Journal of Management Information Systems

ISSN: 0742-1222 (Print) 1557-928X (Online) Journal homepage: http://www.tandfonline.com/loi/mmis20

Understanding the Drivers of Unethical Programming Behavior: The Inappropriate Reuse of Internet-Accessible Code

Manuel Sojer, Oliver Alexy, Sven Kleinknecht & Joachim Henkel

To cite this article: Manuel Sojer, Oliver Alexy, Sven Kleinknecht & Joachim Henkel (2014) Understanding the Drivers of Unethical Programming Behavior: The Inappropriate Reuse of Internet-Accessible Code, Journal of Management Information Systems, 31:3, 287-325, DOI: 10.1080/07421222.2014.995563

To link to this article: http://dx.doi.org/10.1080/07421222.2014.995563

Published online: 09 Mar 2015.

Understanding the Drivers of Unethical Programming Behavior: The Inappropriate Reuse of Internet-Accessible Code

MANUEL SOJER, OLIVER ALEXY, SVEN KLEINKNECHT, AND JOACHIM HENKEL

MANUEL SOJER researches technology and innovation management at TUM School of Management, Technische Universität München, Germany, where he also received his Ph. D. He is also the head of strategy and marketing of the business unit power quality at Maschinenfabrik Reinhausen, a global supplier to the transformer industry. Previously, he worked at the consulting firm Bain & Company for five years. In the management domain, his research interests center on profiting from innovation, open source software, and knowledge reuse. In the field of power engineering, he studies smart grid technologies and the grid integration of renewable energy. His work has been published in the *Journal of the Association for Information Systems, Communications of the ACM,* and the proceedings of multiple international management and power engineering conferences.

OLIVER ALEXY is professor of strategic entrepreneurship at TUM School of Management, Technische Universität München, Germany, where he was also awarded his Ph.D. Previously, he worked as a postdoctoral researcher and assistant professor at Imperial College London. His current research focuses on organizational design, interorganizational collaboration, and open, distributed models of innovation in innovation ecosystems. His work has been published in the *Academy of Management Review, Research Policy, Sloan Management Review,* and other international journals. He serves on the editorial review boards of the *Academy of Management Journal, Journal of Management Studies,* and *IEEE Transactions in Engineering Management.*

SVEN KLEINKNECHT is a researcher affiliated with EBS Universität, Wiesbaden, Germany and TUM School of Management, Technische Universität München, Germany. His research interests include entrepreneurial behavior and ethical decision making. After graduating in business administration (majoring in information systems), he worked for PricewaterhouseCoopers. He has professional experience in evaluating and designing management controls in IT systems and is a certified information systems auditor.

JOACHIM HENKEL is a professor of technology and innovation management at TUM School of Management, Technische Universität München, Germany. His research focuses on open source software, user innovation, patent infringements, and profiting from innovation, and has been published in, among other journals, *Harvard Business Review, Rand Journal of Economics, Research Policy, and Strategic Management Journal*. He serves on the editorial review boards of *Academy of Management Journal, Industrial and Corporate Change, and Research Policy.* He

Journal of Management Information Systems / Winter 2014, Vol. 31, No. 3, pp. 287–325.

ISSN 0742–1222 (print) / ISSN 1557–928X (online)
DOI: 10.1080/07421222.2014.995563

has been a visiting scholar at University College London, MIT Sloan School of Management, and twice at Harvard Business School. He received a degree in physics from the University of Bonn, a Ph.D. in economics from the University of Mannheim, and was an assistant professor (*Habilitand*) at Ludwig-Maximilians-Universität München. After gaining his Ph.D., he worked for two years with the consulting firm Bain & Company.

ABSTRACT: Programming is riddled with ethical issues. Although extant literature explains why individuals in IT would act unethically in many situations, we know surprisingly little about what causes them to do so during the creative act of programming. To address this issue, we look at the reuse of Internet-accessible code: software source code legally available for gratis download from the Internet. Specifically, we scrutinize the reasons why individuals would unethically reuse such code by not checking or purposefully violating its accompanying license obligations, thus risking harm for their employer. By integrating teleological and deontological ethical judgments into a theory of planned behavior model—using elements of expected utility, deterrence, and ethical work climate theory—we construct an original theoretical framework to capture individuals' decision-making process leading to the unethical reuse of Internet-accessible code. We test this framework with a unique survey of 869 professional software developers. Our findings advance the theoretical and practical understanding of ethical behavior in information systems. We show that programmers use consequentialist ethical judgments when carrying out creative tasks and that ethical work climates influence programmers indirectly through their peers' judgment of what is appropriate behavior. For practice, where code reuse promises substantial efficiency and quality gains, our results highlight that firms can prevent unethical code reuse by informing developers of its negative consequences, building a work climate that fosters compliance with laws and professional codes, and making sure that excessive time pressure is avoided.

KEY WORDS AND PHRASES: code reuse, ethical behavior, information systems ethics, Internet-accessible code, open source software, partial least squares, programming ethics, theory of planned behavior.

ETHICAL BEHAVIOR IS CENTRAL TO MANY ASPECTS OF INFORMATION SYSTEMS (IS), and the fast-paced evolvement of IS leads to the continuous emergence of new ethical issues [53, 69]. As a result, the IS context provides a variety of settings in which individuals can and have to decide what is right and what is wrong, with little controlling influence other than their own conscience [40, pp. 183ff.]. In particular in their role as firm employees, it is important that individuals display ethical behavior and do not place their own personal interest over the welfare of their colleagues or employer [e.g., 5, 10, 29, 72]. Notably, these concerns transcend mere philosophical considerations: in day-to-day business, appropriate individual-level behavior is core to vital company-level issues such as information security [14] and product quality [10].

Accordingly, a broad stream of research has tried to analyze why company employees would choose to act unethically, focusing on issues such as software

piracy at work (e.g., [54]), IS abuse (e.g., [31]), intellectual property and privacy issues (e.g., [70]), and data or identity theft (e.g., [8]). Also, extant research has looked at unethical behavior with regard to the output of software development activity, such as knowingly delivering substandard products or exaggerating their efficacy (e.g., [10]). In short, a plethora of work exists trying to explain why individuals at work would break rules, misuse their given surroundings, or willfully consider harming customers with outputs they have produced. At the same time, work that looks at unethical behavior in the actual activity of programming software irrespective of the output to be produced is scarce [10].

This is surprising since, as part of the programming activity, employees continually need to make choices in which they have to weigh their own interests against those of the corporation. Specifically, programming differs from other IS activities in that designing and implementing software is an inherently creative endeavor and as such is not conducive to control. Knuth [39] has likened programming to a creative activity such as "composing poetry or music."[1] It is in the nature of programming that each new problem necessitates the development of novel code.

In this context, two issues are pivotal. First, prior research has established that a controlling management style reduces creativity [65]—programmers need autonomy to be able to produce novel and useful software. However, experimental research has shown how people carrying out creative tasks are more likely to behave dishonestly [27]. Second, software development is inherently complex, rendering the control of development practices almost impossible. Rather, it is individuals' choice to behave ethically and act in a way that prevents harm to their employer (see, e.g., [10, 72]). Yet, combined with high degrees of autonomy, the complexity of software projects creates a situation in which programmers may also justify or hide unethical behavior.

From a theoretical perspective, this argument implies that programming substantially differs from other areas of IS regarding the role of ethical decision making—current knowledge about individual-level ethical decision making cannot simply be "copied" to this setting. Accordingly, in this paper, we strive to address the question, What drives unethical programming behavior by individuals? Notably, answering this question matters not only for theory but also for practice, because managers need to gain insight into whether and how they may intervene to break the link between creativity and dishonesty, while keeping the link to high-quality outputs intact.

To tackle this issue, we build on the rich tradition of ethical studies in the IS domain. We construct an original framework joining different theoretical streams in a research model built on the theory of planned behavior [1]. Similar to Bulgurcu et al. [14], we capture teleological perspectives (i.e., rule-based ethical consideration of actions) by drawing on expected utility theory (e.g., [23, 63]) and deterrence theory [21]. Adding a deontological perspective (i.e., basing ethical judgments on actions' consequences) through ethical work climate theory (e.g., [80]), our framework allows us to study individuals' attitudes, perceived social norms, behavioral control, and intention toward unethical programming behavior.

We apply this framework to a particularly salient situation in which developers may choose to take "unethical shortcuts": the reuse of software code. The recent

emergence of code available as a gratis (free) download from the Internet, particularly open source software (OSS; e.g., [30]), has further increased potential benefits of code reuse [e.g., 25]. Notably, such "Internet-accessible code" (hereafter simply IAC) is often reused by individual professional software developers in ad hoc fashion when these developers—on their own and typically without telling anyone—employ it as a shortcut in their work (e.g., [52]). Levi and Woodard [44, p. 8] even claimed that such ad hoc reuse of IAC has become "standard practice for many programmers."

However, IAC usually has a license attached to it, which may put legal limits on how the software can be used. Because appropriately taking license obligations into account may be an annoying burden for developers, they may sometimes not thoroughly check for the obligations that come with reuse, or even intentionally ignore such obligations [67]. As a result, although IAC reuse may solve programming problems, such reuse without regard for license obligations may entail negative long-term consequences for firms that may far outweigh any individual-level and short-term firm-level benefits.[2]

To operationalize our research question, we look at factors that make software developers more or less likely to disregard potential license obligations when reusing IAC. Following a prestudy consisting of thirty-two interviews with experts on this topic, we survey 869 professional software developers using a vignette study approach [26] that contains multiple scenarios of ad hoc IAC reuse. Simultaneously, we solicit individual software developers' opinions and motivations while including potential controls for problems of common methods bias [58].

Our results underline the importance of teleological and deontological ethical judgments in unethical code reuse. We find that both judgments affect intentions. From a deontological perspective, elements of the ethical work climate touching on larger professional or legal codes affect individuals' intention toward the unethical reuse of IAC indirectly, through subjective norms. From a teleological view, benefit considerations derived from expected utility theory and deterrence theory predict attitude, with deterrence showing higher relative coefficient values and levels of significance, and significant differences between the scenarios we develop.

Taken together, the elements of our research model allow us to capture individual-level decision-making processes behind the unethical reuse of IAC. In doing so, we make three contributions to the IS literature. First, we look at how individuals arrive at an evaluation of unethical behavior. We note that in this context, contrary to adoption decisions (e.g., [78]) or ethical decisions in less creative contexts (e.g., [56]), cost considerations have stronger effects than benefit considerations. We also find that individuals' rationale differs between different types of reuse, raising a note of caution regarding the comparability of ethics studies applying diverse scenarios. Second, we extend work studying the impact of ethical work climates and codes on individual behavior (e.g., [48, 75, 77]) by highlighting the prevalent importance of institutions on individual behavior. At the same time, we explain past mixed results [31, 37, 57] by explicating the indirect nature of this effect. Finally, we extend work on IS ethics by elaborating on earlier studies with more restricted theoretical breadth

(e.g., [31]) and point out the pivotal importance of studying issues and accounts of unethical programming behavior, such as reuse of IAC or of knowledge more generally.

Theory and Hypotheses

Internet-Accessible Code Reuse as an Ethical Issue of Employee's Programming Activities

Since Mason's [49] seminal contribution in which he established the manifold ethical issues in the IS space, a stream of scholarly work has emerged to investigate and explain unethical behavior in the IS context. A clear majority of these studies focus on questions of ethics with regard to the acquisition of software, the use and abuse of existing software programs and IS infrastructures, the production of software that may incorporate morally questionable features, or the provision of software tools that do not meet promised or agreed-upon requirements (e.g., [8, 53, 54, 56, 57]). In many of these studies, the subject at focus is the individual employee, with researchers trying to understand what motivates unethical behavior in these contexts.

Although these efforts are certainly helpful in elucidating ethical issues in software development, they do not look explicitly at actual coding activities. Yet, this is a crucial area in which individuals continuously need to weigh their own interest against that of their employer and their peers [10]. In their daily work, programmers often have the opportunity to take shortcuts that are to their own benefit but may end up harming others, such as their employer, their peers, or customers. For example, individuals knowing the criteria against which their code will be checked may reduce their effort levels regarding other elements of their program, trade in quality against speed of completion [10], cut corners during testing, and thus ship potentially unreliable software [5]. More generally, research has shown how individuals in creative contexts can *and will* behave more dishonestly than in other settings [27]. It is thus questionable whether the insights on ethical decision making in IS that we have gained from other, less creative settings such as software piracy or illegal downloading may also apply to programming.

To shed light on this issue, we look at one specific element of the programming process. We argue that the reuse of existing software code is an issue in software development that is particularly fraught with ethical difficulties. Generally, reuse of existing software code is crucial for firms to increase the effectiveness, efficiency, and quality of their software development, improve time-to-market, and reduce development and maintenance costs (e.g., [38, 42, 45]). In particular, the increasingly large body of IAC available for download at no cost may be leveraged in commercial software development. When such IAC is proven and thoroughly tested, its reuse helps developers to produce artifacts of higher quality and better maintainability, often in much less time, resulting in considerable benefits to firms (e.g., [25]).

However, despite typically being available gratis on the Internet, most IAC is not in the public domain. By putting code under an OSS or some other license, the author waives part of the rights granted under copyright, while binding the receiver of the code to the terms of the license. As such, reuse of IAC is still governed by copyright, and therefore reusing such code requires adherence to its license terms and any obligations these may contain (e.g., [61]). Noncompliance with these rules is not an option, because their enforcement can have serious economic and reputational consequences (e.g., [52, 61]).

As with other forms of unethical programming behavior (e.g., [10]), the literature only acknowledges the existence of unethical IAC reuse by individuals; it does not explain the internal and external drivers of such behavior. Clearly, when contemplating the (potentially unethical) reuse of IAC, programmers are faced with a decision that includes reconciliation of competing interests, and different ways of thinking about the ethical aspects of this decision may result in divergent evaluations. Although unlicensed use of IAC is clearly unethical from a rule-based, deontological perspective, a teleological view rests on weighing the different consequences of usage. IAC usage has positive effects on some stakeholders—on the company, for example, by reducing effort, and on the users, by allowing faster completion and less resource expenditure. Should these positive consequences outweigh potential negative effects, the net positive effects would excuse a deontologically unethical IAC reuse. In turn, programmers have to weigh not only the consequences of their decision but also potentially competing evaluations arising from judgments based on consequentialist and deontologist moral theories.

Modeling Individual-Level Unethical Reuse Behavior

We construct an original theoretical framework that aims at predicting individuals' intention to engage in the unethical reuse of IAC. To anchor our design, we model the individual's decision according to the theory of planned behavior (TPB) [1], one of the most frequently employed theoretical models to investigate (un)ethical behavior in business contexts. Specifically, ethical decision making and its modeling using the TPB has a long tradition in the IS literature (e.g., [8]), including decision contexts such as software copying (e.g., [56]) or compliance in IT (e.g., [14]). The TPB has also been used extensively in ethical contexts beyond IS, such as metal finishing [24] and accounting [13]. An alternative model of ethical decision making has been presented by Hunt and Vittel [34], who also explicitly included a construct, "ethical judgment," influenced by teleological and deontological considerations. However, even though the TPB may not capture ethical judgments directly, it does so indirectly through attitude and subjective norms. Deontological aspects are incorporated into attitude and subjective norms, with personal values influencing attitude, and rules of behavior based on general societal or issue-specific beliefs shaping subjective norms. Attitude captures the ethical evaluation of the behavior (ethical appraisal of consequences and personal-values-based assessment of the action) weighed against the resulting personal gains or losses resulting, thus also

measuring teleological aspects [82]. We chose to use the TPB because its applicability has been verified by prior studies in comparable scenarios and empirically. These studies also found the TPB to exhibit higher explanatory power and model fit [82] than the Hunt-Vittel model. Also, from an ethical perspective, by using a TPB-based model to combine theories that explain ethical decision making, researchers may design original frameworks that remain representative of larger theories of moral development, thinking, and action [40, 54].

We model the individual-level ethical decision-making process for IAC reuse, including the weighing of different ethical and personal considerations. Within a TPB-based framework, attitudes capture the teleological or consequentialist aspects of ethical decision making, in contrast to deontological characteristics, which mainly influence social norms. This framework (see Figure 1) features three aspects that render it clearly distinct from extant research.

First, we draw on ethical work climate theory [80] to provide a deontological perspective on subjective norms. We tie into the debate on the effect of ethical codes on behavior, which has yielded mixed results [31, 37, 57]. We maintain that these mixed results may be explained by inconsistencies in how elements fostering an ethical work climate are modeled and present arguments that extend current thinking of a direct effect of ethical work climates on intention to an indirect effect via perceived subjective norms. Second, to introduce a teleological logic, we analyze the consequences of unethical IAC reuse. To do so, we build on Bulgurcu et al.'s [14] work on information security noncompliance to identify

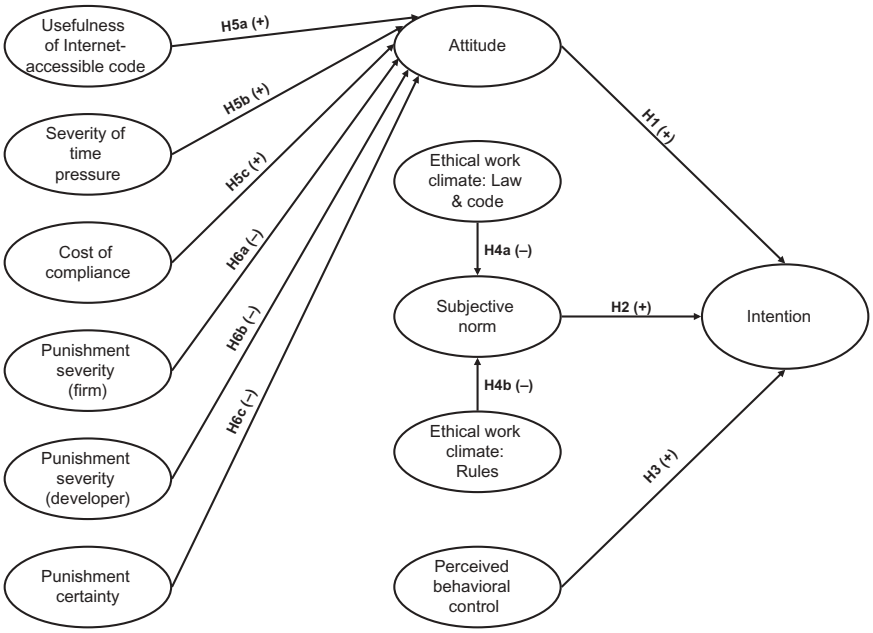

Figure 1. Research Model

relevant consequences of unethical behavior.[3] Following these authors, we rely on expected utility theory and deterrence theory to integrate both positive and negative effects of information security activities. We know from past IS work applying deterrence theory that the fear of punishment stipulates a more negative attitude toward unethical behavior [15, 56]. At the same time, individuals weigh this potential downside against the benefits they hope to achieve, as predicted by expected utility theory [21]. We compare the relative effect of perceived advantages and disadvantages, an approach common to "positive" adoption decision models such as the technology acceptance model (e.g., [78]). Third, the joint consideration of the foregoing theoretical perspectives is novel. Specifically, our simultaneous considerations of other determinants of unethical behavior should allow us to more precisely establish the effect of ethics codes and other elements of the ethical work climate to provide a more elaborate understanding of what drives (un)ethical behavior in IS.

Predicting Intention

The TPB postulates that behavioral intent is driven by individual attitude toward a behavior, perception of peers' opinion about the behavior (termed "subjective norm"), and perceived control over the behavior, which captures both the ability to engage in a behavior as well as the degree of control over its actual enactment. The relative importance of constructs influencing intention is expected to vary with regard to the behavior at stake [1, 9]. As explained earlier, a wide array of literature in various domains has shown the validity of the TPB in ethical decision making. Nonetheless, as a baseline for our research model, we contextualize the three baseline relationships posited by the TPB.

First, we expect that programmer attitude, that is, the degree to which a programmer evaluates the reuse of IAC as favorable or unfavorable, influences the intention to engage in this behavior. Past studies based on the TPB have shown attitude to be the best predictor of intention [1]. This includes research on IS ethics on issues such as software and digital media piracy (e.g., [20, 56]), and also on related topics like information security compliance [14]. Accordingly, we expect that individuals' attitude toward unethical IAC reuse will also drive their intent to do so. Second, we include subjective norms representing the perceived social pressure to engage or not to engage in a certain behavior [1]. Because it is likely that peers' perceived levels of approval influence individuals' decision making regarding unethical IAC reuse, we expect subjective norms to have an impact on intentions [20]: in line with studies on software piracy [56], we expect higher (lower) levels of perceived subjective norm to positively (negatively) affect individuals' intent to reuse IAC unethically. Third, perceived behavioral control reflects the ease of reusing IAC, as perceived by the programmer, including past experiences and anticipated impediments [1]. This includes the perception of control over the reuse situation, such as the ability to copy IAC and to reuse it [20, 72]. A review of the literature on IS ethics

and related topics [14, 56] shows that perceived behavioral control drives behavioral intent in most settings. Accordingly, for IAC reuse as well, we also expect that the absence of actual and perceived obstacles should result in increased IAC reuse.

Hypothesis 1: Developers' attitude toward unethical IAC reuse positively affects their intention to engage in such behavior.

Hypothesis 2: Developers' subjective norm concerning unethical IAC reuse positively affects their intention to engage in such behavior.

Hypothesis 3: Developers' perceived behavioral control regarding unethical IAC reuse positively affects their intention to engage in such behavior.

Predicting Subjective Norm: The Role of an Ethical Work Climate

When ethical decisions are at stake, the wider context in which these are embedded also influences decision makers [75, 81]. Organizational ethics research has found support for the relationship between the ethical climate within a firm and employees' ethical behavior (e.g., [77]). Victor and Cullen [80, p. 101] defined ethical work climate as "the prevailing perceptions of typical organizational practices and procedures that have ethical content." Although ethical work climate is a macro-level construct [81] strongly influenced by written codes of ethics [57], its perception by individuals influences "the types of ethical conflicts considered, the process by which such conflicts are resolved, and the characteristics of their resolution" [79, p. 55].

Two aspects of an ethical work climate are particularly appropriate for inclusion in our research model and allow us to follow Harrington [31] in distinguishing whether its influence on ethical behavior originates from outside or from within a focal organization. First, the *law and code* dimension reflects the importance of compliance with laws and professional codes of conduct originating from *outside the firm.* It is thus evidence of the strength of intrafirm adherence to institutional norms e.g., [55]. Given that obligations from IAC reuse result from legal instruments such as copyright, and that respect for intellectual property is part of IS codes of conduct (e.g., [5]), developers in firms in which such institutional norms and regulations are held in higher regard should have a lower intention to reuse IAC unethically. Second, the *rules* dimension reflects firm-specific, *internal* policies, procedures, and norms [81]. Opposite to the law and code dimension, these aspects of an ethical work climate may be less institutionalized as they either cannot be or are unlikely to be communicated explicitly during a developer's initial socialization, such as during university training (e.g., [35, 55]). Nonetheless, given that some firms have rules of how to deal with IAC (including reuse), socialization may happen once developers join these firms. In turn, developers in firms in which compliance with internal rules is more pronounced should have a lower intention to engage in unethical IAC reuse. Tetlock [71, p. 298], however, proposed that "both individuals and small groups of individuals are constrained by the norms, procedures, and resources of the institutions in which they live and work," suggesting that ethical work climates will not

influence individuals' behavioral intention directly, but rather the subjective norm individuals perceive. Simply put, ethical work climates may guide individuals in developing their evaluation of what others think is acceptable behavior. Thus, we posit an indirect effect of ethical work climates on intention, via subjective norm:

Hypothesis 4a: A firm's ethical work climate perceived to emphasize compliance with laws and codes negatively affects developers' level of subjective norms supporting unethical IAC reuse.

Hypothesis 4b: A firm's ethical work climate perceived to emphasize compliance with firm rules negatively affects developers' level of subjective norms supporting unethical IAC reuse.

Predicting Attitude: Cost-Benefit Estimations and the Role of Punishment

From a strictly consequentialist perspective, developers should favor the unethical reuse of IAC if they consider its net benefit to be greater than that of any alternative, such as "correct" reuse or writing the code themselves. This is in line with rationality-based assessments of alternative courses of action. For example, in their study on IS compliance, Bulgurcu et al. [14] showed that a cost-benefit analysis is an immediate antecedent of attitude. Notably, however, compliance differs from ethical decision making in that its rules are based on company objectives and as such are disconnected from independent individual-level moral considerations. Thus, ethical decision making and compliance are mainly similar with respect to specific outcomes, namely that violating a company rule—irrespective of how it came into being—is unethical behavior.

Like Bulgurcu et al. [14], we employ expected utility theory [23, 63] to account for perceived benefits and employ deterrence theory [21, 69] to include countervailing cost factors. However, given that we expect the behavior we study to theoretically and empirically vary from their phenomenon of interest, we differ in the choice of aspects used to model individuals' cost-benefit considerations in order to reflect the specific circumstances of software development. Specifically, we include the general usefulness of IAC, its specific usefulness in mitigating time pressure, and the avoidance of compliance cost as benefits. In turn, costs are modeled by punishment certainty and severity on a personal and company level. We derive these antecedents based on an extensive review of literature and have corroborated our choice in a series of thirty-two semistructured interviews with development professionals, for example, developers active on the web-based open source development platform SourceForge.net.

Usefulness of Internet-Accessible Code

IAC reuse can ease developers' jobs because it allows them to solve technical problems they could not solve themselves, or to develop (better) software in less

time (e.g., [25]). Perceptions of usefulness differ between programmers and depend on programming style and language, complexity of the problem at hand, and prior experience with IAC, such as previous involvement in OSS development. In this context, Sojer and Henkel showed [66] that anticipated individual-level benefits of code reuse are the most potent indicators of actual reuse behavior in OSS projects. The higher professional developers value the usefulness of IAC in general, the more they should also be inclined to make use of this potential shortcut for their work. In turn, we argue that the higher the value individuals should generate from taking such a shortcut, the more inclined they should be to disregard potentially associated ethical concerns (see also [14, 56]):

Hypothesis 5a: The perceived usefulness of IAC positively affects developers' attitude toward unethical IAC reuse.

Mitigation of Time Pressure

Efficiency gains through IAC reuse may in particular mean faster completion of a project and the meeting of deadlines. Generally, research on software quality shows that developers under time pressure tend to take shortcuts in order to avoid missing deadlines (e.g., [12]). Such shortcuts are "decisions made in private that are motivated by a desire to stay on schedule, but are not in the best interests of the project" [7, p. 195]. It seems likely that unethical IAC reuse is one such shortcut that developers employ and then "hope for the best [and] leave potential sources of difficulty unexplored" [7, p. 195]. The willingness to consider using such an enticing shortcut can be exacerbated by changes in staffing or shifts in responsibilities after reaching project goals. In a theoretical model, Austin [7] showed that developers who perceive more severe consequences from missing deadlines, such as not being considered for promotions or pay raises, are more likely to take shortcuts and ignore the negative issues that might ensue.[4] Accordingly, we posit the following hypothesis:

Hypothesis 5b: The perceived severity of time pressure positively affects developers' attitude toward unethical IAC reuse.

Avoidance of the Cost of Compliance with License Obligations

Compliance with license obligations may represent a work impediment to professional software developers [14]. A benefit specific to unethical IAC reuse is that it helps to avoid costs of compliance, which can be broken down into two components: investigating which obligations come with the IAC under consideration, and ensuring that these obligations are accounted for properly. Developers will in general hold different perceptions with regard to these two cost components. For instance, developers who have received training on the issue of software licenses will find it less costly to explore the obligations linked to a specific piece of code [67], and

thus be less likely to engage in unethical IAC reuse. Oppositely, developers lacking such knowledge face higher costs of compliance, which they can avoid by choosing to reuse the software unethically. Another factor is the legal language of the license itself. Although a lot of effort has been put into standardization of OSS licenses, subtle differences with potentially large practical implications might be difficult to comprehend for a legally untrained programmer. The same discrepancy can be observed in the costs of ensuring that identified obligations are accounted for. Here, developers typically have to engage with others in their firm (often their supervisors) to determine whether and how these obligations can be fulfilled. If developers expect high costs in the form of lengthy and difficult discussions with their firm, combined with a high likelihood of a negative outcome, they might well consider it attractive to avoid this step, reuse the code right away, and disregard those obligations they are aware of. We thus posit the next hypothesis:

> *Hypothesis 5c: Higher perceived costs of compliance positively affects developers' attitude toward unethical IAC reuse.*

Punishment Severity (Firm)

Counteracting the above benefits is the potential punishment that can result for individual software developers and/or their employers from disregarding license obligations. The related literature distinguishes *punishment severity* and *punishment certainty* (e.g., [74]), each of which should exert a negative effect on levels of unethical or illegal behavior [21]. This effect has been repeatedly confirmed in the IS domain (e.g., [56, 69]). For example, Peace and colleagues [56] clearly established a link between both punishment severity and punishment certainty and the intention of employees to pirate software at their workplace.

In our setting, two different types of punishment severity need to be distinguished: severity for the firm [73] and severity for the individual developer [56]. No such distinction needs to be made for punishment certainty. Typically, people outside the developer's firm have access only to the binary code of the software, and thus they cannot identify individual developers committing unethical IAC reuse. The likelihood that an individual will be able to violate licenses and not be punished for it, even when the employer gets caught, will depend on individual programming skills and firm-level code tracking systems. These two factors are sufficiently captured by the perceived behavioral control dimension of the TPB. In additional robustness checks, we also controlled for, but did not find, significant relationships between punishment severity, punishment certainty, and perceived behavioral control. Similarly, we also controlled for, but did not find, an interaction effect between punishment severity and certainty. Thus, we look at how punishment severity for the firm, punishment severity for the individual, and punishment certainty will affect individual-level attitude toward unethical IAC reuse. Looking first at punishment severity for the firm, there should be developers who know and fear the potential legal, economic, and reputational consequences [10, 67] their employers might face

from unethical IAC reuse. Such developers should accordingly hold a more negative view of unethical IAC reuse [32]. In contrast, developers who perceive the potential consequences for their employers as less severe or do not know about them should hold a more positive view. Accordingly, we posit the following hypothesis:

Hypothesis 6a: Perceived punishment severity for their employer negatively affects developers' attitude toward unethical IAC reuse.

Punishment Severity (Developer)

Unethical IAC reuse may further result in direct punishment of the employee in the form of financial or legal sanctions and potential termination of employment. In fact, firms are increasingly designing explicit rules on reuse of IAC, some banning its reuse altogether and some only allowing reuse of specific types, with these rules also specifying punishments for developers who do not comply [67]. Individual punishments also affect the programmer directly—economically by firing, delayed promotions, and withholding of bonuses and socially through the stigma of unemployment or by assigning blame publicly. Developers who are likely to face more severe individual punishments should hold a less positive attitude toward IAC reuse disregarding potential license obligations (see [10, 21]):

Hypothesis 6b: Perceived punishment severity for themselves negatively affects developers' attitude toward unethical IAC reuse.

Punishment Certainty

Punishment certainty captures the likelihood that someone outside the developer's firm finds that the firm's software contains IAC but does not account for license obligations. Subsequently, it will be at the firm's discretion to identify the individual developer responsible. It is generally assumed that "determining whether [IAC] is present in a corporation's code base is a difficult task to perform accurately" [52, p. 8]. Yet, organizations such as gpl-violations.org have recently been founded to actively pursue the violation of obligations from reused IAC by commercial firms. Developers who are more aware of these developments should perceive a higher punishment certainty and consequently be less inclined to behave unethically [15]. Beyond that, various other factors might matter, such as the programming language employed (as the binary code created by some programming languages can be analyzed more easily than that of others) or the deployment mode of the software (e.g., embedded software vs. standalone software, or few customers vs. many customers), all of which can be judged by individual developers.

Hypothesis 6c: Perceived punishment certainty negatively affects developers' attitude toward unethical IAC reuse.

Data and Methods

We tested our research model by deploying a multimethod research design that follows best practice guidelines (e.g., [4, 11]). Primarily, we drew from a survey of 869 professional software developers (see also [67]). We also relied extensively on a qualitative prestudy comprising thirty-two interviews with an average duration of fifty minutes with industry experts in the field of IAC reuse. To gain insights into commercial software development and code reuse issues, we interviewed twelve OSS developers, seven consultants advising on code reuse, six corporate software developers, three lawyers specializing in source code licensing, and three investment professionals who include review of code licensing in their preinvestment due diligence. Based on qualitative content analysis [50], these interviews deepened our understanding of IAC reuse and its ethical aspects and allowed us to create more meaningful survey questions. Finally, before conducting the actual survey, we pretested the questionnaire extensively. First, it was reviewed by four scholars strongly familiar with the topic. After that, 113 developers from the survey population took part in two rounds of piloting to assess the quality of the survey with respect to its content, scope, and language. In drafting the final questionnaire (see Appendix A), we took into account the pretest results and feedback from the pilot tests and fellow researchers.

Questionnaire Development and Considerations of Common Methods Bias

Aided by our prestudy, we tried to develop a survey instrument capable of accounting for issues of common methods bias that should be particularly prevalent in a study focused on ethics. Here, we follow the recommendations by Podsakoff et al. [58] to minimize underlying biases, caused by, for example, apprehension or social desirability, through survey design and administration.

First, to elicit realistic decision making and to create psychological distance for the survey participants, we applied Fredrickson's [26] scenario method. To provide survey participants with real-life and precisely formulated scenarios about the unethical reuse of IAC [76], we devised two different vignettes to which participants were randomly assigned. We started from Anderson et al.'s [5] nine scenarios illustrating situations calling for ethical decision making in the 1992 Association for Computing Machinery code of conduct. In one of these, they describe a developer who—under time pressure and stuck with technical problems—reuses existing code unethically. Using our insights from the qualitative prestudy, we adapted this scenario to reflect the situation of a developer reusing IAC. These modified scenarios (shown in Appendix B) present a developer named Joe who is under time pressure to complete his module of a development project and unsure how to implement a certain piece of functionality specified for the module. Too resolve this situation, Joe reuses IAC in an ad hoc fashion. Specifically, in scenario 1, Joe

reuses a snippet for which he does not check thoroughly for license obligations. In scenario 2, we vary the type of unethical behavior that Joe exhibits—in reusing a snippet, he checks for the associated license obligations, but purposefully ignores them. The nature of this transgression may be seen as more severe, as Joe is deliberately putting his own interest over that of the company [40, 54]. By comparing the results of scenario 1 with those of scenario 2, we might thus gain additional insight into whether and how this contingency affects individuals' unethical IAC reuse intentions. In a similar vein, as a robustness check, we developed an alternative version of scenario 1 in which we vary the type of code that is reused—instead of a small snippet, Joe reuses a software component.

Second, whenever possible, we relied on survey items validated in earlier studies on unethical behavior—preferably in the IS context and with TPB-based models (e.g., [19, 46, 56])—or slightly adapted them. For the ethical work climate constructs, we employed the original items by Victor and Cullen [80]. No suitable items could be identified for usefulness of IAC, cost of compliance, or punishment certainty. We thus developed new ones based on existing literature on IAC reuse and the information gathered during our interviews and pilot study. All survey items are rated on a seven-point Likert-scale ranging from *strongly disagree* to *strongly agree.* To further reduce social desirability effects, we emphasized the completely anonymous nature of our survey and ensured that all survey items were presented in a nonthreatening, neutral tone. Finally, we used the feedback from our survey pilot and fellow researchers to further improve the wording of our questions. We also attempted to empirically control for common method variance, as described below.

Sample and Data Collection

Because professional software developers from only one or a few firms might not be representative in their beliefs and opinions, our study required a broad sample. To accommodate this need, we chose participants in newsgroups for our survey, with the assumption that a substantial share of the participants in this communication channel would also develop software for a living. To construct our survey population, we identified 528 newsgroups dealing with the topic of software development. In July 2009, we extracted e-mail addresses of those 38,212 individuals who had been active in the previous twelve months (Figure 2). We eliminated 13,525 addresses due to issues such as duplication or because they clearly did not refer to potential software developers. Of the remaining 24,687 addresses, 1,212 were utilized for our pilot studies and 23,475 formed our final survey sample. The survey was conducted in fall 2009, when we randomly selected 14,000 individuals from the population and contacted them via e-mail. Because 2,227 invitations did not reach their recipients, the 1,133 valid responses we received correspond to a 9.9 percent response rate, which is typical for Internet surveys [18]. The high number of undelivered invitations reflects the significant share of newsgroup participants who provide fake or temporary contact information in their profiles. A nonresponse

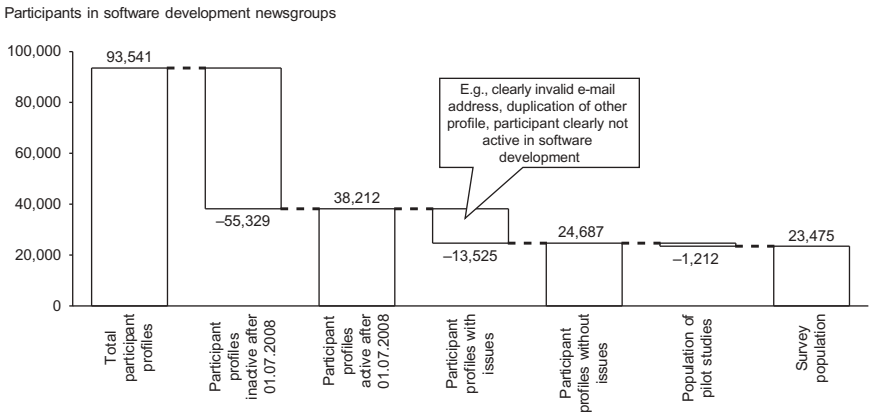

Figure 2. Construction of Survey Population

analysis that compares early to late respondents [6] yielded no indication of bias in the items used in the research model.

Descriptive Statistics

As expected, the majority of respondents to our survey were professional software developers (77 percent). We relied solely on these 869 individuals for our analysis (see Table 1). Of these, 316 completed the questionnaire containing scenario 1 and 297 the one containing scenario 2, and 256 individuals were assigned the robustness check scenario. On average, respondents were 35.6 years old, male (98 percent), and had 9.7 years of programming experience. When self-assessing their software development skills, nearly three-quarters of the developers considered themselves to be above average, and 23 percent even thought of themselves as "excellent." More than half of our sample (56 percent) had participated in OSS projects in the past. Finally, only about one-third of the respondents worked in firms with a policy addressing IAC reuse, suggesting—in accordance with the findings from our pre-study—that many firms still do not address the potential risks of ad hoc IAC reuse by individual developers. We controlled for the presence of such policies in additional robustness checks.

Results

We tested for each scenario in our research model individually using partial least squares (PLS), employing the software SmartPLS [60]. PLS is one of the second-generation multivariate analysis techniques that estimate both measurement and structural models simultaneously in optimal fashion [16]. Given the medium-sized samples for all scenarios and lack of a normal distribution in most of our variables, PLS seemed better suited than the more traditional LISREL approach [36]. All

Table 1. Demographics of Respondents

	Percentage		Percentage
Age (mean: 35.6 years)		Experience as professional software developer (mean: 9.7 years)	
1–19	1	Less than 1 year	4
20–29	34	1–4 years	29
30–39	35	5–9 years	25
40–49	18	10–14 years	23
50+	12	15–19 years	7
Gender		20+ years	12
Male	98	Software development role	
Female	2	Programmer	51
Residence		Software/system architect	28
North America	28	Project manager	4
South America	4	Systems/requirements/business analyst	2
Europe	53	Database designer/developer	2
Asia and rest of world (RoW)	15	Tester	1
Highest level of education		Other role	12
Non-university education	12	Self-assessment of software development skills	
Undergraduate or equivalent	38	Basic	1
Graduate or equivalent	39	Below average	4
Ph.D. or equivalent	11	Average	22
Software development experience in OSS projects		Above average	50
Yes	56	Excellent	23
No	44	Employer has an Internet-accessible code reuse policy	
Working as professional software developer in 2009		Yes	35
Yes	84	No	65
No	16		

Note: $N = 869$.

constructs were modeled as reflective measures. To estimate the significance of path coefficients, we relied on a bootstrapping procedure with 200 samples [33].

Measurement Model Assessment

Evaluating measurement models in PLS requires an examination of their composite reliability, indicator reliability, convergent validity, and discriminant validity [11, 16, 33]. As shown in Table 2, all constructs meet the composite reliability threshold of 0.7, and nearly all items exceed the indicator reliability cutoff value of 0.7. In scenarios 1 and 2, the items PBC2 and PBC4 are slightly below the required value of 0.7. The same is true for item CERT2 in scenario 2. However, the items are retained because the overall constructs exceed the composite reliability threshold of 0.7 in all cases. To ensure convergent validity, that is, establishing that a construct is unidimensional, the average variance extracted (AVE) should be above 0.5. This criterion is met by all constructs in all scenarios (see Table 2). Finally, the results shown in Table 3 confirm the discriminant validity of our constructs, as all of them fulfill the Fornell-Larcker criterion in all three scenarios.

Structural Model Assessment

Following Chin [16] and Henseler et al. [33], we evaluated each model's predictive power through the R^2 values of its ultimate dependent variable (intention) and each model's predictive relevance through the Stone-Geisser criterion (Q^2). The R^2 values for the ultimate dependent variable are 0.417 for scenario 1 and 0.607 for scenario 2 (see Figures 3 and 4), indicating satisfactory explanatory power of our model, and the respective Q^2 values (0.333 and 0.507) suggest acceptable predictive relevance. This increase in values suggests that our research model works better when the scenario describes an actual rather than a potential ethical transgression.

Addressing our baseline hypotheses, attitude and subjective norm significantly drive intention, confirming H1 and H2, with attitude exhibiting a stronger effect than subjective norm. H3, which suggested a positive effect of perceived behavioral control, is not supported.

Of the hypotheses on ethical work climates, we find strong support for H4a, which posited an indirect effect of ethical work climates with respect to compliance with laws and codes on intent, via subjective norms. Oppositely, we have to reject H4b, which argued for an indirect effect of firm-internal rules. We find no direct effects of ethical work climates on intent in either case; additional Sobel tests with regard to H4a produce results on a 5 percent level of significance.

Regarding drivers of attitude, in line with H5b, we find that developers who perceive the consequences of missed deadlines as more severe exhibit a more positive attitude toward unethical IAC reuse. Also, perceived punishment severity for their firm (H6a) or for themselves (H6b) leads to a more negative attitude. Hypotheses 5a and 5c are supported only in scenario 1, in which usefulness (H5a) and the avoidance of the cost

Table 2. Reliablity and Convergent Validity of Measurement Models

	Scenario 1 (N = 316)					Scenario 2 (N = 297)					Robustness check scenario (N = 256)				
Construct/Item	Mean	S.D.	λ	CR	AVE	Mean	S.D.	λ	CR	AVE	Mean	S.D.	λ	CR	AVE
Intention				0.94	0.83				0.94	0.84				0.94	0.84
INT1	2.76	1.66	0.92			2.43	1.55	0.95			2.45	1.56	0.94		
INT2 (R)	5.16	1.77	0.90			5.33	1.76	0.88			5.45	1.66	0.90		
INT3	2.28	1.43	0.91			2.19	1.39	0.93			2.11	1.40	0.91		
Attitude				0.89	0.73				0.89	0.72				0.89	0.72
ATT1 (R)	5.15	1.80	0.81			5.29	1.76	0.77			5.45	1.72	0.84		
ATT2	2.53	1.71	0.90			2.25	1.49	0.89			2.40	1.55	0.87		
ATT3	3.09	1.92	0.85			2.73	1.73	0.88			2.71	1.66	0.83		
Subjective norm				0.92	0.75				0.92	0.73				0.91	0.72
SN1 (R)	4.45	1.76	0.86			4.84	1.68	0.84			4.35	1.76	0.82		
SN2	3.58	1.71	0.86			3.18	1.64	0.85			3.50	1.67	0.83		
SN3	3.52	1.84	0.86			3.45	1.82	0.87			3.33	1.73	0.86		
SN4 (R)	4.54	1.83	0.88			4.82	1.72	0.87			4.89	1.67	0.87		
Perceived behavioral control				0.81	0.52				0.84	0.57				0.87	0.62
PBC1	4.72	2.15	0.84			4.51	2.18	0.87			4.30	2.18	0.90		
PBC2 (R)	3.41	2.14	0.60			3.33	2.15	0.69			3.66	2.17	0.74		
PBC3	4.74	2.00	0.81			4.74	1.98	0.76			4.11	2.02	0.80		
PBC4	5.31	1.75	0.60			5.18	1.86	0.68			4.79	1.91	0.70		

(continues)

Table 2. Continued

	Scenario 1 (N = 316)					Scenario 2 (N = 297)					Robustness check scenario (N = 256)				
Construct/Item	Mean	S.D.	λ	CR	AVE	Mean	S.D.	λ	CR	AVE	Mean	S.D.	λ	CR	AVE
Ethical work climate: Law & code				0.93	0.77				0.92	0.73				0.93	0.77
LAW1	5.01	1.15	0.89			5.05	1.07	0.89			5.11	1.11	0.88		
LAW2	4.91	1.18	0.92			4.93	1.13	0.89			4.96	1.22	0.91		
LAW3	4.99	1.11	0.92			5.03	1.07	0.88			5.09	1.05	0.89		
LAW4	4.18	1.51	0.77			4.12	1.51	0.75			4.36	1.50	0.82		
Ethical work climate: Rules				0.93	0.76				0.91	0.73				0.91	0.72
RULE1	4.55	1.23	0.92			4.53	1.17	0.83			4.59	1.16	0.87		
RULE2	4.57	1.23	0.90			4.60	1.13	0.88			4.71	1.14	0.84		
RULE3	3.95	1.41	0.81			3.99	1.34	0.79			4.08	1.27	0.79		
RULE4	4.65	1.13	0.86			4.59	1.02	0.91			4.62	1.02	0.90		
Usefulness of Internet-accessible code				0.96	0.90				0.96	0.89				0.97	0.90
USE1	4.75	1.68	0.94			4.78	1.66	0.95			4.93	1.66	0.96		
USE2	4.86	1.66	0.97			4.89	1.66	0.92			5.14	1.59	0.96		
USE3	4.98	1.59	0.93			4.88	1.66	0.97			5.06	1.68	0.93		
Severity of time pressure				0.93	0.81				0.94	0.84				0.93	0.83
TIME1 (R)	4.28	1.55	0.92			4.41	1.57	0.95			4.35	1.57	0.93		
TIME2 (R)	4.27	1.55	0.94			4.46	1.62	0.93			4.35	1.59	0.93		
TIME3	3.70	1.56	0.83			3.57	1.58	0.86			3.51	1.53	0.86		

Cost of compliance				0.94	0.89				0.92	0.85			0.94	0.89
COST1*	3.14	1.70	0.95			3.60	1.87	0.87			3.15	1.73	0.94	
COST2*	2.97	1.65	0.94			3.22	1.88	0.97			2.93	1.67	0.95	
Punishment severity (firm)				0.92	0.80				0.92	0.79			0.93	0.82
SFIR1 (R)	3.35	1.77	0.87			3.03	1.76	0.90			2.88	1.77	0.92	
SFIR2	4.68	1.78	0.91			5.00	1.74	0.89			5.13	1.68	0.91	
SFIR3	4.03	1.78	0.90			4.34	1.76	0.88			4.68	1.72	0.89	
Punishment severity (developer)				0.94	0.84				0.96	0.88			0.93	0.82
SDEV1 (R)	3.78	1.86	0.93			3.64	1.79	0.95			3.45	1.76	0.89	
SDEV2	4.16	1.87	0.88			4.52	1.82	0.92			4.64	1.81	0.91	
SDEV3 (R)	3.77	1.82	0.93			3.65	1.82	0.94			3.40	1.77	0.92	
Punishment certainty				0.87	0.70				0.85	0.67			0.92	0.79
CERT1 (R)	4.94	1.73	0.87			4.76	1.82	0.89			4.04	1.88	0.92	
CERT2	2.74	1.61	0.80			2.87	1.65	0.62			3.46	1.80	0.86	
CERT3 (R)	4.25	1.97	0.83			4.34	1.91	0.90			3.80	2.00	0.89	

* The wording of the items differs between the scenarios. See Appendix B.

Notes: (R) = reverse coded item; SD = standard deviation; λ = loading of item on its designated construct; CR = composite reliability; AVE = average variance extracted.

Table 3. Discriminant Validity of Measurement Models

Construct	1	2	3	4	5	6	7	8	9	10	11	12
Scenario 1 (N = 316)												
1. Intention	**0.91**											
2. Attitude	0.63	**0.85**										
3. Subjective norm	0.53	0.68	**0.87**									
4. Perceived behavioral control	0.19	0.23	0.27	**0.72**								
5. Ethical work climate: Law & code	–0.21	–0.21	–0.33	–0.16	**0.88**							
6. Ethical work climate: Rules	–0.15	–0.15	–0.22	–0.19	0.73	**0.87**						
7. Usefulness of Internet-accessible code	0.19	0.11	0.07	0.01	–0.03	0.05	**0.95**					
8. Severity of time pressure	0.17	0.16	0.14	–0.03	0.08	0.11	0.16	**0.90**				
9. Cost of compliance	0.35	0.26	0.27	0.03	–0.15	–0.08	0.04	0.10	**0.94**			
10. Punishment severity (firm)	–0.32	–0.43	–0.39	–0.22	0.27	0.19	0.01	0.03	–0.03	**0.89**		
11. Punishment severity (developer)	–0.36	–0.46	–0.53	–0.21	0.31	0.27	0.00	0.07	–0.07	0.55	**0.91**	
12. Punishment certainty	–0.06	–0.10	–0.16	–0.20	0.00	0.03	–0.05	0.05	–0.02	0.28	0.27	**0.84**
Scenario 2 (N = 297)												
1. Intention	**0.92**											
2. Attitude	0.77	**0.85**										
3. Subjective norm	0.52	0.56	**0.86**									
4. Perceived behavioral control	0.08	0.08	0.17	**0.75**								
5. Ethical work climate: Law & code	–0.27	–0.28	–0.22	–0.08	**0.85**							
6. Ethical work climate: Rules	–0.11	–0.08	–0.13	–0.10	0.65	**0.85**						
7. Usefulness of Internet-accessible code	0.17	0.10	0.07	0.05	0.03	0.07	**0.94**					
8. Severity of time pressure	0.17	0.15	0.03	–0.06	0.12	0.24	0.07	**0.91**				
9. Cost of compliance	0.18	0.15	0.21	0.04	0.04	0.01	–0.05	0.14	**0.92**			
10. Punishment severity (firm)	–0.34	–0.39	–0.36	–0.18	0.22	0.20	–0.02	0.10	–0.08	**0.89**		
11. Punishment severity (developer)	–0.38	–0.42	–0.50	–0.15	0.22	0.14	–0.14	0.16	–0.07	0.59	**0.94**	

12. Punishment certainty	–0.10	–0.14	–0.18	–0.06	0.03	0.03	–0.02	0.13	–0.18	0.20	0.14	**0.82**
Robustness check scenario (N = 256)												
1. Intention	**0.92**											
2. Attitude	0.67	**0.85**										
3. Subjective norm	0.55	0.56	**0.85**									
4. Perceived behavioral control	0.20	0.08	0.30	**0.79**								
5. Ethical work climate: Law & code	–0.26	–0.31	–0.25	–0.07	**0.88**							
6. Ethical work climate: Rules	–0.17	–0.17	–0.17	–0.07	0.65	**0.85**						
7. Usefulness of Internet-accessible code	0.08	0.07	0.12	0.03	0.13	0.01	**0.95**					
8. Severity of time pressure	0.11	0.17	0.09	–0.02	0.06	0.02	0.13	**0.91**				
9. Cost of compliance	0.22	0.14	0.16	0.16	0.03	0.10	–0.06	0.08	**0.95**			
10. Punishment severity (firm)	–0.32	–0.44	–0.41	–0.21	0.24	0.21	–0.06	0.11	–0.11	**0.90**		
11. Punishment severity (developer)	–0.28	–0.34	–0.41	–0.25	0.23	0.21	0.04	0.16	–0.17	0.59	**0.91**	
12. Punishment certainty	–0.15	–0.22	–0.28	–0.26	0.11	0.12	0.08	–0.08	–0.08	0.27	0.33	**0.89**

Notes: The diagonal bolded entries are square roots of the average variance extracted (AVE) of the respective construct; the off-diagonal entries are standardized correlations between constructs. Since the squared values under the diagonal are below the AVEs, the Fornell-Larcker discriminant validity criterion (defined as row and column values being lower than the diagonal values) is satisfied.

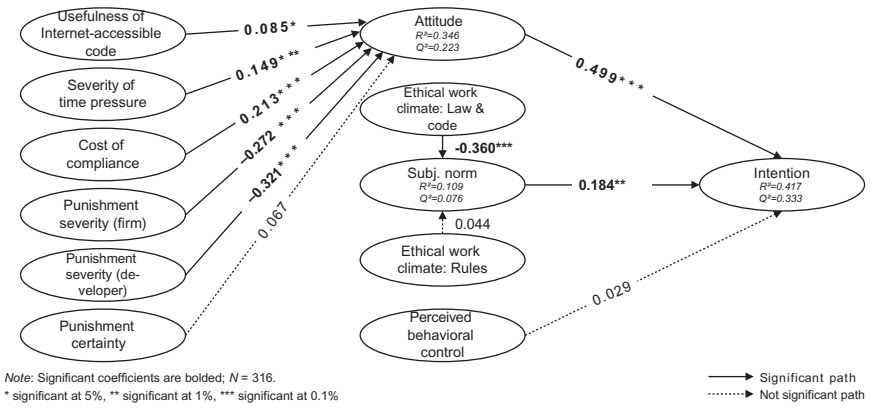

Figure 3. Research Model Estimation in Scenario 1

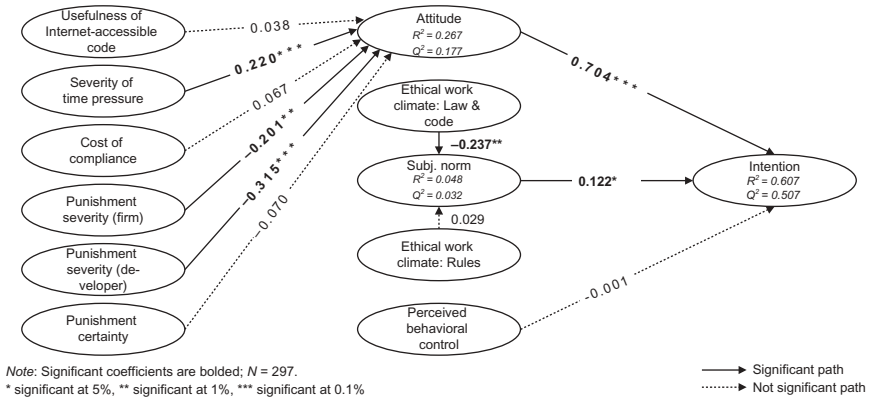

Figure 4. Research Model Estimation in Scenario 2

of compliance (H5c) positively drive attitude. Punishment certainty (H6c) did not exhibit the proposed negative effect. Surprisingly, in the case of scenario 1, we even find a weakly significant effect in the opposite direction. However, it is highly likely that this finding is a statistical artifact.[5] The results of the research model hypothesis testing are summarized in Table 4.

Robustness Checks

We conducted extensive additional analyses to ensure the nonspuriousness of our results. First, to control for common methods variance [see 58], we modeled it separately using (1) a latent variable and (2) a marker variable. For the latter approach, we included the short form of the Marlowe-Crowne social desirability scale [68], which shows cross-loadings with several of our constructs. In doing so, we followed current practice in how to model common methods variance using PLS

Table 4. Summary of Research Model Hypotheses Testing

Hypothesis		Confirmed? Scenario 1	Confirmed? Scenario 2
Theory of planned behavior model			
H1	*Developers' attitude toward unethical Internet-accessible code reuse positively affects their intention to engage in such behavior.*	✓	✓
H2	*Developers' subjective norm concerning unethical Internet-accessible code reuse positively affects their intention to engage in such behavior.*	✓	✓
H3	*Developers' perceived behavioral control regarding unethical Internet-accessible code reuse positively affects their intention to engage in such behavior.*	✗	✗
Ethical work climate			
H4a	*The more a firm's ethical work climate is perceived to emphasize compliance with laws and codes, the lower will be developers' level of subjective norms supporting unethical Internet-accessible code reuse.*	✓	✓
H4b	*The more a firm's ethical work climate is perceived to emphasize compliance with firm rules, the lower will be developers' level of subjective norms supporting unethical Internet-accessible code reuse.*	✗	✗
Cost-benefit calculations			
H5a	*The perceived usefulness of Internet-accessible code will have a positive effect on developers' attitude toward unethical Internet-accessible code reuse.*	✓	✗
H5b	*The perceived severity of time pressure will have a positive effect on developers' attitude toward unethical Internet-accessible code reuse.*	✓	✓
H5c	*Higher perceived costs of compliance will have a positive effect on developers' attitude toward unethical Internet-accessible code reuse.*	✓	✗
The role of punishment			
H6a	*Perceived punishment severity for their firm will have a negative effect on developers' attitude toward unethical Internet-accessible code reuse.*	✓	✓
H6b	*Perceived punishment severity for themselves will have a negative effect on developers' attitude toward unethical Internet-accessible code reuse.*	✓	✓
H6c	*Perceived punishment certainty will have a negative effect on developers' attitude toward unethical Internet-accessible code reuse.*	✗	✗

(e.g., [17]). We found that all approaches produce results that are qualitatively identical to our standard models, so we only report those herein.

Second, because of the high correlation between the two ethical work climate constructs (see Table 3), we also studied separate models containing only one of them, as well as one that combines them into one. We found that these checks also do not qualitatively change any of our results. On its own, the rules construct has the hypothesized negative effect on subjective norm (H4b), but the effect is lower in magnitude than that of the law and code construct; the same applies to the combined construct. This result suggests that, although the effect of the rules dimension might be underreported in our main model, it is still weaker than the effect of the law and code dimension. We also conducted a subsample analysis for each scenario, including only those developers who had indicated that their firms had devised specific rules for IAC reuse, to see whether the respective ethical work climate construct would have a different effect for this group. No differences were found for this construct or any other hypothesis.

Third, one might argue that individuals need to positively value the reuse of IAC in general so that their responses to questions on doing so unethically are not merely hypothetical. To do so, we reduced our sample to include only individuals who think that IAC reuse is at least "somewhat" beneficial according to our usefulness measure. Our results remain qualitatively unchanged, independent of whether or not we still include the usefulness measure in the model.

Fourth, we compared the results of scenario 1 against the robustness check scenario, in which individuals reuse a component instead of a snippet. We found a weakly significant effect of perceived behavioral control, suggesting that individuals' perception of their ability to get away with unethical IAC reuse matters more in this case. Beyond that, we found that significant paths and coefficient sizes more closely resemble scenario 2 than scenario 1, indicating that varied types of IAC reuse are motivated and evaluated differently by different individuals.[6]

Fifth, because component-based estimation models like PLS do not easily allow for inclusion of multiple "simple" control variables, we conducted exploratory multigroup analyses to see whether contextual variables not included in our model might extend our analysis. In separate estimations, we compared individuals whose employers had implemented policies on IAC reuse with those whose employers had not, individuals in different job roles, or individuals of different self-reported skill levels. Yet again, our results remained qualitatively unchanged. Employing other individual-level variables such as developers' age, their home region [40], or their level of job satisfaction for multigroup analysis showed that these are (expectedly) cause of some between-group variation. However, across the three scenarios, they did not cause consistent changes in our variables of interest.

Finally, we applied finite mixture PLS to uncover unobserved heterogeneity in our coefficient estimates (e.g., [59]). If such heterogeneity existed, it might have caused misleading interpretations. However, this analysis did not reveal any consistent latent classes that would need to be accounted for specifically. Consequently, we conclude that our findings are robust.

Discussion and Implications

We set out to explain why individuals engage in a type of unethical programming behavior, the unethical reuse of IAC. This type of unethical behavior, so we argued, is substantially different from established research on IS ethics, in that its creative nature might fundamentally alter the process of unethical behavior, as well as its drivers. Developing an original TPB-based research model, we found the intent to reuse Internet-accessible code unethically to be driven by subjective norms, which are in turn guided by perceptions of a work climate referencing larger professional and legal codes. In addition, although attitude mattered significantly in all our scenarios, we found that drivers of attitude strongly vary between our scenarios, unlike the findings in previous studies such as Bulgurcu et al.'s work on compliance [14]. We also found no significant effect of perceived behavioral control on developers' intent to engage in unethical IAC reuse. Earlier work on ethical behavior (e.g., [43]) has typically explained this finding by arguing that the analyzed behavior is under complete volitional control. Yet, this reasoning seems inapplicable because the standard deviations are not lower than those of other constructs (see Table 3). Rather, we point to a general gap in the literature that seems to fall short of coherently explaining the role of perceived behavioral control when ethical behavior is at stake, which future research should address.

Implications for Theory

Our findings allow us to contribute to the IS literature with regard to ethical decision making in programming and reusing code, evaluation of behavior, and the role of an ethical work climate.

First, by investigating the determinants of individuals' attitude toward unethical IAC reuse, this study sheds light onto how individuals arrive at ethical judgments while programming. This finding extends insights gained from studies focusing on the (passive) adoption of new tools, processes, or techniques that individuals in IS may draw upon—for example, those building on the technology acceptance model (e.g., [78])—in which benefit components are usually found to outweigh cost factors. Oppositely, we find that a teleological weighing of positive and negative consequences, in particular regarding the risk of punishments resulting from unethical behavior, helps to explain individual-level attitudes: our results clearly show that the constructs capturing costs from eventual punishments seem to be of higher importance than any construct reflecting perceived benefits. Notably, such pronounced differences between costs and benefits are also not found in studies of unethical behavior in less creative settings, such as software piracy [56].

In this vein, we also note substantial differences across scenarios. This finding is in contrast to a purely rationality-based approach, as exemplified by Bulgurcu et al. [14]. In such an approach, the components from which a cost-benefit analysis is derived should exhibit uniform behavior when only the magnitude of unethical behavior is manipulated. We conjecture that the unequal importance of certain factors in teleological

evaluations stems from the associated deontological assessment of the scenarios. In scenario 1, programmers might be able to convince themselves that a violation of license terms is unlikely and a careless approach is justifiable, but scenario 2 describes actual, unambiguous unethical reuse. We call for future research to further explore the dependency of teleological evaluations on deontological judgments. Our study provides some initial insights in this direction by showing individuals' changing rationale in their cost-benefit evaluation of unethical IAC reuse: In scenario 1, they incorporate a full set of consequences when forming their attitudes. In scenario 2, only the severity of time pressure facilitates a positive evaluation, making clear how the context of action will matter tremendously for deriving ethical and moral judgment (also see [40, 54]) in programming.

In sum, we show how ethical judgments in TPB-based models may vary, even if the underlying ethical issue is effectively the same. For researchers, this implies that to ensure compatibility of results of studies on IS ethics, they should reuse existing scenarios to extend prior research. A potential explanation for these differences may be found in the social psychology literature, more notably self-concept maintenance theory [51]: When weighing opposing arguments in an ethical dilemma, people want to maximize benefits while at the same time maintaining a positive view of themselves [28, 51]. So when there is the possibility to behave unethically, self-serving rationalizations and justifications may be idiosyncratically created to convince oneself that an activity is acceptable. Creative persons, such as programmers, are especially able to come up with such justifications [28, 51], particularly in scenarios in which justifications include only minor reinterpretation of what is unethical behavior, or in ambiguous situations [64].

Second, we contribute to work on the ethical work climate construct, particularly in the IS context (e.g., [48, 75, 77]). By integrating its "law and code" and "rules" dimensions into a TPB-based model, we respond to Flannery and May's call to "examine the direct effect of organizational climate on individual ethical decision making" [24, p. 656]. We show that an ethical work climate has no direct effect on intention but becomes a constituting element of individuals' subjective norm, in that it drives their assumption about their peers' evaluation of what is acceptable behavior. We thus suggest that individuals do not exhibit ethical behavior because of an ethical work climate. Rather, the ethical work climate influences what individuals perceive to be their peers' judgment of what is appropriate behavior (which in turn guides individual behavior), explaining the mixed findings of earlier studies on the role and efficacy of ethical climates and codes [31, 37, 57].

Relatedly, we also uncover an important relational element, an aspect of ethical decision making often underrepresented even in the literature on ethics and morality (e.g., [41, pp. 320–386]). Specifically, we highlight that individual perceptions are driven *more* by the desire to adhere to institutional norms that originate from beyond the boundary of the firm, as opposed to firm-specific rules. In this context, Harrington [31] suggested that firm-specific codes are less effective than professional codes because they are not IS specific. We show how external norms remain more important *even if* internal rules are IS specific. We argue that this consistent difference in evaluation is a result of the higher degree of institutionalization of

external aspects of ethical work climates: it is early and strong socialization that creates the feeling of belonging to a profession or practice. If such an institution, in turn, advocates ethical programming behavior, it will strongly affect individuals' assessment of the subjective norm—more strongly than firm-specific rules individuals may get to know later in their careers. In this vein, future research might thus analyze IAC reuse as a practice, based on institutional or structuration theory, both of which have previously been applied to the field of IS (e.g., [35, 55]).

Finally, our overall model allows us to contribute to ethical research in IS. In general, our study is the first to concomitantly theorize deontological and teleological judgments by jointly drawing on expected utility, deterrence, and ethical work climate theory. Doing so allows us to study the effect of the respective constructs net of each other, whereas prior research would have either ignored one of these elements or operationalized them jointly. For example, Harrington [31] argued that an ethical work climate fosters a deterrent character, especially if a professional code of ethics is enforced, but she does not control for an actual deterrent effect in her study. Bulgurcu et al. [14] looked at deterrence and expected utility theory, but their study in turn did not look into the roles of punishment certainty and ethical work climates. We encourage future research to follow this comprehensive approach to foster understanding of ethical decision making. To do so, researchers might apply TPB-based models similar to ours, enriched by theorizing on perceived behavioral control, to explain the mixed findings of our work and earlier work. Furthermore, we extend ethical research in IS to include unethical behavior in the context of programming, with unethical IAC reuse as our object of study. So far, neither the stream of ethics research in the IS context (e.g., [49, 56]) nor the body of literature that investigates IAC reuse in commercial software development [e.g., 25] has looked at the potentially unethical nature of developers' ad hoc IAC reuse. We show how IAC reuse can be modeled as an individual-level decision substantially affected by ethical considerations as reflected in the effects of deterrence and an ethical work climate. In this context, our findings also add an ethical component to the literature on reuse of knowledge, of which software is just one instance (e.g., [47]). Here, we point toward a potential darker side of reuse and provide an explanation for its existence. In doing so, we provide a basis for more work on the ethics of knowledge reuse, its theoretical and individual-level foundations, and its implementation.

Limitations and Suggestions for Future Research

This study has some limitations that need to be taken into account. First, with a response rate of 9.9 percent, our study is in line with many other Internet-based surveys but still relies on a relatively small share of the population when generalizing its results. Second, the study was conducted among professional software developers active in newsgroups. However, these may well be younger, more OSS savvy, and more proficient in software development. Third, although we have taken measures to reduce social desirability effects, we still cannot completely eliminate a possible distortion of our results. Fourth,

as is common in the literature [1, 9], we use intention and not the actual behavior as our dependent variable. However, attempting to link individual coding behavior to a survey soliciting individuals' motivation to unethically reuse IAC would in itself raise ethical questions and also severe concerns of social desirability, far beyond their possible impact in our current study. Nonetheless, future studies should strive to capture actual ethical behavior to scrutinize the intention-behavior relationship and potential moderators, as well as the direct effects of the variables on behavior. Fifth, future research might inquire into the efficacy of some of the levers we identify. For example, researchers could look into how varied levels of perceived punishment severity and certainty affect unethical behavior.[7] In line with our findings, such work might also incorporate considerations of being outcast by one's colleagues or profession to more elaborately account for the effects of an ethical work climate we describe. Sixth, given our research design, we could not measure individuals' level of awareness of the ethical dilemma.[8] Still, awareness has been shown to influence outcome beliefs in similar contexts [14], and its inclusion in work extending ours might shed further insight into how individuals form ethical evaluations. Seventh, we do not differentiate between punishment certainty for firms and developers. Although it is possible that programmers may perceive punishment certainty differently for the firm than for themselves, we argue that individual skill is the main determinant of getting caught. This ability in turn is, in our view, sufficiently captured by the perceived behavioral control dimension of the TPB. Finally, although programming is a creative task, not all programmers are equally creative. Our results need additional verification in a context in which programmers are less skilled than the ones in our sample. In such contexts, we would expect the lure of unethical IAC reuse to be even higher (in particular, the benefits of usefulness of code and alleviating time pressure). At the same time, people may be less driven by external than by firm-internal normative forces, so that the balance between the law and code dimension, on the one hand, and the rules dimension, on the other, may change. In this vein, we would expect similar effects if the task at hand to be carried out by a programmer becomes duller, as would be caused by, for example, a change of setting from programming from scratch to mundane updating or maintenance work.

Both our findings and our limitations point out avenues for future research. Beyond scrutinizing external validity, future work should try to capture the dependent variable of our research model through means other than self-reporting to reduce worries about social desirability. Data on the dependent variable might be gathered from, for example, firms scanning the software they develop for reused IAC with automated tools to detect instances of violated license obligations. In addition, the finding that perceived behavioral control does not influence, or only weakly influences, developers' intent to engage in unethical IAC reuse is surprising and questions the effect of tools integrated into developers' workstations to protect them from the risks of ad hoc IAC reuse. The growing popularity of such tools warrants further and more explicit examination to understand exactly how such tools affect professional software developers and their work.

Implications for Practice

Our findings highlight several levers firms can employ to influence individual-level ethical decision making to ameliorate potential firm-level risks. In particular, they support recent practitioner-oriented work calling on firms to reuse IAC *systematically* [62]. Regarding ad hoc reuse, first, firms should inform developers about the considerable consequences of disregarding license obligations, and enforce them in order to maintain perceived severity. Second, developers in firms in which missed deadlines have less severe negative consequences tend to engage in less unethical IAC reuse. A general policy of not enforcing deadlines is, of course, unrealistic given that firms themselves face deadlines enforced by their customers [7]. Yet, firms need to learn to set realistic deadlines in project management to make severity considerations from missing them less relevant. Third, firms should deploy measures to reduce developers' costs of compliance. For example, firms might want to create internal databases or Wikis, or FAQs, or appoint internal IAC experts to serve as first point of information for developers (also see [62]). Finally, firms should strive to build or foster a work climate in which compliance with the norms provided by institutions such as the law and the professions is deemed important.

Conclusion

This study makes several contributions to the literature on ethical decision making. We introduce programming, as one of the most central IS activities, into ethical decision-making research. First, we show that this context differs from other unethical behavior, such as software piracy, in that programmers are weighing the consequences of potential unethical actions, giving more weight to potential punishment than perceived benefits. Second, we offer an explanation for mixed findings in earlier studies on the effect of ethical work climates and codes. We show that climate influences intention indirectly, through what individuals perceive to be their peers' ethical standards. Also, unlike previous literature suggests, we find that external norms carry more weight than internal ones, even if those rules are specific to the subject matter. Third, our study is the first to theorize a model of unethical behavior based on deontological and teleological judgments by jointly incorporating utility, deterrence, and ethical work climate theory. This comprehensive approach also allows us to study concurrent effects net of each other and explain previous mixed results, while at the same time adding an ethical view on the literature on reuse of knowledge. Finally, our investigation on how programmers make their ethical judgments enables us to give concrete practical advice on preventing unethical code use.

Acknowledgments: This article has greatly benefited from the guidance of Professor Zwass, an associate editor, and three reviewers. Oliver Alexy is grateful for the support of both the UK Innovation Research Centre (RES/G028591/1; sponsored by the Economic and Social Research Council; the National Endowment for Science, Technology and the Arts; the UK

Department for Business, Innovation and Skills; and the Technology Strategy Board) and the Engineering and Physical Sciences Research Council (GR/R95371/01). We appreciate valuable comments from Burcu Bulgurcu and Carol Saunders, and from seminar participants at Technische Universität München and the 2011 Academy of Management Conference. All errors are, of course, our own.

Notes

1. Amabile [3] defined creativity as a product or process that is both novel and appropriate (i.e., useful, valuable) and requires that the problem it solves be heuristic rather than algorithmic. Programming also satisfies this definition: programming requires both a deep understanding of the principles that underlie a problem and a very high degree of accuracy in its implementation, similar to mathematics or creative writing [22].

2. For example, Linksys developed router software that contained a substantial share of OSS code from the Internet not properly accounted for. As a result, Cisco, which later acquired Linksys, was obligated to share large parts of the router's source code with its customers, permitting modification and redistribution without a fee. Similarly, all software companies relying on secrecy may fear the consequences of unethical reuse of IAC. For example, VMware wrote in the "risk" section of its June 2008 quarterly filings that reused IAC could require it to "release the source code of our proprietary software, which could substantially help our competitors develop products that are similar to or better than ours."

3. Notably, however, compliance behavior is to be considered significantly different from unethical behavior. In this vein, regarding their own study, Bulgurcu et al. explicitly called for future work to "investigate the joint role of consequence-based motivations and morality/values on employee compliance behavior" [14, p. 544].

4. The research model does *not* test a relationship between the general existence of time pressure and developers' attitude. Whereas most software development projects have some degree of time pressure, developers only react to this if they perceive "severe" consequences from not meeting the resulting deadlines. Furthermore, existence of time pressure may vary from project to project and could even differ for different points in time within a project. Contrary to this, severity of time pressure should be rather stable over time within a firm and thus should relate better to attitude, which is also expected to be rather stable over time.

5. When we eliminate the two punishment severity items that show medium correlations with punishment certainty (see Table 3), the effect of punishment certainty becomes negative and significant, as proposed by H6c for scenarios 1 and 2.

6. The corresponding robustness check scenario for scenario 2, the reuse of a complete module in conscious violation of its license obligations, was considered rather rare by our interview partners and hence was not included in our analysis.

7. We thank our careful reviewers for this suggestion as well as the two suggestions listed as numbers 6 and 7.

8. Given that we used a scenario experiment, we would have needed to ask for individuals' perception of Joe's awareness, which would have been unlikely to produce meaningful or reliable results. Nonetheless, in robustness checks for which we try to control for individuals' awareness in their personal coding situations, we find that our results remain qualitatively unchanged.

References

1. Ajzen, I. The theory of planned behavior. *Organizational Behavior and Human Decision Processes*, *50*, 2 (1991), 179–211.

2. Ajzen, I. Constructing a theory of planned behavior questionnaire: Conceptual and methodological considerations, 2002 (available at http://people.umass.edu/aizen/pdf/tpb.measurement.pdf).

3. Amabile, T.M. The social psychology of creativity: A componential conceptualization. *Journal of Personality and Social Psychology*, *45*, 2 (1983), 357.

4. Anderson, J.C., and Gerbing, D.W. Structural equation modeling in practice: A review and recommended two-step approach. *Psychological Bulletin*, *103*, 3 (1998), 411–423.

5. Anderson, R.E.; Johnson, D.G.; Gotterbarn, D.; and Perrolle, J. Using the new ACM code of ethics in decision making. *Communications of the ACM*, 36, 2 (1993), 98–107.

6. Armstrong, J.S., and Overton, T.S. Estimating nonresponse bias in mail surveys. *Journal of Marketing Research*, *14*, 3 (1977), 396–402.

7. Austin, R.D. The effects of time pressure on quality in software development. *Information Systems Research*, *12*, 2 (2001), 195–207.

8. Banerjee, D.; Cronan, T.P.; and Jones, T.W. Modeling IT ethics: A study in situational ethics. *MIS Quarterly*, *22*, 1 (1998), 31–60.

9. Beck, L., and Ajzen, I. Predicting dishonest actions using the theory of planned behavior. *Journal of Research in Personality*, *25*, 3 (1991), 285–301.

10. Berenbach, B., and Broy, M. Professional and ethical dilemmas in software engineering. *Computer*, 42, 1 (2009), 74–80.

11. Boudreau, M.-C.; Gefen, D.; and Straub, D.W. Validation in information systems research: A state-of-the-art assessment. *MIS Quarterly*, *25*, 1 (2001), 1–16.

12. Brooks, F.P. *The Mythical Man-Month: Essays on Software Engineering*. Reading, MA: Addison-Wesley, 1975.

13. Buchan, H.F. Ethical decision making in the public accounting profession: An extension of Ajzen's theory of planned behavior. *Journal of Business Ethics*, *61*, 2 (2005), 165–181.

14. Bulgurcu, B.; Cavusoglu, H.; and Benbasat, I. Information security policy compliance: An empirical study of rationality-based beliefs and information security awareness. *MIS Quarterly*, *34*, 3 (2010), 523–548.

15. Chen, Y.; Ramamurthy, K.; and Wen, K.-W. Organizations' information security policy compliance: Stick or carrot approach? *Journal of Management Information Systems*, *29*, 3 (2012), 157–188.

16. Chin, W.W. The partial least squares approach to structural equation modeling. In G.A. Marcoulides (ed.), *Modern Methods for Business Research*. Mahwah, NJ: Lawrence Erlbaum, 1998, pp. 295–358.

17. Chin, W.W.; Thatcher, J.B.; Wright, R.T.; and Steel, D. Controlling for common method variance in PLS analysis: The measured latent marker variable approach. In H. Abdi, W.W. Chin, V. Esposito Vinzi, G. Russolillo, and L. Trinchera (eds.), *New Perspectives in Partial Least Squares and Related Methods*. New York: Springer, 2013, pp. 231–239.

18. Couper, M.P. Web surveys: A review of issues and approaches. *Public Opinion Quarterly*, *64*, 4 (2000), 464–494.

19. Coyle, J.R.; Gould, S.J.; Gupta, P.; and Gupta, R. "To buy or to priate": The matrix of music consumers' acquisition-mode decision-making. *Journal of Business Research*, *62*, 10 (2009), 1031–1037.

20. Cronan, T.P., and Al-Rafee, S. Factors that influence the intention to pirate software and media. *Journal of Business Ethics*, *78*, 4 (2008), 527–545.

21. Ehrlich, I. Participation in illegitimate activities: A theoretical and empirical investigation. *Journal of Political Economy*, *81*, 3 (1973), 521–565.

22. Ershov, A.P. Aesthetics and the human factor in programming. *Communications of the ACM*, *15*, 7 (1972), 501–505.

23. Fishburn, P. *Utility Theory for Decision Making*. New York: John Wiley & Sons, 1970.

24. Flannery, B.L., and May, D.R. Environmental ethical decision making in the U.S. metal-finishing industry. *Academy of Management Journal*, *43*, 4 (2000), 642–662.

25. Frakes, W.B., and Kang, K. Software reuse research: Status and future. *IEEE Transactions of Software Engineering*, *31*, 7 (2005), 529—536

26. Fredrickson, J.W. An exploratory approach to measuring the perceptions of strategic decision process constructs. *Strategic Management Journal*, *7*, 5 (1986), 473–483.

27. Gino, F., and Ariely, D. The dark side of creativity: Original thinkers can be more dishonest. *Journal of Personality and Social Psychology*, *102*, 3 (2012), 445–459.

28. Gino, F.; Ayal, S.; and Ariely, D. Contagion and differentiation in unethical behavior: The effect of one bad apple on the barrel. *Psychological Science*, *20*, 3 (2009), 393–398.

29. Gotterbarn, D., and Miller, K.W. Unmasking your software's ethical risks. *IEEE Software*, *27*, 1 (2010), 12–13.

30. Haefliger, S.; von Krogh, G.; and Spaeth, S. Code reuse in open source software. *Management Science*, *54*, 1 (2008), 180–193.

31. Harrington, S.J. The effect of codes of ethics and personal denial of responsibility on computer abuse judgements and intentions. *MIS Quarterly*, *20*, 3 (1996), 257–278.

32. Henkel, J. Champions of revealing—The role of open source developers in commercial firms. *Industrial and Corporate Change*, *18*, *3* (2009), 435–471.

33. Henseler, J.; Ringle, C.M.; and Sinkovics, R. The use of partial least squares modeling in international marketing. *Advances in International Marketing*, *20* (2009), 277–320.

34. Hunt, S.D., and Vitell, S. A general theory of marketing ethics. *Journal of Macromarketing*, *6*, 1 (1986), 5–16.

35. Jones, M.R., and Karsten, H. Giddens's structuration theory and information systems research. *MIS Quarterly*, *32*, 1 (2008), 127–157.

36. Jöreskog, K.G., and Wold, H. *The ML and PLS Technique for Modelling with Latent Variables: Historical and Comparative Aspects*. Amsterdam: North-Holland, 1982.

37. Kaptein, M., and Schwartz, M. The effectiveness of business codes: A critical examination of existing studies and the development of an integrated research model. *Journal of Business Ethics*, *77*, 2 (2008), 111–127.

38. Kim, Y.E., and Stohr, E.A. Software reuse: Survey and research directions. *Journal of Management Information Systems*, *14*, 4 (1998), 113–147.

39. Knuth, D.E. Computer programming as an art. *Communications of the ACM*, *17*, 12 (1974), 667–673.

40. Kohlberg, L. *The Philosophy of Moral Development*. San Francisco: Harper & Row, 1981.

41. Kohlberg, L. *The Psychology of Moral Development*. San Francisco: Harper & Row, 1984.

42. Krueger, C.W. Software reuse. *ACM Computer Surveys*, *24*, 2 (1992), 131–183.

43. Kurland, N. Ethical intentions and the theories of reasoned action and planned behavior. *Journal of Applied Social Psychology*, *25*, 4 (1995), 297–313.

44. Levi, S.D., and Woodard, A. Open source software: How to use it and control it in the corporate environment. *Computer & Internet Lawyer*, *21*, 8 (2004), 8–13.

45. Lim, W.C. Effects of reuse on quality, productivity, and economics. *IEEE Software*, *11*, 5 (1994), 23–28.

46. Limayem, M., and Hirt, S.G. Force of habit and information systems usage: Theory and initial validation. *Journal of the AIS*, *4* (2003), 65–97.

47. Markus, M.L. Towards a theory of knowledge reuse: Types of knowledge reuse situations and factors in reuse success. *Journal of Management Information Systems*, *18*, 1 (2001), 57–93.

48. Martin, K., and Cullen, J. Continuities and extensions of ethical climate theory: A meta-analytic review. *Journal of Business Ethics*, *69*, 2 (2006), 175–194.

49. Mason, R.O. Four ethical issues of the information age. *MIS Quarterly*, *10*, 1 (1986), 5–12.

50. Mayring, P. Qualitative content analysis. In U. Flick, E; von Kardoff, E., and I. Steinke (eds.), *A Companion to Qualitative Research*. London: Sage, 2004, pp. 266–269.

51. Mazar, N., Amir, O., and Ariely, D. The dishonesty of honest people: A theory of self-concept maintenance. *Journal of Marketing Research*, *45*, 6 (2008), 633–644.

52. McGhee, D.D. Free and open source software licenses: Benefits, risks, and steps toward ensuring compliance. *Intellectual Property & Technology Law Journal*, *19*, 11 (2007), 5–9.

53. Mingers, J., and Walsham, G. Toward ethical information systems: The contribution of discourse ethics. *MIS Quarterly*, *34*, 4 (2010), 833–854.

54. Moores, T.T., and Chang, J.C.J. Ethical decision making in software piracy: Initial development and test of a four-component model. *MIS Quarterly*, *30*, 1 (2006), 167–180.

55. Orlikowski, W.J., and Barley, S.R. Technology and institutions: What can research on information technology and research on organizations learn from each other? *MIS Quarterly*, *25*, 2 (2001), 145–165.

56. Peace, A.G.; Galletta, D.F.; and Thong, J.Y.L. Software piracy in the workplace: A model and empirical test. *Journal of Management Information Systems*, *20*, 1 (2003), 153–177.

57. Pierce, M.A., and Henry, J.W. Judgements about computer ethics: Do individual, co-worker, and company judgements differ? Do company codes make a difference. *Journal of Business Ethics*, *28*, 4 (2000), 307–322.

58. Podsakoff, P.M.; MacKenzie, S.B.; Lee, J.-Y.; and Podsakoff, N.P. Common method biases in behavioral research: A critical review of the literature and recommended remedies. *Journal of Applied Psychology*, *88*, 5 (2003), 879–903.

59. Ringle, C.M.; Sarstedt, M.; and Mooi, E.A. Response-based segmentation using FIMIX-PLS: Theoretical foundations and an application to American customer satisfaction index data. In R. Stahlbock, S.F. Crone, and S. Lessmann (eds.), *Annals of Information Systems*, vol. 8. Special issue. Berlin: Springer, 2010, pp. 19–49.

60. Ringle, C.M.; Wende, S.; and Will, S. SmartPLS 2.0 (M3). Hamburg: SmartPLS, 2005 (available online www.smartpls.de).

61. Rosen, L. *Open Source Licensing: Software Freedom and Intellectual Property Law*. Englewood Cliffs, NJ: Prentice-Hall, 2004.

62. Ruffin, C., and Ebert, C. Using open source software in product development: A primer. *IEEE Software*, *21*, 1 (2004), 82–86.

63. Schoemaker, P.J.H. The expected utility model: Its variants, purposes, evidence and limitations. *Journal of Economic Literature*, *20*, 2 (1982), 529–563.

64. Schweitzer, M., and Hsee, C. Stretching the truth: Elastic justification and motivated communication of uncertain information. *Journal of Risk and Uncertainty*, *25*, 2 (2002), 185–201.

65. Shalley, C.E.; Zhou, J.; and Oldham, G.R. The effects of personal and contextual characteristics on creativity: Where should we go from here? *Journal of Management*, *30*, 6 (2004), 933–958.

66. Sojer, M., and Henkel, J. Code reuse in open source software development: Quantitative evidence, drivers, and impediments. *Journal of the AIS*, *11*, 12 (2010).

67. Sojer, M., and Henkel, J. License risks from ad-hoc reuse of code from the Internet: An empirical investigation. *Communications of the ACM*, *54*, 12 (2011), 74–81.

68. Strahan, R., and Gerbasi, K.C. Short, homogeneous versions of the Marlow-Crowne Social Desirability Scale. *Journal of Clinical Psychology*, *28*, 2 (1972), 191–193.

69. Straub, D.W. Effective IS security: An Empirical Study. *Information Systems Research*, *1*, 3 (1990), 255–276.

70. Straub, D.W., and Collins, R.W. Key information liability issues facing managers: Software piracy, proprietary databases, and individual rights to privacy. *MIS Quarterly*, *14*, 2 (1990), 143–156.

71. Tetlock, P.E. Accountability: The neglected social context of judgement and choice. In L.L. Cummings and B.M. Staw (eds.), *Research in Organizational Behavior*. Greenwich, CT: JAI Press, 1985, pp. 297–332.

72. Thong, J.Y.L., and Yap, C.-S. Testing an ethical decision-making theory: The case of softlifting. *Journal of Management Information Systems*, *15*, 1 (1998), 213–237.

73. Thornton, D.; Gunningham, N.A.; and Kagan, R.A. General deterrence and corporate environmental behavior. *Law & Policy*, *27*, 2 (2005), 262–288.

74. Tittle, C.R. *Sanctions and Social Deviance: The Question of Deterrence*. New York: Praeger, 1980.

75. Trevino, L. Ethical decision making in organizations: A person-situation interactionist model. *Academy of Management Review, 11*, 3 (1986), 601–617.

76. Vance, A.; Lowry, P.B.; and Eggett, D. Using accountability to reduce access policy violations in information systems. *Journal of Management Information Systems*, *29*, 4 (2013), 263–290.

77. VanSandt, C.V.; Shepard, J.M.; and Zappe, S.M. An examination of the relationship between ethical work climate and moral awareness. *Journal of Business Ethics*, *68*, 4 (2006), 409–432.

78. Venkatesh, V.; Morris, M.G.; Davis, G.B.; and Davis, F.D. User acceptance of information technology: Toward a unified view. *MIS Quarterly*, *27*, 3 (2003), 425–478.

79. Victor, B., and Cullen, J.B. A theory and measure of ethical climate in organizations. In W.C. Frederick and L.E. Preston (eds.), *Research in Corporate Social Performance and Policy*, Greenwich, CT: JAI Press, 1987, pp. 51–71.

80. Victor, B., and Cullen, J.B. The organizational bases of ethical work climates. *Administrative Science Quarterly*, *33*, 1 (1988), 101–125.

81. Wyld, D.C., and Jones, C.A. The importance of context: The ethical work climate construct and models of ethical decision making—An agenda for research. *Journal of Business Ethics*, *16*, 4 (1997), 465–472.

82. Yoon, C. Digital piracy intention: A comparison of theoretical models. *Behaviour & Information Technology*, *31*, 6 (2012), 565–576.

Appendix A. Questionnaire Items

Construct/ item code	Item	Items adapted from
Intention: How likely is it that while working at YourCo, you will do what Joe did as described in the scenario?		
INT1	I may do what Joe did in the future	[46]
INT2 (R)	I would never do what Joe did	
INT3	It is likely that I will do what Joe did in the future	
Attitude: If you, while working at YourCo, were in a situation similar to Joe's, what would you think about doing what he did?		
ATT1 (R)	For me at YourCo, doing what Joe did would be foolish in a similar situation	[9]
ATT2	For me at YourCo, doing what Joe did would be justified in a similar situation	
ATT3	When doing what Joe did in a similar situation at YourCo, the benefits would outweigh the downsides for me	
Subjective norm: What would other people say if they learned that while working at YourCo, you had done what Joe did in the scenario?		
SN1 (R)	Most of my friends would disapprove	[9]
SN2	Most of my friends would think that it is okay	
SN3	Most of my colleagues at YourCo would not mind	
SN4 (R)	Most of my colleagues at YourCo would disapprove	
Perceived behavioral control: Do you think that while working at YourCo, you would be able to do what Joe did in the scenario?		
PBC1	Personally, I could easily do what Joe did if I wanted to	[2, 9]
PBC2 (R)	Based on my knowledge and skills, I would find it difficult to do what Joe did	
PBC3	There is nothing outside of my control which could prevent me from doing what Joe did	
PBC4	It would be mostly up to me whether or not I do what Joe did	

(continues)

Appendix A. Continued

Ethical work climate: Law & code: Please provide some information about the general climate at YourCo by indicating how true the following statements are for YourCo.		
LAW1	People at YourCo are expected to comply with the law and professional standards over and above other considerations	[80]
LAW2	At YourCo, the law and ethical codes are a major consideration	
LAW3	At YourCo, people are expected to strictly follow legal and professional standards	
LAW4	At YourCo, the first consideration is whether a decision violates any law	
Ethical work climate: Rules: Please provide some information about the general climate at YourCo by indicating how true the following statements are for YourCo.		
RULE1	It is very important to follow the company's rules and procedures at YourCo	[80]
RULE2	At YourCo, everyone is expected to stick by company rules and procedures	
RULE3	Successful people at YourCo go by the book	
RULE4	People at YourCo strictly obey the company policies	
Usefulness of Internet-accessible code: How useful do you think would it be for your work at YourCo to download snippets/components from the internet and integrate them into the software you are developing for YourCo?		
USE1	It would improve my job performance	*NEW*
USE2	It would increase my productivity	
USE3	It would make it easier for me to do my job	
Severity of time pressure: How serious do you think it would be for you personally if you failed to deliver required functionality on time at YourCo?		
TIME1 (R)	It would not hurt my career much	[19]
TIME2 (R)	It would not affect my future much	
TIME3	There would be major negative consequences for me	
Cost of compliance (used in scenario 1 and the robustness check scenario): How easy would it be for you to check thoroughly for potential obligations that come with snippets/components from the Internet that you want to integrate when working at YourCo?		
COST1	It would take very long for me to thoroughly check for all obligations that come with the snippets/components	*NEW*
COST2	It would be very difficult for me to check for all potential obligations of the snippets/components	
Cost of compliance (used in scenario 2): How easy do you think would it be for you to discuss with YourCo about complying with the obligations of snippets that you want to integrate in your work?		
COST1	Such discussions would take very long	*NEW*
COST2	Such discussions would be very difficult	
Punishment severity (firm): How serious do you think would be the consequences for YourCo if it became public that their software includes snippets/components from the Internet, but does not fulfill the obligations of these snippets/components?		
SFIR1 (R)	There would be no or very low consequences for YourCo	[56]

(continues)

Appendix A. Continued

SFIR2	YourCo would be in serious legal trouble	
SFIR3	YourCo would incur major financial losses	
Punishment severity (developer): How serious do you think would be the consequences for you personally if you were caught doing what Joe did in the scenario while working at YourCo?		
SDEV1 (R)	It would not hurt my career much	[19]
SDEV2	There would be major negative consequences for me	
SDEV3 (R)	It would not affect my future much	
Punishment certainty: How easy do you think it would be to detect that YourCo's software contains snippets/components from the Internet?		
CERT1 (R)	It would be very difficult for anybody to find out	*NEW*
CERT2	The probability that anybody would find out is very high	
CERT3 (R)	Scanning for snippets/components from the internet in YourCo's software is virtually impossible	

Note: (R) = reverse-coded item; *NEW* = because no existing items were available to measure the respective construct, new items were developed based on a thorough review of relevant literature and our qualitative prestudy. The items were further tested and refined during two rounds of survey pretests; survey participants were asked to assume "YourCo" is the last firm for which they have been developing software. All of the above items are measured on a seven-point Likert scale anchored by 1 (*strongly disagree*) and 7 (*strongly agree*).

Appendix B. Questionnaire Case Vignettes

Scenario 1: Not checking thoroughly for obligations from snippet reuse

a. Setting: Joe works as programmer for a software firm and is responsible for developing one module of the firm's next software product for private consumers. In the project there is enormous time pressure and Joe is already behind schedule.
b. Problem: While Joe is a good programmer, there is a certain functionality specified in the requirements of his module that he is not sure how to implement. Additionally, he figures that even if he manages to implement this functionality, it would take very long to do so and he would be late with his module.
c. Joe's Approach: In order to avoid missing his deadline, he searches the Internet for source code that implements the required functionality. He is happy to find a project with such code, accesses its code base and—with minor modifications—copies and pastes the lines of code from the Internet project that implement the required functionality for his own project. When copying and pasting the snippets, Joe does not check thoroughly whether there are obligations that he has to fulfill when integrating them.

d. The End: Through the use of the snippets, Joe manages to deliver his module with all required functionality on time. In the end, the firm's product comes to stores and is sold many times, so that Joe and his bosses are happy.

Scenario 2: Knowingly ignoring obligations from snippet reuse

a. Setting: *[SAME AS IN SCENARIO 1]*
b. Problem: *[SAME AS IN SCENARIO 1]*
c. Joe's Approach: In order to avoid missing his deadline, he searches the Internet for source code that implements the required functionality. He is happy to find a project with such code, accesses its code base and—with minor modifications—copies and pastes the lines of code from the Internet project that implement the required functionality for his own project. Further, Joe checks for obligations that come with using code from the Internet project and learns that it demands that he make the source code of his whole module also available on the Internet. However, Joe knows that discussing this within his firm would take very long, and there is a chance that his firm would disagree. Thus, to avoid losing precious time over discussions, he does not account for the obligation and makes some further changes to the code copied and pasted to make sure that it is not a direct copy anymore. He removes some comments, changes some variable names, and restructures the code sequence a little.
d. The End: Through the use of the snippets, Joe manages to deliver his module with all required functionality on time. In the end, the firm's product comes to the stores and is sold many times, so that Joe and his bosses are happy.

Robustness check scenario: Not checking thoroughly for obligations from component reuse

a. Setting: *[SAME AS IN SCENARIO 1]*
b. Problem: *[SAME AS IN SCENARIO 1]*
c. Joe's Approach: In order to avoid missing his deadline, he searches the Internet for a component that provides the required functionality. He is happy to find one, downloads it, and integrates it into his module. When downloading and integrating the component, Joe does not check thoroughly whether there are obligations that he has to fulfill when integrating this component.
d. The End: Through the use of the component, Joe manages to deliver his module with all required functionality on time. In the end, the firm's product comes to the stores and is sold many times, so that Joe and his bosses are happy.