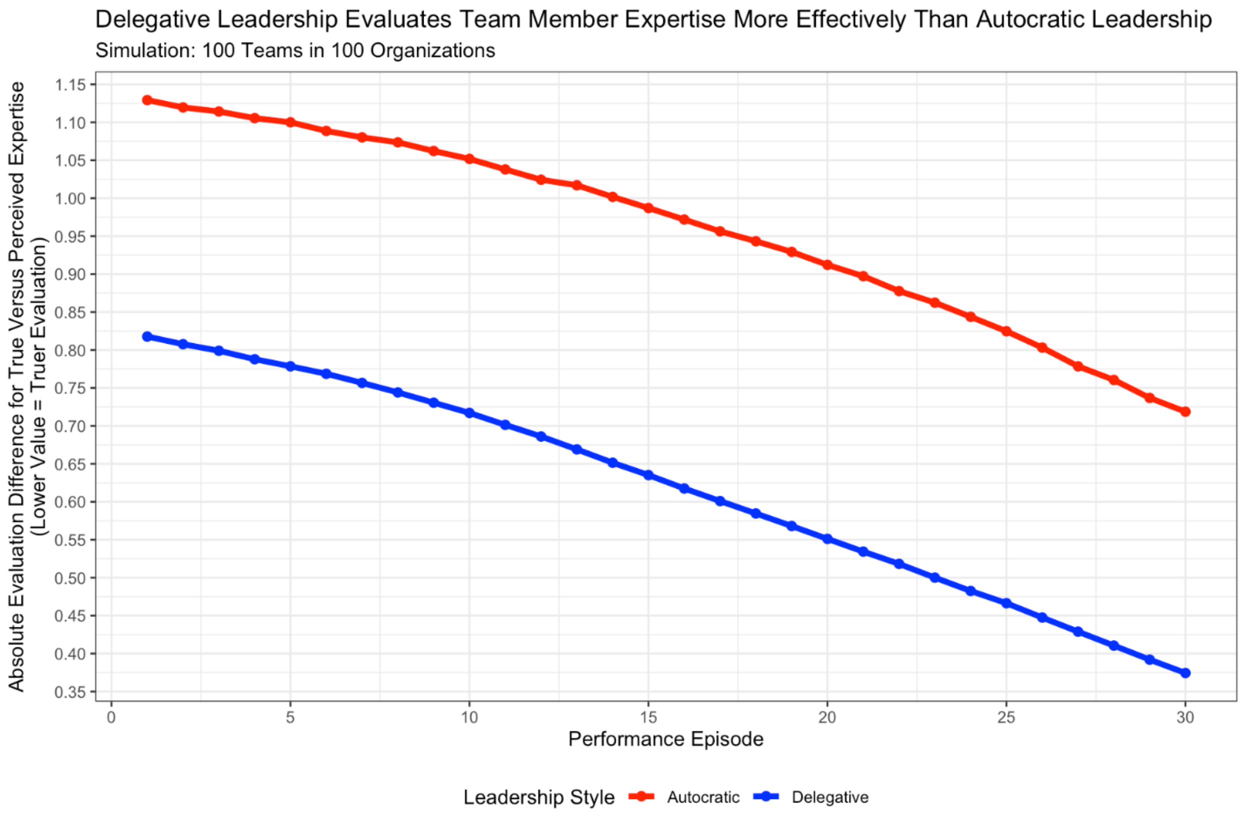
The differences in the evaluative action stem from the generative mechanisms of the evaluation process. Teams operating under an autocratic leadership style rely on their leaders to evaluate the expertise of team members. In this case, each leader represents a single viewpoint (perspective) for evaluating team member expertise. When we consider this operational procedure, we know that, across team members, autocratic leaders will, on average, make unbiased evaluations, as they will overestimate the expertise of some team members but underestimate the expertise of other team members. That fact suggests autocratic leaders might execute the evaluation process well. However, at the same time, the single viewpoint also implies that, for any single team member, autocratic leaders will make biased evaluations, either over- or under-estimating the team member's actual expertise. Ultimately, these biased evaluations of single team members lead to leaders less effectively ranking team members by their expertise.

Teams operating under a delegative leadership style rely on all team members to evaluate the expertise of their teammates. In this case, each team member represents a single viewpoint (perspective) for evaluating teammate expertise. Just like a single leader, any single team member makes biased evaluations of any single teammate by either over- or under-estimating the teammate's expertise. However, when team members average their individually biased evaluations for a single teammate, they make less biased evaluations of each teammate. As a result, the multiple viewpoints provided by several team members allows them to more effectively rank teammates by their expertise. We note that the evaluative mechanism under delegative leadership operates similarly to a peer-based 360-degree feedback system and provides a similar evaluative advantage (London & Beatty, 1993). More generally, the evaluative mechanism under delegative leadership operates as a "wisdom of the crowd" (Broomell & Davis-Stober, 2023; Page, 2018) and follows Condorcet's jury theorem (Ladha, 1992), all of which provide credibility for its operation.

We can measure how well assessors evaluate the expertise of team members by computing the absolute difference in true versus perceived expertise. Just like for rankings, a lower absolute difference between true and perceived expertise indicates a more accurate evaluation. In Figure 7, we see that teams operating under delegative leadership more accurately evaluate team members on their expertise than teams operating under autocratic leadership right from the first performance episode and maintain a constant advantage across performance episodes. The observed difference in evaluation accuracy between the two leadership styles directly reflects the differences in the evaluative mechanism of the two leadership styles (i.e.,

Figure 7

Comparing Two Leadership Styles on Team Member Expertise Evaluations



relying on an evaluation consisting of a single perspective for autocratic leadership versus relying on an evaluation consisting of an average of multiple perspectives for delegative leadership). By showing differences in the evaluative mechanism, we derived one of the generative mechanisms highlighted in the bottom of Figure 2. Most importantly, the difference in the evaluative mechanism for the two leadership styles generates the differences we observe in the evaluative action, which, ultimately, holds downstream consequences for subsequent processes.

By considering the evaluation process in depth, we gain a much more complete understanding of how differences in team performance emerge for the two leadership styles. When we applied a construct approach, we discovered a true insight that team member motivation matters for team performance. Yet, at the same time, the construct approach offered a misleading indication of the importance of team member motivation in accounting for differences in team performance between the two leadership styles. Furthermore, the construct approach did not provide insights into how to enact an operational intervention. On the other hand, when we applied a process approach, we examined the difference in the generative mechanism the two leadership styles apply in making evaluations of team members. By doing so, we began to untangle the operational differences in how the two leadership styles execute pertinent processes that ultimately underly observed construct relationships portrayed in the mediation model and account for the observed differences in team performance. In turn, our deeper understanding of the processes in operation affords us the possibility of eventually instantiating an efficient and effective intervention to address team performance shortcomings (i.e., under the specified conditions presented, autocratic leaders should implement delegative mechanisms).

Advancing Theory Development in Organizational Science

Computational process theories afford several unique advantages for developing theoretical accounts of organizational phenomena. In this section, we highlight and elaborate several ways in which computational process theories offer certain "default" advantages for developing effective theories of organizational phenomena. We provide a summary of key arguments in Table 4.

Explanations

The identification and formalization of generative mechanisms in computational process theories serves as a central tenet of theoretical explanation. At first blush, organizational researchers might think narrative construct theories already offer such explanations, and thus, might question whether computational process theorizing offers anything different. However, we assert that narrative construct theories seldom address explanatory processes or mechanisms with sufficient clarity (Braun et al, 2022; Mohr, 1982). Generative mechanisms define rules that represent how actors/environments relevant to phenomena of interest operate and perform functions. Multiple generative mechanisms entwined together drive the emergence of

Table 4

Theory Development via Narrative Construct Theories Versus Computational Process Theories

Comparison	Narrative Construct Theories	Computational Process Theories
Explanations	Narrative argumentation concerning relationships between constructs	Specification of computations as representations of process operations
Hypotheses	Propose construct hypotheses focused on linear relationships	Propose process hypotheses focused on sequences of actions or research foci dedicated to explorations of generative mechanisms
Interventions	Interventions focused on changing levels of constructs	Interventions focused on altering operations of generative mechanisms
Specificity	Typically consists of linear directional construct hypotheses, construct measurement, and construct interventions	Aids in the development of hypotheses with expected effect sizes, identifying precise targets for measuring phenomena focused on actions, and informing the design of targeted manipulations focused on generative mechanisms
Complexity	Generally, does not incorporate dynamics, bottom-up emergence, self-organization, open environments, nor non-linearity	Typically incorporates dynamics, open environments, bottom-up emergence, self-organization, and non-linearity
Logic	No inherent checks on the consistency of theoretical logic	Explicit check on theoretical logic as a function of specifying and simulating computations
Empirical Results	No method to account for known empirical results due to a lack of simulated logic	Explicit check on whether simulated computational logic accounts for known empirical results

Comparison	Narrative Construct Theories	Computational Process Theories
Assumptions	Uncommon to specify assumptions explicitly and no method to formally evaluate any explicit assumptions	Explicit specification of assumptions and possibility to evaluate assumptions in computational environment

action/event sequences. These action/event sequences denote actor/environment processes in operation. From processes in operation, we may summarize actions to measure/represent construct states (Kozlowski, 2022). Thus, we observe variance in levels of constructs and/or covariance among constructs because a series of actions/events unfold as a function of activated generative mechanisms.

Computational process theories position processes (i.e., the identification of a series of actions/events capable of representing phenomena) and generative mechanisms (i.e., the specification of how actions/events occur) as the critical targets of theory development. As a point of difference, empirical support for narrative construct theories, which generally focus on predicting construct covariances, does not confer direct support for the explanations incorporated in narrative construct theories. In other words, the observation of a construct relationship does not directly explain how the construct relationship emerged (Kozlowski, 2022). On the other hand, computational process theories directly account for the emergence of construct relationships by focusing explanations on process actions and generative mechanisms that produce those actions. Coming back to the example model, a process explanation for team performance differences between the autocratic and delegative leadership styles would focus on processes in operation, such as the evaluative process we discussed in presenting the simulation results. Generating process explanations proves critical to organizational science because it affords a principled window into exploring how variances and covariances among organizational constructs emerge.

Hypotheses and Interventions

Given the focus of computational process theories on process explanations, we may formulate different types of hypotheses and interventions relative to narrative construct theories.

Of note, we may derive process hypotheses and interventions in which a theorist proposes expected sequences of actions to occur as a function of generative mechanisms. Process hypotheses make more specific and granular predictions, and thus, they make process theories easier to falsify, which constitutes a desired characteristic of all theories (Meehl, 1978; Popper, 2002). Indeed, we can simulate computational process theories to make several types of predictions including the emergence of patterns (e.g., Grand et al., 2016), short-term and long-term expectations (e.g., Samuelson et al., 2019), the optimization of results for a particular entity (e.g., individuals) compared to a collective of entities (e.g., teams) or vice versa (e.g., Kuljanin, 2011), and the illumination of multiple, branching pathways of actions (e.g., Kuljanin et al., 2021). A process intervention looks to directly influence the actions of actors by intervening through the mechanisms that generate (cause) actions. Interventions targeted to specific actions lead to effective and efficient emergence of desired outcomes (see Grand et al., 2016, as an empirical exemplar). Coming back to the example model, we would provide specific advice to autocratic leaders to focus on incorporating multiple perspectives in their evaluative process to eventually bolster team performance as opposed to providing generic advice about increasing team member motivation, which does not reference the operational mechanism in play. Such process hypotheses and interventions advance empirical investigations undertaken in organizational research (Hazy, 2008).

To further highlight such an advancement, consider the example of leader behaviors and team performance. In a two-condition experiment, a construct intervention might tell leaders in the experimental condition to “direct and consider their team members frequently,” while leaders in the control condition might receive some neutral, non-leadership advice. Assuming a correct theory, such a construct intervention would produce a causal effect on team performance in favor of the experimental condition. If we can replicate such an effect, then organizations may safely implement such an intervention as a successful policy without necessarily requiring a process explanation. At the same time, we would also observe variance in team performance even when just looking at the experimental condition because undoubtedly leaders would take the generic advice provided and implement it in a variety of ways, some more successfully and others less so. Furthermore, team members might respond to the same kind of leadership in very different ways. Altogether, these comments imply that advice to increase the quality or quantity of directive leader behaviors will misfire quite often with respect to improving team performance. Not all ways of directing will produce the same effect nor will the same way of directing produce the same effect every time.

Construct interventions, as opposed to process interventions, remain one step removed from specific actions and two steps removed from the mechanisms that generate those actions (Kozlowski, 2022). Recommendations from construct interventions lack mechanistic specificity due to their focus on construct levels and covariances (Braun et al., 2022). In contrast, computational process theories draw attention to and propose what specific directives leaders should enact, when and how they should enact them, and whom they should target with their actions. Such advice pierces into the generative mechanisms that leaders may deploy to motivate team members and impact their performance. In summary, we might successfully implement a construct intervention as an organizational policy without knowing generative mechanisms, but knowing the generative mechanisms affords us the opportunity to more efficiently and thoroughly control how outcomes emerge.

Specificity

Given their focus, narrative construct theories tend to advance hypotheses about the relative levels of constructs and relations between levels of constructs (i.e., construct hypotheses). However, organizational scientists routinely criticize construct hypotheses (e.g., linear directional construct-to-construct relationships) commonly advanced by narrative construct theories for their lack of specificity, the relatively weak evidentiary bar they pose for evaluating claims, and their general triviality (Antonakis, 2017; Edwards & Berry, 2010; Sutton & Staw, 1995). These limitations inhibit the effective accumulation of knowledge (Wulff et al., 2023). Directing attention to the underlying processes and generative mechanisms responsible for such construct relationships, by way of developing computational process theories, affords a principled means for advancing rigorous, more informative, and “riskier” (i.e., falsifiable) hypotheses regarding organizational phenomena (e.g., Meehl, 1967, 1978; Simon, 1992; Wulff et al., 2023). For example, researchers can perform simulations of computational process theories to explore/establish competitive process-focused counterfactual scenarios and the extent to which changes in the presence/operation of specific mechanisms predict specific distributional patterns or effect sizes, just as we demonstrated that the difference in team performance emerges as a function of differences in evaluative mechanisms for autocratic and delegative leadership in the example model. In turn, these observations can better inform key decisions related to measurement (i.e., what and when to measure) and research design (i.e., developing research manipulations, constructing study conditions) to aid hypothesis testing and theory validation (Götz et al., 2022; Grand et al., 2016; McDermott, 2023; Wulff et al., 2023). Taken together, computational process theories can aid the formulation of more precise theories and hypotheses to improve the efficiency of organizational research and practice by better informing us how to allocate resources (e.g., time, funds, people) and direct interventions more productively (Braun et al., 2022; Watts et al., 2022).

Complexity

When we develop computational process theories, we position ourselves to incorporate the defining features of complex systems. Complex systems operate in open environments, evolve over time, exhibit non-linearity, dynamically self-organize, and generate emergent phenomena (Aiken, et al., 2019; Gell-Mann, 1994; Kozlowski et al., 2013; Kozlowski & Klein, 2000; Nicolis & Prigogine, 1989; Sayama, 2015; Strauss & Grand, 2022) — features long acknowledged as relevant to understanding the phenomena studied by organizational scientists as well (Hazy, 2007, 2008; Marion & Uhl-Bien, 2001; Rosenhead et al., 2019). With the focus on specifying bottom-up generative mechanisms that produce distinct action sequences, which, in turn, produce patterns of construct relations, we explicitly incorporate such characteristics into the explanatory philosophy of computational process theories.

We can highlight these features of complexity with a team leadership example. Leaders who manage teams over long periods of time undoubtedly deal with open environments requiring non-linear, evolving, and dynamically self-organized responses that result in continuously emergent phenomena. Teams operate in an open environment because team members and tasks come and go. Each change in the environment may necessitate a response by leaders with respect to how their teams should operate in the forthcoming period. Previous operating procedures may no longer apply, and leaders and team members must adapt to and evolve as a function of their changing circumstances (Hazy & Uhl-Bien, 2015). For instance, leaders may need to reallocate team members to tasks, and team members may form new collaborative structures on their own or through the influence of a team leader. This example highlights how leaders and team members may dynamically and continuously self-organize as they respond to their evolving environments. With such responses, the completion of tasks, performance trajectories of individuals, and social appraisals of team members may rise and decline raggedly, change abruptly, and showcase discontinuities, all indicative of non-linear empirical patterns. As leaders and team members respond to their evolving environments, emergent phenomena continuously arise and change, such as the state of a team's cohesion, efficacy, trust, and performance. Computational process theories provide a means for more effectively capturing this reality, and therefore, hold greater potential to generate higher fidelity accounts of organizational complexities (we note the important distinction between theories incorporating complexity versus complicated theories; see Aiken et al., 2019, and Olenick and Dishop, 2022, for more information).

Logic

When we write computational process theories, we can directly evaluate our theoretical logic by allowing the proposed mechanisms to play out over actors, time, and/or contexts (Macal & North, 2010). Generally, we do so by simulating computational process theories on computers. Simulating a computational process theory affords the means to check the theoretical logic by evaluating whether the simulations conclude without logical errors. If simulations generate logical errors unrelated to mistakes in programming/implementation, then we know the theoretical logic faltered somewhere. In such cases, we could fix the logical errors and simulate the updated logic until a simulation completes without such logical errors. Although one might consider such a check very basic, we commonly find that the initial translation of the logic we develop in our minds to the logic we write for computers does produce logical errors — and, in many cases, these logical errors often result from a lack of clarity and specificity in the theory of interest. Thus, such logical errors provide a useful diagnostic for identifying areas requiring further conceptual attention. Importantly, narrative construct theories do not afford such a formal means for evaluating core premises. The theoretical logic of narrative construct theories exists only in narrative form. As such, we cannot subject them to similarly principled assessments, and therefore, we must rely on more subjective assessments (Vancouver et al., 2020). In this respect, converting narrative construct theories into computational process theories advances the evaluation of theories by providing the means to simulate their logic on a computer, which requires precise and specific instructions to generate output without logical errors.

After we check that a computational process theory generates simulated data without logical errors, we can compute anything of interest from the generated data. For instance, using the generated data, we might compute mean differences between groups, correlate variables, fit descriptive statistical models, extract sequences of actions, or visualize patterns of dynamics, just like we demonstrated with the simulated data from the example model. Using these summaries, we can perform additional evaluations of our theoretical logic (see Rand and Rust, 2011, for more information about methods and foci for evaluating computational models/theories). Of note, we can check whether those summaries meet our expectations based on the theory we intended to develop (Epstein, 1999). For example, if we expect to develop a theory in which differences in team performance develop over performance episodes as a function of leadership styles, then we can conduct simulations to verify that the data generated by our computational model exhibits such a pattern. If not, then we either developed faulty theoretical logic, or we did not understand its implications. The former may suggest we need to make changes to the theoretical logic, whereas the latter may suggest a potentially interesting discovery. Importantly, narrative construct theories cannot perform this check without posing new, untested narrative logic. In summary, translation of theory to computations provides an inherent check on theoretical logic.

Empirical Plausibility

In addition to a conceptual evaluation of logic, we may also check whether the simulated data generated by a computational process model reflects what we observe in existing empirical data. Given the emphasis on "default" benefits of theory development in this section, we focus the discussion here on comparing theory with already available empirical results as opposed to comparing theory with new data collection efforts. We discuss what kinds of new data collection efforts can offer thorough evaluations of computational process theories in a subsequent section.

In striving to build a cumulative science, theories should account for known empirical results and not just new empirical results so that we can build integrated, architecturally assembled "bricks of scientific knowledge" on top of each other as opposed to scattering the "bricks of scientific knowledge" all around us (Wulff et al., 2023, p. 2). For any simulated results from a computational process model, we observe some degree of correspondence with known empirical results. In the case of a high degree of correspondence, the computational process theory serves as a plausible explanation for the available empirical evidence. In the case of a low degree of correspondence, we can consider possible issues with the theoretical logic, whether a different set of parameter values or generative mechanisms would reproduce the empirical data, or if altering the operations of certain generative mechanisms might better account for the empirical data. Importantly, narrative construct theories can only perform a weak version of this empirical check because they tend to advance hypotheses without specification of effect sizes (Edwards & Berry, 2010). Hence, computational process theories can advance our assessment of theoretical logic by allowing for more specific comparisons to known empirical results. Theories that can reproduce known empirical results gain credibility, and theories that cannot reproduce known empirical results lose credibility (Borsboom et al., 2021; Eronen & Bringmann, 2021; Guest & Martin, 2021).

Moreover, we can stress test the credibility of any computational process theory against competing theories. In other words, we can develop multiple competing computational process theories to evaluate which theories best reproduce known empirical results, and thus, enact a process-focused counterfactual investigation whereby we consider alternative explanations for known empirical results (Antonakis, 2023; Ballard et al., 2021; McDermott, 2023; Vancouver et al., 2020; Wulff et al., 2023). Once we specify each computational process theory as a model, we can then simulate them and evaluate to what extent they reproduce known empirical results. Sometimes, we might find that one computational process theory reproduces the empirical results better, but other times, we might find that multiple computational process theories reproduce the same empirical results equally well. Such cases can then inform additional process-focused counterfactual empirical research with more targeted data collection goals (e.g., sequences of actions, timing of actions) that we could use to evaluate whether all or only some of the processes proposed by different computational process theories reproduce the empirical results of interest (Antonakis et al., 2010). In any event, constructing and systematically probing computational process theories offers a principled and efficient means for identifying, pursuing, and evaluating explanations for empirical results.

Assumptions

We make assumptions whenever we write theories. Describing and stating our assumptions proves vital to understanding the boundaries, sensitivities, and general limits of our theories. If we do not explicitly state assumptions, then we cannot begin to understand their implications. Generally, when we write narrative construct theories, we do not possess at our disposal any inherent check to clearly specify our assumptions, whereas, when we write computational process theories, we necessarily must specify assumptions explicitly, otherwise such theories cannot compute (Rand & Rust, 2011; Smith & Rand, 2018).

For computational process theories, we make assumptions about actors, environments, interactions between actors, and responsiveness of actors and environments to each other. We decide what inputs and processes to include or exclude and which to fix or control as we simulate computational process models. Any inputs fixed to specific values, or processes defined to operate in specific ways, reflect assumptions we make about the actors, environments, and their interactions in our computational process theories. When we simulate a computational process model under one set of assumptions, we can evaluate the sensitivity of the model to those assumptions by comparing the data we generate from those assumptions to the data we generate from a different set of assumptions (Smaldino, et al., 2015). This assessment helps us evaluate under what set of assumptions we might expect reasonable or unreasonable results. In turn, it helps us develop more sensible and specific interventions (Braun et al., 2022). Thus, testing the assumptions of our theories promotes their utility, and writing computational process theories allows us to evaluate our assumptions thoroughly. Ultimately, a thorough evaluation of our theoretical assumptions offers an opportunity for a more robust and open organizational science (Antonakis, 2023).

Long-Term Benefits of Computational Process Theories for Organizational Science

To conclude, we consider the long-term benefits of developing computational process theories for organizational science. We discuss benefits centered on: (1) process-focused empirical investigations, (2) embracing the heterogeneity of people, (3) incorporating temporal perspectives, (4) developing explicit, transparent, and reproducible theories, and (5) building a cumulative science. Together, we think these benefits provide a compelling vision for organizational scientists to develop computational process theories.

Process-Focused Empirical Investigations

We focused this manuscript on discussing how the development of computational process theories advances the theoretical rigor of organizational science. However, theory alone proves insufficient for developing a rich, actionable, process-oriented knowledge base in the organizational sciences. We must marry well-developed theories with (1) rigorously designed and well-executed data collection efforts and (2) effective analytical work focused on thoroughly evaluating theoretical argumentation and propositions (Shadish et al., 2001). In brief, the advent of computational process theories should coincide with well-designed, process-focused counterfactual experimentation and/or observational data to evaluate the veracity of generative mechanisms and establish empirical causal relations (Antonakis, 2017; Antonakis et al., 2010; Eden, 2017; 2021).

Fortunately, the fun, iterative interplay of developing, simulating, and evaluating computational process theories readily brings to light ways in which we may advance the empirical rigor of organizational science as well. Present data collection strategies overwhelmingly focus on measuring construct states between units and/or (less frequently) within units across time periods. These kinds of construct measurements happen post-hoc to the operation of processes. In other words, these measurements miss the process action. We note a major difference between measuring team performance from one performance episode to the next under a particular leadership style versus measuring how leaders instructed, interacted, and influenced team members, and how team members collaborated, communicated, coordinated, and performed their tasks during performance episodes. The latter reflects a process-focused empirical investigation (e.g., Cui et al., 2023; Grand et al., 2016). Furthermore, we note that merely fitting traditional statistical models (e.g., structural equations models) to construct data to estimate parameters of interest and evaluating model fit to observed construct variances/covariances does not itself provide direct support for a proposed process rationale, just as we demonstrated with the construct mediation model estimated on the simulated data from the example computational process model. A theory-based model might "fit the data (i.e., construct variances/covariances of interest) well" but not because of the theoretically specified reasons (i.e., processes). To evaluate whether proposed processes generate construct states of interest, we must measure the empirical processes in operation. Ideally, we would collect data from a series of process-focused counterfactual experimental studies with robust measurement protocols that evaluate the operations of different generative mechanisms (Antonakis, 2017; 2023). Marrying computational process theories with thorough empirical process investigations provides a powerful scientific platform for understanding process operations (Grand et al., 2016). Such a theory-data collaboration provides the opportunity to build effective (causal) process interventions (Braun et al., 2022). In turn, effective process interventions raise the strategic profile of organizational science by providing uniquely operative organizational policies. Thus, we see efforts to live in the theoretical and empirical action as an imperative for organizational scientists.

Embracing the Heterogeneity of People

Organizational science takes individuals as the foundational cornerstone of organizational functioning. As we referenced from the start, the people make the place (Kuljanin & Lemmon, 2024; Schneider, 1987). Yet, studies of organizational phenomena tend to treat people as mere carriers of constructs. In doing so, we lose sight of how people operate. If we do not know how people operate, then we stand no chance of understanding how organizations of people operate. Indeed, construct theories, measurements, and analytics typically rest on the assumption of the interchangeability of people within the population of interest, which implies that one person equals another person. In other words, we lose the heterogeneity of people under such an assumption. Perhaps the most important part of the heterogeneity of people consists of the heterogeneity of how people operate (Borsboom et al., 2003; 2004; Molenaar, 2004). The development of computational process theories that investigate and explore the heterogeneity of how people operate (as in the differential functioning of teams because of the specific operational mechanics of leadership styles) holds much promise for the advancement of organizational science. Indeed, such an embrace of the heterogeneity of people would finally help us understand how people make the place.

Incorporating Temporal Perspectives

Organizational scholars commonly document that our theories lack a clear and precise specification of time (e.g., Ancona et al., 2001; Dormann & Griffen, 2015; George & Jones, 2000; Mitchell & James, 2001). Our theories do not explicate when phenomena occur, the time scales at which phenomena evolve, nor when measurement may best capture the constructs or processes involved. Such a lack of temporal precision implies that organizational scientists can merely hope that their measurement timing aligns with the underlying phenomena, potentially resulting in wasted resources (e.g., Braun et al., 2022). Furthermore, it leaves organizational researchers susceptible to inaccurately quantifying relationships, potentially resulting in misleading conclusions and inaccurate inferences (e.g., Maxwell & Cole, 2007). Similarly, most theories and longitudinal empirical investigations only consider generic time frames (e.g., days, weeks, months) consisting of a relatively few measurement occasions (e.g., three measurements each day for two weeks) (Braun et al., 2013). Therefore, phenomena that develop more slowly, exhibit complex dynamics requiring many measurement occasions to observe, or only occur at generational or evolutionary time scales remain largely unknown.

Fortunately, developing computational process theories naturally incorporates temporal perspectives. When we simulate computational process models, it allows us to see the short- and long-run temporal implications of our theories in a manner that bypasses some of our natural cognitive limitations to do so with narrative construct theories (Cronin et al., 2009; Farrell & Lewandowsky, 2010). Current computational power enables modeling of any number of measurement points, which enables the discovery of complex dynamics in phenomena that practical constraints render infeasible in most empirical settings (Braun et al., 2022; Kozlowski et al., 2013). Moreover, most computational process theories necessitate that researchers specify the event-based timing of processes to iterate actions and update states (as demonstrated in the pseudocode in Table 3). The time scales we specify in a computational process model may vary from months to years or even generations (e.g., March, 1991; Samuelson et al., 2019; Scullen et al., 2005), which suggests that we may better understand the implications of our theories across different time horizons. Such an expanded temporal scope allows for theorizing about long-term, even evolutionary, trends of organizational phenomena in a manner far easier and more expansive than afforded by narrative construct theorizing. Indeed, examining the same phenomena across different time horizons allows for the inspection of critical phases of processes in operation, which we may utilize to identify critical measurement points and intervene as needed to ensure effective operational functioning (Braun et al., 2022). In brief, the development of computational process theories can help us rigorously incorporate temporal perspectives in our theories.

Developing Explicit, Transparent, and Reproducible Theories

Researchers widely recognize the utilities of explicit, transparent, and reproducible research methods and analytics to effectively evaluate scientific findings (Nosek et al., 2015). Yet, we can similarly define, and thereby, recognize the utilities of explicit, transparent, and reproducible theories to evaluate the plausibility of a theory even before subjecting it to new empirical evaluations (Antonakis, 2023). To define these qualities for theories, we can say an explicit theory clearly details its arguments and specifies its conditions; a transparent theory makes itself open to public scrutiny, critique, and falsification; and a reproducible theory makes it possible for anyone to derive the same conclusions from the same theoretical arguments. Increasing the number of explicit, transparent, and reproducible theories in a scientific domain facilitates the development, evaluation, and advancement of its ideas (Wulff et al., 2023).

Computational process theories neatly exhibit these three qualities. First, when we develop computational process theories, we must explicitly specify our logic to perform computations. Otherwise, a computational process theory cannot compute. Second, when we develop computational process theories, we provide any evaluator (including future us) access to our complete set of computations, which also includes giving access to the subset of computations that define generative mechanisms and the sequencing of those computations that define processes in operation. Third, when we develop computational process theories, we may reproduce our predictions, descriptions, interventions, hypotheses, etc., by simulating specified computations. Anyone simulating the same set and sequence of computations would reproduce the exact same consequences. Taken together, these qualities help us clearly communicate our theories and ideas, while limiting ambiguities in interpretation, logic, and application.

Lastly, we note that when we develop explicit, transparent, and reproducible theories, it likely improves the internal (e.g., fellow research colleagues) and external (e.g., granting agencies, media, organizational leaders) perception of the quality of research performed by individual organizational researchers and the collective of organizational scientists (Eronen & Bringmann, 2021; Nosek et al., 2015). Indeed, explicit, transparent, and reproducible organizational theories better prepare us for methodological replication and analytical reproducibility (Antonakis, 2023). The ability of a science to explain and replicate empirical findings builds its scientific reputation. With a strong scientific reputation, organizational science can see its influence consistently flourish.

Building a Cumulative Science

Present practices in organizational research proliferate individual narrative construct theories for specific empirical studies without concern for integrating those theories into overarching theories (Gigerenzer, 2010) or pruning those theories without empirical support (Ferris et al., 2012). Researchers tend to present either their own individual narrative theoretical argumentation focused on the relationships of a few constructs or use theoretical arguments made by other researchers to propose some new construct relationships (Cucina & McDaniel, 2016; Mathieu, 2016). In either case, organizational researchers tend to move from one individual theory to another individual theory as they conduct their empirical studies without sufficiently integrating or pruning the set of individual theories. The database of our journals consists of a patchwork of individual theories with little connective tissue for researchers to derive how they all fit together even though each individual theory discusses similar subject matter. In brief, the database of our journals does not reflect a cumulative science (Götz et al., 2022; Irvine, 2021; Meehl, 1978).

The typical theoretical focus on constructs in organizational science seems to lead researchers to treat each set of constructs as their own independent areas of investigation. This behavior, in turn, treats people as a collection of individual construct parts as opposed to an integrated whole of actions. In contrast, computational process theories provide the possibility to identify common generative mechanisms across a range of organizational phenomena, and thereby, create the potential for integrating disparate organizational theories. In this way, organizational science becomes a cumulative science built from an understanding of process operations. In closing, we encourage organizational researchers to embrace the study of process operations to build a cumulative science ready to embrace the full complexities of organizational systems and human work processes.

References

Adner, R., Pólos, L., Ryall, M., & Sorenson, O. (2009). The case for formal theory. *Academy of Management Review*, *34*(2), 201–208. https://doi.org/10.5465/AMR.2009.36982613

Aiken, J. R., Hanges, P. J., & Chen, T. (2019). The means are the end: Complexity science in organizational research. In S. E. Humphrey & J. M. LeBreton (Eds.), *The handbook of multilevel theory, measurement, and analysis* (pp. 115–140). Washington, DC, US: American Psychological Association. https://doi.org/10.1037/0000115-006

Ancona, D. G., Okhuysen, G. A., & Perlow, L. A. (2001). Taking time to integrate temporal research. Academy of Management Review, 26, 512-529.

Anderson, J. R. (1990). *The adaptive character of thought*. Psychology Press. https://doi.org/10.4324/9780203771730

Anghel, M., Toroczkai, Z., Bassler, K. E., & Korniss, G. (2004). Competition-driven network dynamics: Emergence of a scale-free leadership structure and collective efficiency. *Physical Review Letters*, *92*(5), 058701. https://doi.org/10.1103/PhysRevLett.92.058701

Antonakis, J. (2017). On doing better science: From thrill of discovery to policy implications. *The Leadership Quarterly*, *28*(1), 5–21. https://doi.org/10.1016/j.leaqua.2017.01.006

Antonakis, J. (2023). In support of slow science: Robust, open, and multidisciplinary. *The Leadership Quarterly*, *34*(1), 101676. https://doi.org/10.1016/j.leaqua.2023.101676

Antonakis, J., Bendahan, S., Jacquart, P., & Lalive, R. (2010). On making causal claims: A review and recommendations. *The Leadership Quarterly*, *21*(6), 1086–1120. https://doi.org/10.1016/j.leaqua.2010.10.010

Arthur, W., Jr., Bennett, W., Jr., Edens, P. S., & Bell, S. T. (2003). Effectiveness of training in organizations: A meta-analysis of design and evaluation features. *Journal of Applied Psychology*, *88*(2), 234–245. https://doi.org/10.1037/0021-9010.88.2.234

Aust, F., & Barth, M. (2022). *papaja: Prepare reproducible APA journal articles with R Markdown*. Retrieved from https://github.com/crsh/papaja

Bache, S. M., & Wickham, H. (2022). *magrittr: A forward-pipe operator for r*. Retrieved from https://CRAN.R-project.org/package=magrittr

Ballard, T., Palada, H., Griffin, M., & Neal, A. (2021). An integrated approach to testing dynamic, multilevel theory: Using computational models to connect theory, model, and data. *Organizational Research Methods*, *24*(2), 251–284. https://doi.org/10.1177/1094428119881209

Barth, M. (2022). *tinylabels: Lightweight variable labels*. Retrieved from https://cran.r-project.org/package=tinylabels

Bastardoz, N., Matthews, M. J., Sajons, G. B., Ransom, T., Kelemen, T. K., & Matthews, S. H. (2023). Instrumental variables estimation: Assumptions, pitfalls, and guidelines. *The Leadership Quarterly*, 101673. https://doi.org/10.1016/j.leaqua.2022.101673

Bengtsson, H. (2021). *A unifying framework for parallel and distributed processing in R using futures*. *The R Journal, 13*(2), 208-227.

Bezanson, J., Karpinski, S., Shah, V. B., & Edelman, A. (2012). *Julia: A fast dynamic language for technical computing* (arXiv:1209.5145). arXiv. http://arxiv.org/abs/1209.5145

Black, J. A, Oliver, R. L, Howell, J. P, & King, J. P. (2006). A dynamic system simulation of leader and group effects on context for learning. *The Leadership Quarterly*, *17*, 39-56.

Borsboom, D., Mellenbergh, G. J., & van Heerden, J. (2003). The theoretical status of latent variables. *Psychological Review, 110*(2), 203-209. https://doi.org/10.1037/0033-295X.110.2.203

Borsboom, D., Mellenbergh, G. J., van Heerden, J. (2004). The concept of validity. *Psychological Review, 111*(4), 1061-1071. https://doi.org/10.1037/0033-295X.111.4.1061

Borsboom, D., van der Maas, H. L. J., Dalege, J., Kievit, R. A., & Haig, B. D. (2021). Theory construction methodology: A practical framework for building theories in psychology. *Perspectives on Psychological Science*, *16*(4), 756–766. https://doi.org/10.1177/1745691620969647

Brass, D. J., & Borgatti, S. P. (2019). Multilevel thoughts on social networks. In *The handbook of multilevel theory, measurement, and analysis* (pp. 187–200). Washington, DC, US: American Psychological Association. https://doi.org/10.1037/0000115-009

Braun, M. T., Kuljanin, G., & DeShon, R. P. (2013). Spurious results in the analysis of longitudinal data in organizational research. *Organizational Research Methods*, *16*(2), 302–330. https://doi.org/10.1177/1094428112469668

Braun, M. T., Kuljanin, G., Grand, J. A., Kozlowski, S. W. J., & Chao, G. T. (2022). The power of process theories to better understand and detect consequences of organizational interventions. *Industrial and Organizational Psychology*, *15*(1), 99–104. https://doi.org/10.1017/iop.2021.125

Broomell, S. B., & Davis-Stober, C. P. (2023). The Strengths and Weaknesses of Crowds to Address Global Problems. *Perspectives on Psychological Science: A Journal of the Association for Psychological Science*, 17456916231179152. https://doi.org/10.1177/17456916231179152

Carley, K. M. (2002). Computational organizational science and organizational engineering. *Simulation Modelling Practice and Theory*, *10*(5), 253–269.

Carley, K. M., & Prietula, M. J. (1994). *Computational organization theory.* Lawrence Erlbaum Associates.

Carter, D. R., DeChurch, L. A., Braun, M. T., & Contractor, N. S. (2015). Social network approaches to leadership: An integrative conceptual review. *Journal of Applied Psychology*, *100*(3), 597–622. https://doi.org/10.1037/a0038922

Cheng, J., Sievert, C., Schloerke, B. Chang, W., Xie, Y., & Allen, J. (2023). *htmltools: Tools for HTML*. Retrieved from https://CRAN.R-project.org/package=htmltools

Coveney, P. V., Dougherty, E. R., & Highfield, R. R. (2016). Big data need big theory too. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *374*(2080), 20160153. https://doi.org/10.1098/rsta.2016.0153

Cronin, M. A., Gonzalez, C., & Sterman, J. D. (2009). Why don't well-educated adults understand accumulation? A challenge to researchers, educators, and citizens. *Organizational Behavior and Human Decision Processes*, *108*(1), 116–130. https://doi.org/10.1016/j.obhdp.2008.03.003

Cronin, M. A., Weingart, L. R., & Todorova, G. (2011). Dynamics in groups: Are we there yet? *The Academy of Management Annals*, *5*(1), 571–612. https://doi.org/10.1080/19416520.2011.590297

Csaszar, F. A. (2020). Certum quod factum: How formal models contribute to the theoretical and empirical robustness of organization theory. *Journal of Management*, *46*(7), 1289–1301. https://doi.org/10.1177/0149206319889129

Cucina, J. M., & McDaniel, M. A. (2016). Pseudotheory proliferation is damaging the organizational sciences. *Journal of Organizational Behavior*, *37*(8), 1116–1125. https://doi.org/10.1002/job.2117

Cui, J., Hasselman, F., & Lichtwarck-Aschoff, A. (2023). Unlocking nonlinear dynamics and multistability from intensive longitudinal data: A novel method. *Psychological Methods*. https://doi.org/10.1037/met0000623

Davis, J. P., Eisenhardt, K. M., & Bingham, C. B. (2007). Developing theory through simulation methods. *Academy of Management Review*, *32*(2), 480–499. https://doi.org/10.5465/AMR.2007.24351453

DeRue, D. S., Barnes, C., & Morgeson, F. (2010). Understanding the motivational contingencies of team leadership. *Small Group Research*, *41*, 621–651. https://doi.org/10.1177/1046496410373808

Dionne, S. D, & Dionne, P. J. (2008). Levels-based leadership and hierarchical group decision optimization: A simulation. *The Leadership Quarterly, 19*, 212-234.

Dionne, S. D., Sayama, H., Hao, C., & Bush, B. J. (2010). The role of leadership in shared mental model convergence and team performance improvement: An agent-based computational model. *The Leadership Quarterly*, *21*(6), 1035–1049. https://doi.org/10.1016/j.leaqua.2010.10.007

Dong, J., Liu, R., Qiu, Y, & Crossan, M. (2021). Should knowledge be distorted? Managers' knowledge distortion strategies and organizational learning in different environments. *The Leadership Quarterly, 32*, 101477.

Dormann, C., & Griffin, M. A. (2015). Optimal time lags in panel studies. *Psychological Methods*, *20*(4), 489–505. https://doi.org/10.1037/met0000041

Eden, D. (2017). Field experiments in organizations. *Annual Review of Organizational Psychology and Organizational Behavior*, *4*(1), 91–122. https://doi.org/10.1146/annurev-orgpsych-041015-062400

Eden, D. (2021). The science of leadership: A journey from survey research to field experimentation. *The Leadership Quarterly*, *32*(3), 101472. https://doi.org/10.1016/j.leaqua.2020.101472

Edwards, J. R., & Berry, J. W. (2010). The presence of something or the absence of nothing: Increasing theoretical precision in management research. *Organizational Research Methods*, *13*(4), 668–689. https://doi.org/10.1177/1094428110380467

Epstein, J. M. (1999). Agent-based computational models and generative social science. *Complexity*, *4*(5), 41–60. https://doi.org/10.1002/(SICI)1099-0526(199905/06)4:5<41::AID-CPLX9>3.0.CO;2-F

Eronen, M. I., & Bringmann, L. F. (2021). The theory crisis in psychology: How to move forward. *Perspectives on Psychological Science*, *16*(4), 779–788. https://doi.org/10.1177/1745691620970586

Farrell, S., & Lewandowsky, S. (2010). Computational models as aids to better reasoning in psychology. *Current Directions in Psychological Science*, *19*(5), 329–335. https://doi.org/10.1177/0963721410386677

Ferris, G. R., Hochwarter, W. A., & Buckley, M. R. (2012). Theory in the organizational sciences: How will we know it when we see it? *Organizational Psychology Review*, *2*(1), 94–106. https://doi.org/10.1177/2041386611423696

Fischer, T., Dietz, J., & Antonakis, J. (2017). Leadership process models: A review and synthesis. *Journal of Management*, *43*(6), 1726–1753. https://doi.org/10.1177/0149206316682830

Gell-Mann, M. (1994). Complex adaptive systems. In G. Cowan, D. Pines, & D. Meltzer (Eds.), *Complexity: Metaphors, models, and reality* (pp. 17-45). Reading, MA: Addison-Wesley.

George, J. M., & Jones, G. R. (2000). The role of time in theory and theory building. *Journal of Management, 26*, 657-684.

Gigerenzer, G. (2010). Personal reflections on theory and psychology. *Theory & Psychology*, *20*(6), 733–743. https://doi.org/10.1177/0959354310378184

Gohel, D. (2023). *officer: Manipulation of Microsoft Word and PowerPoint documents*. Retrieved from https://CRAN.R-project.org/package=officer

Gohel, D., & Skintzos, P. (2023). *flextable: Functions for tabular reporting*. Retrieved from https://CRAN.R-project.org/package=flextable

Götz, F. M., Gosling, S. D., & Rentfrow, P. J. (2022). Small effects: The indispensable foundation for a cumulative psychological science. *Perspectives on Psychological Science*, *17*(1), 205–215. https://doi.org/10.1177/1745691620984483

Grabner, R. H., Stern, E., & Neubauer, A. C. (2007). Individual differences in chess expertise: A psychometric investigation. *Acta Psychologica*, *124*(3), 398–420. https://doi.org/10.1016/j.actpsy.2006.07.008

Grand, J. A., Braun, M. T., & Kuljanin, G. (in press). Hello world! Building computational models to represent social and organizational theory. *Organizational Research Methods*.

Grand, J. A., Braun, M. T., Kuljanin, G., Kozlowski, S. W. J., & Chao, G. T. (2016). The dynamics of team cognition: A process-oriented theory of knowledge emergence in teams. *Journal of Applied Psychology*, *101*(10), 1353–1385. https://doi.org/10.1037/apl0000136

Guest, O., & Martin, A. E. (2021). How computational modeling can force theory building in psychological science. *Perspectives on Psychological Science*, 789–802. https://doi.org/10.1177/1745691620970585

Harrison, J. R., Lin, Z., Carroll, G. R., & Carley, K. M. (2007). Simulation modeling in organizational and management research. *The Academy of Management Review*, *32*(4), 1229–1245. https://doi.org/10.2307/20159364

Hazy, J. K. (2007). Computer models of leadership: Foundations for a new discipline or meaningless diversion? *The Leadership Quarterly*, *18*(4), 391–410. https://doi.org/10.1016/j.leaqua.2007.04.007

Hazy, J. K. (2008). Toward a theory of leadership in complex systems: Computational modeling explorations. *Nonlinear Dynamics, Psychology, and Life Sciences*, *12*(3), 281–310.

Hazy, J. K., & Uhl-Bien, M. (2015). Towards operationalizing complexity leadership: How generative, administrative, and community-building leadership practices enact organizational outcomes. *Leadership*, *11*(1), 79–104. https://doi.org/10.1177/1742715013511483

Henry, L., & Wickham, H. (2023). *rlang: Functions for base types and core R and 'tidyverse' features*. Retrieved from https://CRAN.R-project.org/package=rlang

Hester, J., Wickham, H., & Csárdi, G. (2023). *fs: Cross-platform file system operations based on 'libuv'*. Retrieved from https://CRAN.R-project.org/package=fs

Hunter, J. E., & Hunter, R. F. (1984). Validity and utility of alternative predictors of job performance. *Psychological Bulletin*, *96*(1), 72–98. https://doi.org/10.1037/0033-2909.96.1.72

Hyman, H. (1955). *Survey design and analysis: Principles, cases, and procedures*. Glencoe, IL: The Free Press.

Ilgen, D. R., Hollenbeck, J. R., Johnson, M., & Jundt, D. (2005). Teams in organizations: From input-process-output models to IMOI models. *Annual Review of Psychology*, *56*, 517–543. https://doi.org/10.1146/annurev.psych.56.091103.070250

Ilgen, D. R., & Hulin, C. L. (2000). *Computational modeling of behavior in organizations: The third scientific discipline*. Washington, DC, US: American Psychological Association. https://doi.org/10.1037/10375-000

Irvine, E. (2021). The role of replication studies in theory building. *Perspectives on Psychological Science*, *16*(4), 844–853. https://doi.org/10.1177/1745691620970558

Izrailev, S. (2023). *tictoc: Functions for timing R scripts, as well as implementation of "stack" and "stacklist" structures*. Retrieved from https://CRAN.R-project.org/package=tictoc

Jacobsen, C., & House, R. J. (2001). Dynamics of charismatic leadership: A process theory, simulation model, and tests. *The Leadership Quarterly, 12,* 75-112.

Katz, D., & Kahn, R. L. (1966). *The social psychology of organizations*. Wiley.

Kellen, D., Davis-Stober, C. P., Dunn, J. C., & Kalish, M. L. (2021). The problem of coordination and the pursuit of structural constraints in psychology. *Perspectives on Psychological Science*, *16*(4), 767–778. https://doi.org/10.1177/1745691620974771

Klein, K. J., Ziegert, J. C., Knight, A. P., & Xiao, Y. (2006). Dynamic delegation: Shared, hierarchical, and deindividualized leadership in extreme action teams. *Administrative Science Quarterly*, *51*(4), 590–621. https://doi.org/10.2189/asqu.51.4.590

Kline, R. B. (2015). The mediation myth. *Basic and Applied Social Psychology*, *37*(4), 202–213. https://doi.org/10.1080/01973533.2015.1049349

Kozlowski, S. W. J. (2015). Advancing research on team process dynamics: Theoretical, methodological, and measurement considerations. *Organizational Psychology Review*, *5*(4), 270–299. https://doi.org/10.1177/2041386614533586

Kozlowski, S. W. J. (2022). The data revolution and the interplay between theory and data. In K. R. Murphy, *Data, Methods and Theory in the Organizational Sciences* (1st ed., pp. 206–240). New York: Routledge. https://doi.org/10.4324/9781003015000-12

Kozlowski, S. W. J., & Chao, G. T. (2012). The dynamics of emergence: Cognition and cohesion in work teams. *Managerial & Decision Economics*, *33*(5–6), 335–354. https://doi.org/10.1002/mde.2552

Kozlowski, S. W. J., Chao, G. T., Grand, J. A., Braun, M. T., & Kuljanin, G. (2013). Advancing multilevel research design: Capturing the dynamics of emergence. *Organizational Research Methods*, *16*(4), 581–615. https://doi.org/10.1177/1094428113493119

Kozlowski, S. W. J., Chao, G. T., Grand, J. A., Braun, M. T., & Kuljanin, G. (2016). Capturing the multilevel dynamics of emergence: Computational modeling, simulation, and virtual experimentation. *Organizational Psychology Review*, *6*(1), 3–33. https://doi.org/10.1177/2041386614547955

Kozlowski, S. W. J., Gully, S. M., Nason, E. R., & Smith, E. M. (1999). Developing adaptive teams: A theory of compilation and performance across levels and time. In D. R. Ilgen & E. D. Pulakos (Eds.), *The changing nature of work performance: Implications for staffing, personnel actions, and development* (pp. 240-292). San Francisco: Jossey-Bass.

Kozlowski, S. W. J., & Klein, K. J. (2000). A multilevel approach to theory and research in organizations: Contextual, temporal, and emergent processes. In K. J. Klein & S. W. J. Kozlowski (Eds.), *Multilevel theory, research and methods in organizations: Foundations, extensions, and new directions* (pp. 3-90). Jossey-Bass.

Kuljanin, G. (2011). *A computational study of team collaboration and performance* [Doctoral dissertation, Michigan State University]. MSU Libraries Digital Collections. https://doi.org/10.25335/M53M87

Kuljanin, G., Braun, M. T., Grand, J. A., Kozlowski, S. W. J., & Chao, G. T. (2021, April). Local versus global optimization in multi-team systems. In S. W. J. Kozlowski, G. T. Chao, & G. A. Ruark (Chairs), *Dissecting the dynamics of team, multiteam, and organizational systems*. Symposium presented at the 36th Annual Conference of the Society for Industrial and Organizational Psychology, New Orleans, LA.

Kuljanin, G., & Lemmon, G. (2024). The role of work psychologists in the development of antiwork sentiments. *Industrial and Organizational Psychology: Perspectives on Science and Practice*, *17*(1), 45–49.

Ladha, K. K. (1992). The Condorcet jury theorem, free speech, and correlated votes. *American Journal of Political Science*, *36*(3), 617–634. https://doi.org/10.2307/2111584

LePine, J. A., Piccolo, R. F., Jackson, C. L., Mathieu, J. E., & Saul, J. R. (2008). A meta-analysis of teamwork processes: Tests of a multidimensional model and relationships with team effectiveness criteria. *Personnel Psychology*, *61*(2), 273–307. https://doi.org/10.1111/j.1744-6570.2008.00114.x

Levitt, R. E. (2012). The Virtual Design Team: Designing project organizations as engineers design bridges. *Journal of Organization Design*, *1*(2), 14-41. https://doi.org/10.7146/jod.6345

Lewin, K., & Lippitt, R. (1938). An experimental approach to the study of autocracy and democracy: A preliminary note. *Sociometry*, *1*(3/4), 292–300. https://doi.org/10.2307/2785585

Lewin, K., Lippitt, R., & White, R. A. (1939). Patterns of aggressive behavior in experimentally created "social climates." *The Journal of Social Psychology*, *10*(2), 271–299.

Likert, R. (1961). *New patterns of management* (pp. ix, 279). McGraw-Hill.

Liu, C., Eubanks, D. L., & Chater, N. (2015). The weakness of strong ties: Sampling bias, social ties, and nepotism in family business succession. *The Leadership Quarterly, 26*(3), 419-435.

Locke, E. A., & Latham, G. P. (1990). *A theory of goal setting and task performance*. Englewood Cliffs, NJ: Prentice-Hall, Inc.

London, M., & Beatty, R. W. (1993). 360-degree feedback as a competitive advantage. *Human Resource Management*, *32*(2–3), 353–372. https://doi.org/10.1002/hrm.3930320211

Lungeanu, A., DeChurch, L. A., & Contractor, N. S. (2022). Leading teams over time through space: Computational experiments on leadership network archetypes. *The Leadership Quarterly*, *33*(5), 101595. https://doi.org/10.1016/j.leaqua.2021.101595

Macal, C. M., & North, M. J. (2010). Tutorial on agent-based modelling and simulation. *Journal of Simulation*, *4*(3), 151–162. https://doi.org/10.1057/jos.2010.3

Macy, M. W., & Willer, R. (2002). From factors to actors: Computational sociology and agent-based modeling. *Annual Review of Sociology*, *28*(1), 143–166. https://doi.org/10.1146/annurev.soc.28.110601.141117

March, J. G. (1991). Exploration and exploitation in organizational learning. *Organization Science*, *2*(1), 71–87. Retrieved from http://www.jstor.org/stable/2634940

Marion, R., & Uhl-Bien, M. (2001). Leadership in complex organizations. *The Leadership Quarterly*, *12*, 389–418. https://doi.org/10.1016/S1048-9843(01)00092-3

Marks, M. A., Mathieu, J. E., & Zaccaro, S. J. (2001). A temporally based framework and taxonomy of team processes. *The Academy of Management Review*, *26*(3), 356–376. https://doi.org/10.2307/259182

Maruping, L. M., Venkatesh, V., Thatcher, S. M. B., & Patel, P. C. (2015). Folding under pressure or rising to the occasion? Perceived time pressure and the moderating role of team temporal leadership. *Academy of Management Journal*, *58*(5), 1313–1333. https://doi.org/10.5465/amj.2012.0468

Mathieu, J. E. (2016). The problem with [in] management theory. *Journal of Organizational Behavior*, *37*(8), 1132–1141. https://doi.org/10.1002/job.2114

Maxwell, S. E., & Cole, D. A. (2007). Bias in cross-sectional analysis of longitudinal mediation. *Psychological Methods*, 12(1), 23–44.

McDermott, R. (2023). On the scientific study of small samples: Challenges confronting quantitative and qualitative methodologies. *The Leadership Quarterly*, 101675. https://doi.org/10.1016/j.leaqua.2023.101675

McHugh, K. A., Yammarino, F. J., Dionne, S. D., Serban, A., Sayama, H., & Chatterjee, S. (2016). Collective decision making, leadership, and collective intelligence: Tests with agent-based simulations and a Field study. *The Leadership Quarterly*, *27*(2), 218–241. https://doi.org/10.1016/j.leaqua.2016.01.001

Meehl, P. E. (1967). Theory-testing in psychology and physics: A methodological paradox. *Philosophy of Science*, *34*(2), 103–115. Retrieved from http://www.jstor.org/stable/186099

Meehl, P. E. (1978). Theoretical risks and tabular asterisks: Sir Karl, Sir Ronald, and the slow progress of soft psychology. *Journal of Consulting and Clinical Psychology*, *46*(4), 806–834.

Mitchell, T. R., & James, L. R. (2001). Building better theory: Time and the specification of when things happen. *Academy of Management Review, 26*, 530-547.

Mohr, L. B. (1982). *Explaining organizational behavior: The limits and possibilities of theory and research*. San Francisco: Proquest Info & Learning.

Molenaar, P. C. M. (2004). A manifesto on psychology as idiographic science: Bringing the person back into scientific psychology, this time forever. *Measurement: Interdisciplinary Research & Perspective*, *2*(4), 201–218. https://doi.org/10.1207/s15366359mea0204_1

Müller, K., & Wickham, H. (2023). *Tibble: Simple data frames*. Retrieved from https://CRAN.R-project.org/package=tibble

Nicolis, G., & Prigogine, I. (1989). *Exploring complexity: An introduction*. New York: W. H. Freeman.

Niederman, F., & March, S. T. (2018). An exposition of process theory and critique of Mohr's (1982) conceptualization thereof. *Philosophy of Management*, *17*(3), 321–331. https://doi.org/10.1007/s40926-017-0082-x

Nosek, B. A., Alter, G., Banks, G. C., Borsboom, D., Bowman, S. D., Breckler, S. J., … Yarkoni, T. (2015). Promoting an open research culture. *Science (New York, N.Y.)*, *348*(6242), 1422–1425. https://doi.org/10.1126/science.aab2374

Olenick, J., & Dishop, C. (2022). Clarifying dynamics for organizational research and interventions: A diversity example. *Organizational Psychology Review*, *12*(4), 365–386. https://doi.org/10.1177/20413866221112427

Page, S. E. (2018). *The model thinker: What you need to know to make data work for you* (1st edition). Basic Books.

Park, S., Sturman, M. C., Vanderpool, C., & Chan, E. (2015). Only time will tell: The changing relationships between LMX, job performance, and justice. *Journal of Applied Psychology*, *100*(3), 660–680. https://doi.org/10.1037/a0038907

Paustian-Underdahl, S. C., Walker, L. S., & Woehr, D. J. (2014). Gender and perceptions of leadership effectiveness: A meta-analysis of contextual moderators. *Journal of Applied Psychology*, *99*(6), 1129–1145. https://doi.org/10.1037/a0036751

Pentland, B. T. (1999). Building process theory with narrative: From description to explanation. *Academy of Management Review*, *24*(4), 711–724. Retrieved from http://www.proquest.com/docview/210974423/abstract/357D3047A69549BCPQ/1

Popper, K. R. (2002). *The logic of scientific discovery*. London; New York: Routledge.

Posit Team. (2023). *RStudio: Integrated Development Environment for R* [Computer software]. Boston, MA: Posit Software, PBC. https://posit.co/

Prietula, M. J., Carley, K. M., & Gasser, L. (1998). *Simulating organizations*. Menlo Park, CA: AAAI Press.

R Core Team. (2023). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Rand, W., & Rust, R. T. (2011). Agent-based modeling in marketing: Guidelines for rigor. *International Journal of Research in Marketing*, *28*(3), 181–193. https://doi.org/10.1016/j.ijresmar.2011.04.002

Richard, B. W., Holton III, E. F., & Katsioloudes, V. (2014). The use of discrete computer simulation modeling to estimate return on leadership development investment. *The Leadership Quarterly, 25*, 1054-1068.

Roethlisberger, F. J., & Dickson, W. J. (1939). *Management and the worker* (pp. xxiv, 615). Harvard Univ. Press.

Ronay, R., Oostrom, J. K., Lehmann-Willenbrock, N., Mayoral, S., & Rusch, H. (2019). Playing the trump card: Why we select overconfident leaders and why it matters. *The Leadership Quarterly, 30*(6), 1-19.

Rosseel, Y. (2012). lavaan: An R package for structural equation modeling. *Journal of Statistical Software*, *48*(2), 1-36. http://www.jstatsoft.org/v48/i02/

Rosenhead, J., Franco, L. A., Grint, K., & Friedland, B. (2019). Complexity theory and leadership practice: A review, a critique, and some recommendations. *The Leadership Quarterly*, *30*(5), 101304. https://doi.org/10.1016/j.leaqua.2019.07.002

Samuelson, H. L., Levine, B. R., Barth, S. E., Wessel, J. L., & Grand, J. A. (2019). Exploring women's leadership labyrinth: Effects of hiring and developmental opportunities on gender stratification. *The Leadership Quarterly*, *30*(6), 101314. https://doi.org/10.1016/j.leaqua.2019.101314

Sayama, H. (2015). *Introduction to the modeling and analysis of complex systems*. Open SUNY Textbooks.

Schelling, T. C. (1971). Dynamic models of segregation. *The Journal of Mathematical Sociology*, *1*(2), 143–186. https://doi.org/10.1080/0022250X.1971.9989794

Schneider, B. (1987). The people make the place. *Personnel Psychology*, *40*(3), 437–453. https://doi.org/10.1111/j.1744-6570.1987.tb00609.x

Scullen, S. E., Bergey, P. K., & Aiman-Smith, L. (2005). Forced distribution rating systems and the improvement of workforce potential: A baseline simulation. *Personnel Psychology*, *58*(1), 1–32. https://doi.org/10.1111/j.1744-6570.2005.00361.x

Serban, A., Yammarino, F. J., Dionne, S. D., Kahai, S. S., Hao, C., McHugh, K. A., Lee Sotak, K., Mushore, A. B. R., Friedrich, T. L., & Peterson, D. R. (2015). Leadership emergence in face-to-face and virtual teams: A multi-level model with agent-based simulations, quasi-experimental and experimental tests. *The Leadership Quarterly, 26,* 402-418.

Shadish, W. R., Cook, T. D., & Campbell, D. T. (2001). *Experimental and quasi-experimental designs for generalized causal inference* (2nd edition). Belmont, CA: Cengage Learning.

Simon, H. A. (1957). *Models of man; social and rational*. Oxford, England: Wiley.

Simon, H. A. (1992). What is an "explanation" of behavior? *Psychological Science, 3*(3), 150-161. https://doi.org/10.1111/j.1467-9280.1992.tb00017.x

Smaldino, P. E., Calanchini, J., & Pickett, C. L. (2015). Theory development with agent-based models. *Organizational Psychology Review*, *5*(4), 300–317. https://doi.org/10.1177/2041386614546944

Smith, E. R. (2012). Editorial. *Journal of Personality and Social Psychology*, *102*(1), 1–3. https://doi.org/10.1037/a0026676

Smith, E. B., & Rand, W. (2018). Simulating macro-level effects from micro-level observations. *Management Science*, *64*(11), 5405–5421. https://doi.org/10.1287/mnsc.2017.2877

Strauss, J. A., & Grand, J. A. (2022). Applying systems science to advance research in team phenomena. In B. Murray, J. Dulebohn, & D. Stone (Eds.), *Managing team centricity in modern organizations* (pp. 17-52). IAP.

Sutton, R. I., & Staw, B. M. (1995). What theory is not. *Administrative Science Quarterly*, *40*(3), 371. https://doi.org/10.2307/2393788

Sveidqvist, K., & Jain, A. (2021). *The official guide to Mermaid.js: Create complex diagrams and beautiful flowcharts easily using text and code*. Packt Publishing.

Tuncdogan, A., Acar, O. A., & Stam, D. (2017). Individual differences as antecedents of leader behavior: Towards an understanding of multi-level outcomes. *The Leadership Quarterly*, *28*(1), 40–64. https://doi.org/10.1016/j.leaqua.2016.10.011

Van Der Vegt, G., Emans, B., & Van De Vliert, E. (1998). Motivating effects of task and outcome interdependence in work teams. *Group & Organization Management*, *23*(2), 124–143. https://doi.org/10.1177/1059601198232003

Van Rossum, G., & Drake, F. L. (2009). *Python 3 reference manual*. Scotts Valley, CA: CreateSpace.

Vancouver, J. B., Wang, M., & Li, X. (2020). Translating informal theories into formal theories: The case of the dynamic computational model of the integrated model of work motivation. *Organizational Research Methods*, *23*(2), 238–274. https://doi.org/10.1177/1094428118780308

Vaughan, D., & Dancho, M. (2022). *furrr: Applyl mapping functions in parallel using futures.* Retrieved from https://CRAN.R-project.org/package=furrr

Watts, L. L., Gray, B. E., & Medeiros, K. E. (2022). Side effects associated with organizational interventions: A perspective. *Industrial and Organizational Psychology*, *15*(1), 76–94. https://doi.org/10.1017/iop.2021.93

Wickham, H. (2023). *forcats: Tools for working with categorical variables (factors)*. Retrieved from https://CRAN.R-project.org/package=forcats

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. New York: Springer-Verlag. Retrieved from https://ggplot2.tidyverse.org

Wickham, H., François, R., Henry, L., Müller, K., & Vaughan, D. (2023). *dplyr: A grammar of data manipulation*. Retrieved from https://CRAN.R-project.org/package=dplyr

Wickham, H., & Henry, L. (2023). *purrr: Functional programming tools*. Retrieved from https://CRAN.R-project.org/package=purrr

Wickham, H., & Seidel, D. (2022). *scales: Scale functions for visualization.* Retrieved from https://CRAN.R-project.org/package=scales

Wilensky, U. (1999). *NetLogo*. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. Retrieved from http://ccl.northwestern.edu/netlogo/

Will, T. E. (2016). Flock leadership: Understanding and influencing emergent collective behavior. *The Leadership Quarterly, 27*, 261-279.

Wright, S. (1920). The relative importance of heredity and environment in determining the piebald pattern of guinea-pigs. *Proceedings of the National Academy of Sciences*, *6*(6), 320–332.

Wulff, J. N., Sajons, G. B., Pogrebna, G., Lonati, S., Bastardoz, N., Banks, G. C., & Antonakis, J. (2023). Common methodological mistakes. *The Leadership Quarterly*, 101677. https://doi.org/10.1016/j.leaqua.2023.101677

Xie, Y. (2015). *Dynamic documents with R and knitr* (2nd ed.). Boca Raton, Florida: Chapman; Hall/CRC. Retrieved from https://yihui.org/knitr/

Xie, Y., Allaire, J. J., & Grolemund, G. (2018). *R markdown: The definitive guide*. Boca Raton, Florida: Chapman; Hall/CRC. Retrieved from https://bookdown.org/yihui/rmarkdown

Xie, Y., Cheng, J., & Tan, X. (2023). *DT: A wrapper of the JavaScript library 'DataTables'*. Retrieved from https://CRAN.R-project.org/package=DT

Xie, Y., Dervieux, C., & Riederer, E. (2020). *R markdown cookbook*. Boca Raton, Florida: Chapman; Hall/CRC. Retrieved from https://bookdown.org/yihui/rmarkdown-cookbook

Appendix A

Concise Review of Computational Modeling of Organizational Phenomena

To highlight the historical advocacy for using computational modeling to advance organizational theory development, Table A1 briefly summarizes selected works to provide context and background, highlight a variety of exemplars, and serve as a point of entry for interested scholars. Previous work includes pioneering efforts to advance computational modeling for theory development in the organizational sciences (e.g., Carley & Prietula, 1994; Harrison et al., 2007; March, 1991), exemplar demonstrations of building and evaluating computational models against empirical data (e.g., Ballard et al., 2021; Grand et al., 2016), and a listing of computational modeling and research published in *The Leadership Quarterly* (e.g., Dionne et al., 2010; McHugh et al., 2016; Samuelson et al., 2019). Although some organizational scholars advocated for the application of computational modeling to develop organizational theory and conduct empirical research for over 30 years, computational modeling failed to gain widespread traction in the organizational sciences. Nonetheless, there exists renewed interest in computational modeling for developing theory, demonstrating the plausibility of theories, and coupling it with thorough empirical evaluations for validation and evidence for application. This previous work lays the foundation for advancing the use of computational modeling to develop process theories of organizational phenomena.

Table A1

Concise Review of Computational Modeling of Organizational Phenomena

Reference	Description
Pioneering Theorizing With Computational Modeling	
March (1991)	Developed a computational model to examine exploration and exploitation in the context of organizational socialization and learning.
Carley & Prietula (1994)	Edited book with chapters describing applications of computational modeling to organizational design and theory using agents, tasks, and networks.

Reference	Description
Prietula et al. (1998)	Edited book with chapters describing applications of computational modeling to team and organizational behavior.
Epstein (1999)	Compared computational modeling with inductive and deductive scientific approaches to theory building and described advantages of CM for the study of emergence and dynamics.
Ilgen & Hulin (2000)	Edited book with chapters describing the application of computational modeling to individual, group/team, and organizational behavior.
Carley (2002)	Described different aspects of computational modeling linked it to evaluations with empirical data.
Harrison et al. (2007)	Described different types of computational modeling and encouraged applications for studying organizational phenomena.
Hazy (2008)	Described a new leadership theory based on computational modeling approaches.
Kozlowski et al. (2013)	Described the advantages of computational modeling for theorizing about processes and emergent phenomena in organizations with examples applied to team effectiveness.
Exemplar Empirical Evaluations of Computational Modeling	
Grand et al. (2016)	Demonstrated a four-step approach via an example: (1) developed a narrative process theory of team knowledge emergence, (2) translated the narrative process theory into a formal computational process model, (3) instantiated the computational process model into an agent-based simulation and explored the effects of generative mechanisms of learning and information sharing on team knowledge emergence, and (4) conducted an empirical experiment to evaluate whether proposed generative mechanisms of learning and information sharing directly improved team knowledge emergence in human teams.
Ballard et al. (2021)	Demonstrated a four-step approach via an example: (1) developed a primary computational model of goal-striving and demonstrated model sufficiency, (2) evaluated the primary computational model against empirical data, (3) compared the primary model to an alternative model, and (4) interpreted estimated values of primary model parameters.
Computational Modeling Publications in The Leadership Quarterly	
Jacobsen & House (2001)	Developed a system dynamics model of charismatic leadership and tested model with six historical charismatic leaders.
Black et al. (2006)	Described agent-based simulations of various leader-group developmental paths that support organizational learning.

Reference	Description
Dionne & Dionne (2008)	Developed a computational model to compare effects of different types of leadership (autocratic, individualized, participative, leader-member exchange) on decision optimization.
Dionne et al. (2010)	Developed a computational model based on McComb's theory of phases of mental model development (orientation, differentiation, integration) to compare participative leadership with leader-member exchange.
Richard et al. (2014)	Developed discrete-event computer simulations used to estimate the return on leadership development investments.
Serban et al. (2015)	Developed agent-based simulations used to examine how leaders emerge in face-to-face and virtual teams.
Liu et al. (2015)	Used computational modeling to show how nepotism might explain a leader's sampling bias and strong family ties.
McHugh et al. (2016)	Used agent-based simulations and field data to examine relationships between individual intelligence and knowledge, collective intelligence, and collective decision quality.
Will (2016)	Developed agent-based model to examine how technical capacity and adaptive capacity affect emergent collective behavior.
Ronay et al. (2019)	Developed agent-based simulations to examine how political candidates' overconfidence evolved and affected election turnout and election results.
Samuelson et al. (2019)	Developed agent-based simulations to examine how external hiring and developmental opportunities may present barriers to women seeking leadership positions.
Dong et al. (2021)	Developed agent-based simulations to examine environmental conditions that relate to positive or negative effects of managerial knowledge distortion strategies on organizational learning.
Lungeanu et al. (2022)	Developed agent-based simulations with the aid of empirical data to examine effects of leadership structures on team mental models.

Appendix B

Detailed Pseudocode for the Example Computational Process Model

We provide a detailed pseudocode for the example computational process model we discuss in the manuscript. Specifically, we provide the details of each computational step described at a high-level in Table 3. Readers may find the implementation of the complete executable computational code in the supplemental *R Markdown* script. As we discuss different parameters and states, we reference them by the names we assigned them in the executable computational code.

Step 1

The first step consists of defining several static inputs of the computational process model. We define the number of: (1) leaders in an organization, (2) team members per leader, (3) tasks team members may perform, (4) actions team members may execute per task, and (5) performance episodes that actors perform; we define (6) the maximum possible reward team members may receive when performing a task via any action; we define boundary values for process parameters related to actor (7) social influence, (8) expertise, and (9) perceptual accuracies; we define (10) a process parameter for team member work priorities. We may alter these inputs across simulations to evaluate their impact on what emerges, but we fix them to specific values to generate results from the example model that highlight the process operations of autocratic and delegative leadership. In Table B1, we define the specific values we set for the results presented in this manuscript.

Step 2

The second step consists of defining initial values for the dynamic states of actors and the static maximum reward for each work task. This step differs as a function of the two leadership styles. For autocratic leadership, we first establish a connection between leaders and team members in an organization by linking identifiers. Given 100 leaders, we create identifiers from one to 100 for leaders; given 20 members per leader, we create identifiers from one to 20 for team members. The combination of a pair of leader and team member identifiers establishes a

Table B1

Static Inputs for the Example Computational Process Model

Parameter	Definition	Example Value
n_leaders	Number of leaders in an organization	100
n_members	Number of team members per leader	20
n_tasks	Number of possible tasks to perform during each performance episode	20
n_actions	Number of task action strategies available to perform any task	4
n_episodes	Number of performance episodes	30
task_max_value_all	Maximum possible reward available for performing any task	4
infl_wgt	Weight applied to change social influence	0.05
infl_change	Possible fluctuation applied to change social influence	0.03
infl_imp	Importance of social influence in updating work priorities	0.80
le_min_change	Minimum value in the fluctuation applied to change leader expertise	0.00
le_max_change	Maximum value in the fluctuation applied to change leader expertise	0.05
me_min_change	Minimum value in the fluctuation applied to change team member expertise	0.00
me_max_change	Maximum value in the fluctuation applied to change team member expertise	0.05
mp_change	Possible fluctuation applied to change team member work priorities	0.03

Parameter	Definition	Example Value
perc_min_change	Minimum value in the fluctuation applied to change actor perceptual accuracy	0.00
perc_max_change	Maximum value in the fluctuation applied to change actor perceptual accuracy	0.05
perc_max_bound	Maximum possible value of actor perceptual accuracy	0.80

leader-member dyad. Second, leaders and team members both possess expertise to perform their work. We generate initial expertise values for each actor by making independent draws from the standard normal distribution (i.e., mean value set to zero, and standard deviation value set to one). We use normal distributions given their representativeness of individual difference states (Grabner et al., 2007). The independent draws imply that we treat each actor independent of one another (i.e., the expertise value of one actor does not influence the expertise value of another actor). Third, team members hold work priorities for performing work in their focal teams. We generate initial work priorities for each team member by making independent draws from the continuous standard uniform distribution (i.e., minimum value set to zero, and maximum value set to one). Using the continuous standard uniform distribution allows us to consider team member work priorities as probabilities for team members performing their assigned tasks. Fourth, as performance on tasks unfolds, leaders evaluate team members to some degree of accuracy and influence them socially to some degree as well. We assume leaders do not know anything about team members before their work begins, and thus, we initialize each leader-member perceptual accuracy and social influence state to zero. Fifth, we establish a connection between teams and work tasks by linking identifiers. Given 20 tasks, we create identifiers from one to 20 for work tasks. The combination of a pair of leader and work task identifiers establishes a team-task link. Sixth, we generate maximum reward values for each task team members may perform. We generate these values using a discrete probability distribution defined by the maximum possible reward value across tasks (i.e., *task_max_value_all*), and we assume all tasks contribute some discrete, positive reward to project accomplishment up to the maximum reward value across tasks. To do so, we define the probability of drawing a particular reward value for a task as the inverse of the reward value relative to the sum of all possible task reward values (i.e., reverse normalization). Defining the probabilities in such a way implies we sample relatively higher rewarded tasks with relatively lower probability. As an example, for a maximum reward value of four across all tasks, tasks may receive reward values from one to four. In this case, the probability of drawing a task reward value of one, two, three, or four equals 0.4, 0.3, 0.2, and 0.1, respectively.

For delegative leadership, we initialize dynamic states in the same ways as in the autocratic leadership case with two exceptions. First, given team members enact all processes under the delegative leadership style, then team members, instead of leaders, hold perceptual accuracies and social influence. Second, given that delegative leadership requires team members to enact all processes, then we must establish a connection between all team members on the same team. We establish a member-member dyad in each team by referencing the identifiers for team members. Otherwise, we formulate the same computations. In Table B2, we provide additional details on the initialization of dynamic states.

Step 3

The third step consists of ranking tasks for each team from most to least valued. We take the maximum task reward values calculated in Step 2 and sort them from highest to lowest. Then, we assign each task an integer rank from one to the number of tasks (i.e., *n_tasks*) whereby one represents the most valued task. Several tasks may hold the exact same maximum task reward value. In those cases, we still assign ranks in ascending integer order as it does not make a difference for subsequent processes.

Step 4

The fourth step begins to define processes in operation during each performance episode. In the fourth step, we define how assessors evaluate the expertise of team members to perform

Table B2

Dynamic Actor States and Task Rewards for the Example Computational Process Model

State	Definition	Note
leader_expertise	A leader's expertise to enact work processes	Sampled from the standard normal distribution: $N(0, 1)$
member_expertise	A team member's expertise to enact work processes	Sampled from the standard normal distribution: $N(0, 1)$
member_priority	A team member's work priority for performing tasks of the focal team	Sampled from the continuous standard uniform distribution: $U(0, 1)$
leader_percept_acc	A leader's perceptual accuracy of a team member's expertise	Set to zero initially
leader_influence	A leader's social influence over a team member	Set to zero initially
member_percept_acc	A team member's perceptual accuracy of a teammate's expertise	Set to zero initially
member_influence	A team member's social influence over a teammate	Set to zero initially
task_max_value	Maximum possible reward for performing a task	Sampled from a discrete probability distribution as described in the text

Note. N = normal distribution; U = uniform distribution.

tasks. We explicate this step in the second paragraph of the *A Guide for Developing Computational Process Theories of Organizational Phenomena – Process Operations – Mechanisms* section of the manuscript. We refer readers to that paragraph.

Steps 5 and 6

The fifth and sixth steps involve actors assigning team members to perform tasks. Specifically, the fifth step consists of rank-ordering team members based on perceived expertise; the sixth step consists of matching the rank-orders of team members and tasks. We explicate these steps in the third paragraph of the *A Guide for Developing Computational Process Theories of Organizational Phenomena – Process Operations – Mechanisms* section of our manuscript. We refer readers to that paragraph.

Step 7

The seventh step involves actors determining performance action strategies for tasks. The exact process for determining performance action strategies depends on the leadership style. For autocratic leadership, leaders use their own expertise of tasks to choose performance action strategies team members should execute for their assigned tasks, while, for delegative leadership, team members use their own expertise to choose performance action strategies (Lewin et al., 1939; Lewin & Lippitt, 1938). Each performance action strategy for a given task provides a different reward, which implies we can rank order the effectiveness of available performance action strategies. Leader expertise for tasks determines their ability to choose the best action strategy for any task. Specifically, we first calculate cumulative probabilities using the standard logistic distribution function, $1/(1 + e^{-x})$, in which *x* equals a sequence of values, *thresholds – AE*. The *thresholds* parameter represents a sequence of values to indicate at which point agents take the next best action; the sequence starts at *1 – (n_actions / 2),* ends at *(n_actions / 2) – 1*, and increments by one; the *n_actions* parameter represents the number of available performance actions while satisfying the constraint of exceeding or equaling two. The *AE* parameter represents an actor's (dependent on the leadership style) expertise. We then take the difference between cumulative probabilities to represent non-cumulative probabilities of selecting a particular action. As an example, for four possible actions and an actor's expertise of 0.5, we first calculate three cumulative (with respect to the lower tail) probabilities of 0.182, 0.377, and 0.622, per the standard logistic distribution function as we defined it; we then take the difference between these cumulative probabilities including the probabilities of zero and one to compute four non-cumulative probabilities of 0.182, 0.195, 0.245, and 0.378, which represents the probability of choosing each of the four actions, respectively, from lowest to largest performance reward. Based on these probabilities, actors make a performance action choice for team members to execute their assigned tasks (*task_action*).

Step 8

The eighth step involves team members deciding if they will perform their assigned tasks (*perform_choice: 1 = yes, perform assigned task; 0 = no, do not perform assigned task*). In such a way, we consider that team members hold other work obligations outside of their teams in consideration (e.g., individual job responsibilities in their organizations; work for other teams in their organizations), and hence, they do not automatically perform their assigned tasks for their teams in a specific performance episode. For team members to make their performance choice, we compare their probabilistic work priorities (*member_priority*) to a random draw from a continuous standard uniform distribution. If team member work priorities exceed the value of the random draw, then they do perform their assigned tasks. Otherwise, team members do not perform their assigned tasks.

Step 9

The ninth step involves team members performing their assigned tasks. For team members who do choose to perform their assigned tasks, then they do so via the chosen performance action strategy from the prior step. By performing their assigned task, they accumulate a performance reward proportional to the quality of their performance action strategy. For team members who do not choose to perform their assigned tasks, they do not accumulate any performance rewards for their teams. We calculate performance rewards via the following equation:

$$\textit{perform\_reward} = \textit{perform\_choice} * \textit{task\_max\_value} * (\textit{task\_action} / \textit{n\_actions}). \tag{B1}$$

In Equation B1, *perform_reward* represents the performance reward team members receive from executing their assigned tasks, *perform_choice* represents whether team members performed their assigned tasks, *task_max_value* represents the maximum possible reward for assigned tasks, and *task_action* represents the action strategy selected for performing tasks with values ranging from one (worst action strategy) to *n_actions* (best action strategy). We record the performance choices and rewards for each episode, which we analyze after execution of simulations to understand the performance consequences of the two leadership styles.

Step 10

The tenth step consists of updating states for the subsequent performance episode, which depends on the two leadership styles. In Table B3, we provide the computational details on the dynamic updates to actor states. For autocratic leadership, we update five states: team member expertise, leader social influence, team member work priorities, leader perceptual accuracies, and leader expertise. First, team members update their expertise to perform tasks by a small positive increment, drawn from a continuous uniform distribution (i.e., from *me_min_change* to *me_max_change*), if they performed their assigned task (i.e., team members develop their expertise by performing tasks). Otherwise, their expertise to perform tasks remains the same. Second, leader influence over team members updates as a function of (1) a small fraction (i.e., *infl_wgt*) of each team member's performance reward relative to the maximum reward, which reflects effectiveness of performance, plus (2) a small negative or positive fluctuation, drawn from a continuous uniform distribution (i.e., from negative to positive *infl_change*), that reflects a random change in influence. Third, team members update their priorities to work on tasks as a function of (1) a small negative or positive fluctuation, drawn from a continuous uniform distribution (i.e., from negative to positive *mp_change*), that reflects updated work demands, plus (2) a fraction of the updated social influence from leaders. Given that we conceptualize both leader social influence and team member work priorities as probabilities ranging from zero to one, we bind those values to zero or one when calculations breach those boundaries. Fourth, leader perceptual accuracies of team member expertise change by a small positive (due to observational experience) fluctuation, drawn from a continuous uniform distribution (i.e., from *perc_min_change* to *perc_max_change*), if team members performed their assigned tasks (i.e., leaders improve their assessments as they observe team members perform). Otherwise, for team members who did not perform their assigned tasks, leader perceptual accuracies of team member expertise remain the same. Given that we conceptualize leader perceptual accuracies as a

Table B3

Dynamic Updates to Actor States in the Example Computational Process Model

State	Computation
	if (perform_reward > 0) {
	member_expertise$_{(pe+1)}$ = member_expertise$_{(pe)}$ +
member_expertise	*U*(me_min_change, me_max_change)
(autocratic, delegative)	} else {
	member_expertise$_{(pe+1)}$ = member_expertise$_{(pe)}$
	}
	leader_influence$_{(pe+1)}$ = leader_influence$_{(pe)}$ +
	infl_wgt * (perform_reward / task_max_value) +
	U(-infl_change, infl_change)
leader_influence	
(autocratic);	if (leader_influence$_{(pe+1)}$ < 0) {
member_influence	leader_influence$_{(pe+1)}$ = 0
(delegative)	} else if (leader_influence$_{(pe+1)}$ > 1) {
	leader_influence$_{(pe+1)}$ = 1
	} else {
	leader_influence$_{(pe+1)}$ = leader_influence$_{(pe+1)}$
	}
	member_priority$_{(pe+1)}$ = member_priority$_{(pe)}$ +
	infl_imp * leader_influence$_{(pe+1)}$ +
	U(-mp_change, mp_change)
	if (member_priority$_{(pe+1)}$ < 0) {
member_priority	member_priority$_{(pe+1)}$ = 0
(autocratic, delegative)	} else if (member_priority$_{(pe+1)}$ > 1) {
	member_priority$_{(pe+1)}$ = 1
	} else {
	member_priority$_{(pe+1)}$ = member_priority$_{(pe+1)}$
	}
leader_percept_acc	if (perform_reward > 0) {
(autocratic);	leader_percept_acc$_{(pe+1)}$ = leader_percept_acc$_{(pe)}$ +

State	Computation

member_percept_acc
(delegative)
	U(perc_min_change, perc_max_change)
	} else {
	  leader_percept_acc$_{(pe + 1)}$ = leader_percept_acc$_{(pe)}$
	}
	
	if (leader_percept_acc$_{(pe + 1)}$ > perc_max_bound) {
	  leader_influence$_{(pe + 1)}$ = perc_max_bound
	} else {
	  leader_percept_acc$_{(pe + 1)}$ = leader_percept_acc$_{(pe + 1)}$
	}

leader_expertise (autocratic)	leader_expertise$_{(pe + 1)}$ = leader_expertise$_{(pe)}$ + sum(perform_choice)$_{(members)}$ / n_members + *U*(le_min_change, le_max_change) sum(perform_choice)$_{(members)}$: sum of perform_choice across team members

Note. U = uniform distribution; pe = performance episode; information in the parentheses of the first column indicates the names we assigned to states in the computational script for each leadership style; when we assigned different names to states, then the second column provides the computational details for just the autocratic leadership case for a more concise presentation.

validity, we bind it to a maximum value (e.g., *perc_max_bound*). Fifth, leaders update their expertise as a function of (1) a small positive change, drawn from a continuous uniform distribution (e.g., from *le_min_change* to *le_max_change*), that reflects a general gain in decision-making experience multiplied by (2) the proportion of team members who performed their assigned tasks that reflects specific decision-making experience about team members.

For delegative leadership, we update four states all belonging to team members: social influence, work priorities, expertise, and perceptual accuracies. We update these variables in the same way as we do for autocratic leadership. As a result, we only show the autocratic computations in Table B3.

Step 11

The eleventh step consists of creating an iterative loop for processes to continue until the last performance episode. Once teams complete the last performance episode, they complete their work. At that point, we can evaluate all the actions of actors and the emergence of various state (e.g., performance, work priorities, expertise, social influence) trajectories.