Computer Programs, Dialogicality, and Intentionality

Josh Tenenberg
University of Washington Tacoma
Institute of Technology
Tacoma, WA 98402
+1-253-692-5860
jtenenbg@uw.edu

Yifat Ben-David Kolikant
Hebrew University of Jerusalem
School of Education
Jerusalem, 91905
+972-2-588-2056
yifat.kolikant@mail.huji.ac.il

ABSTRACT

Computer programs are addressed to two different audiences: to the computer, which interprets the program according to the formal semantics of the programming language in which it is written, and to human readers, who try to discern how the program will operate in a real-world context. In this paper, we use Bakhtin's notion of *dialogicality,* along with recent research in psycholinguistics and evolutionary psychology, as a theoretical basis for reflecting on the way in which computer programs embed cooperative communicative norms between programmers and program readers, and how these can be and sometimes are exploited in the program text. In doing so, this provides an important set of theoretical lenses for undertaking and interpreting empirical research in computer science education.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education—*Computer science education*

General Terms

Human Factors, Languages.

Keywords

Dialogicality; shared intentionality; collective intentionality; common ground; cooperative discourse maxims; Bakhtin.

1. INTRODUCTION

This paper began as a dialog between its authors about the relationship between dialogicality as originally conceptualized by Bakhtin [1, 2] and computer science education. Our argument in a nutshell is this. According to Bakhtin, human speech is explicitly shaped by the speaker with the addressee in mind. Each unit of speech uttered by a speaker during their speech turn references and responds to prior discourse and anticipates the response of the recipient that is yet to come. Such speech is constrained not only by the formal system of grammar that defines the language [7] but also by conventionalized speech forms or *genres* [1]. Recent empirical research in psycholinguistics, establishes that interlocutors, in whatever written or spoken genre that they use for communication, shape their speech cooperatively for mutual intelligibility. People follow normative cooperative speech maxims [17], and in so doing, rely upon the shared common ground between them, that is, the mutual knowledge that each has not only about the shared topic of interest but about one another's knowledge [10]. They thus, for instance, take as normative that utterances are relevant, perspicacious, and honest. The cooperative motive in developing common ground *through* speech relies upon a tacit assumption that speakers have concerning themselves and one another not only as goal-directed intentional agents, but as agents capable of *shared* intentionality [28]. As a result, they rely upon their mutual capability to engage in socially recursive inference about one another's accruing mental state about the topic at hand as well as self-monitor their success at achieving mutual intelligibility [32].

ICER '14, August 11 - 13 2014, Glasgow, United Kingdom
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2755-8/14/08?$15.00.
http://dx.doi.org/10.1145/2632320.2632351

These characteristics of human communication provide an important set of theoretical lenses for empirical research in computer science education, especially, the genres that emerge as students interact within their educational arena and the interlocutors they take into account. Looking at human communication as cooperative and dialogical helps to articulate processes of knowledge *creation* (of individual and groups), and emphasizes individual agency as an individual finds "voice" (without neglecting the power that the societal world puts on the individual).

To illustrate this perspective in relation to concrete research questions, we consider the question of whether novices consider computer programs to be a form of speech, and if so, who are the recipients for this speech, with what characteristics? If they consider the computer to be a recipient, then do they construe the computer as intentional, whether weakly, as they might consider a food-seeking animal, or strongly, in the human, socially recursive way? Taking this intentional stance concerning the computer may cause the learner considerable difficulty when the computer does not act as an intelligent interlocutor that responds to the learner's program as either a goal-seeking or socially recursive agent would. As importantly, there are additional research questions concerning the extent to which learners recognize that other people may be recipients of their program text, and hence these readers will seek to render the program intelligible with respect to the imputed intentions of the authoring programmer. If no such audience is recognized, then novice learners may see little reason to use conventionalized program genres to express their computational ideas nor for providing linguistic cues to aid the human who will read the program in pursuit of their goal of understanding the programmer's intentions. Although we have anecdotal information that some students treat the computer as having strong intentionality and that many students ignore other human readers as an audience, we have virtually no systematic studies of either of these beliefs and their prevalence (though [29], discussed below is an exception). And in particular, we have no theoretical account for when and why students might have these beliefs, without which it becomes difficult to structure educational interventions to help students develop more expert-like communicative beliefs and practices.

This paper, then, explicitly relies upon both old and new theoretical perspectives on social cognition and human communication, speculating that these may be crucial to understanding how people learn to program computers. In doing so, we situate this work amidst recent CS education research that borrows from an important strand of recent research in Science Education, in that learners of the discipline have internal cognitive resources developed from their work in other domains that they can and do bring to bear within the programming activity and learning, e.g. [12, 19]. In particular, by casting computer programs as speech acts, we consider that novices learning to program might, can, and sometimes do rely upon their prior (and extensive) experience as skilled natural language users.

2. BAKHTINIAN DIALOGICALITY

In his writings, Bakhtin asserts that human speech is *dialogical* [1, 2, 40]. That speech is dialogical means that it always occurs within some existing speech context, in which (other) people have already spoken beforehand and will speak afterward. "[A]ny speaker is himself a respondent to a greater or lesser degree. He is not, after all, the first speaker, the one who disturbs the eternal silence of the universe. And he presupposes not only the existence of the language system he is using, but also the existence of preceding utterances—his own and others'—with which his given utterance enters into one kind of relation or another (builds on them, polemicizes with them, or simply presumes that they are already known to the listener). Any utterance is a link in a very complexly organized chain of other utterances" [1]. Each new utterance adds meaning to the discourse as it unfolds through time.

An *utterance* is the primitive speech unit, and the corresponding atomic communicative event is the *speech turn*, the change of speakers from one to another (for a similar approach from a sociological perspective, see [27]). An utterance, then, is the speech that emerges from a single speaker from the beginning to the end of their speech turn. Speakers alternate their speech, giving a direction and linearity to it. In being dialogical, uttered speech always presupposes a hearer, an *addressee*, and so speech responds to prior utterances, and anticipates the responses of recipients to the utterance. Speech is therefore explicitly shaped with the addressee in mind.

Bakhtin also underscores the polyphonic, heteroglossic nature of speech: extant in any individual's utterance is reference to and echoes of prior utterances of self and others. This reference to past speech is what Garfinkel [16] and Cicourel [8] have called *indexicality*. The sedimentation of such past references to others from a discourse community are conventionalized in speech genres. To Bakhtin, such speech genres stand between the formal, generative grammar of any particular human language (e.g. as formalized by Chomsky [7]) and concrete utterances. That is, speech users are constrained not only by the grammatical structure of their particular natural language (even as they might extend, play with, and satirize this formal structure), they are also constrained to speak within particular genres (e.g. the joke, the greeting, the novel, the tweet) that are appropriate for the audience, the context, and their communicative goals. "We are given these speech genres in almost the same way that we are given our native language, which we master fluently long before we begin to study grammar. We know our native language—its lexical composition and grammatical structure—not from dictionaries and grammars but from concrete utterances that we hear and that we ourselves reproduce in live speech communication with people around us. We assimilate forms of language only in forms of utterances and in conjunction with these forms. The forms of language and the typical forms of utterances, that is, speech genres, enter our experience and our consciousness together, and in close connection with one another. To learn to speak means to learn to construct utterances (because we speak in utterances and not in individual sentences, and, of course, not in individual words)" [1].

Viewing human communication as dialogical thus can be viewed as operating at two distinct levels. On the first level, communication is mutualistic, a relation between the speaker and the recipient of the speech who in turn responds, reversing the speaking and recipient roles, and on and on. This mutualism diachronically unfolds in the back-and-forth of real voices in sequential utterance. On the second level, it is collectivized and conventionalized, abstracted above the specific utterances and experiences of pairs of individuals into generic forms that are shared by entire cultural communities. Natural language is the outermost and broadest form for human communication and speech genres are forms specialized to particular, repeated kinds of human interaction.

3. THE PSYCHOLINGUISTICS OF HUMAN COMMUNICATION

3.1 Cooperative discourse maxims

Psycholinguistic research within the last several decades has provided empirical support for this Bakhtinian perspective of dialogicality in human communication, at both the mutualistic and the collective levels. In his seminal work on speech use, Grice [17] argues that speakers shape their locutions and hearers make inferences from such locutions based on shared principles of cooperativeness that each implicitly and normatively follows; and that, in fact, mutual intelligibility is virtually impossible without such underlying assumptions. In short, speakers and hearers assume that the other uses *cooperative norms* (what Grice called *maxims*) in order to carry out discourse, which is always a joint production. These maxims relate to *quantity* ("make your contribution as informative as is required (for the current purposes of the exchange)" (p45)), *quality* ("do not say what you believe to be false" (p46)), and *manner* ("avoid obscurity of expression … be brief" (p46)). So for instance, if I say to you "I went shopping yesterday," you can, in general, infer that I in fact did go shopping, that this fact is relevant to a prior or upcoming conversation, and that I would not so inform of you of this fact if I believed you knew that I had gone shopping yesterday. Grice thus presupposes and explicates both levels of communication outlined in our discussion of dialogicality: cooperative mutuality between speaker and hearer, and conventionalized cultural norms about speech use.

3.2 Common ground

Clark [10, 11] extended Grice's cooperative principle by arguing that one of the key cooperative activities that interlocutors engage in is *grounding*, the establishment of common ground between them. Common ground is the knowledge that each holds in common, and that each knows the other knows is held in common, recursively ad infinitum. For instance, suppose that you and I are watching a classroom lecture, and the teacher has just written "$f = ma$" on the whiteboard. If I see that you are looking at the whiteboard, and I see that you see that I am looking at the whiteboard, then I can take it that not only do I know what is on the whiteboard, but that I know that you know what is on the whiteboard, and you know that I know what is on the whiteboard, and I know that you know that I know what is on the whiteboard, etc. In having common ground, you and I can and do make use of this mutually held knowledge in shaping and interpreting our utterances. So for example, I can say to you "I wonder if we'll get tested on this," under the assumption that you and I both understand what *this* refers to, since we both hold it in common, and we both assume that I am following the Gricean cooperative maxims (e.g. of relevance, informativeness, quantity, quality, etc.). But if you had not attended the lecture, and we are having coffee afterward, I will instead say "I wonder if we will get tested on the formula relating force to mass and acceleration" or similar. The concept of common ground is thus important because cooperative principles of speech such as what Grice propose exploit the common ground between interlocutors.

From any speaker's perspective, however, common ground is never guaranteed; I may shape my speech under assumptions about your knowledge that turn out to be false. According to Clark, in their turn-taking behavior, speakers *present* an utterance, which they do not take as being mutually understood (i.e. common ground) until there is *acceptance* by the hearer. 'Note that the acceptance process is recursive. B's evidence in response to A's presentation is itself a presentation that needs to be accepted. But where does the recursion stop? . . . What keeps the process from spinning out indefinitely?' [9]. What prevents infinite recursion is (a) that the interlocutors 'mutually believe that the partners have understood what the contributor meant to a criterion sufficient for current purposes' [11], i.e. they settle for weaker evidence depending on context, and (b) they settle for decreasing strength of evidence as the conversation continues [9].

Common ground occurs from several sources. First, we can have common ground due to shared background knowledge, e.g. that we are both lawyers and we each mutually know that we are lawyers, hence we can in mutual knowledge make reference to things that in common ground all lawyers know. Second, we can develop common ground from sharing experience in mutual knowledge, e.g. that we are both observing the same lecture and can see that we are doing so. Third, common ground emerges from the shared and ongoing conversation that we together create, i.e. we mutually assume that the other knows the conversation that has already transpired between us.

3.3 Least collaborative effort

Based on the notion of common ground, and recognizing that interlocutors never act simply as speakers but in both roles as speaker and hearer in an ongoing conversation, Clark and colleagues proposed a principle of communicative efficiency between conversational participants relative to the *joint* work that they undertake. This is in contrast to Grice's maxims, which are from the point of view of the speaker. Clark calls this *the principle of least collaborative effort*: "In conversation, the participants try to minimize their collaborative effort—the work that both do from the initiation of each contribution to its mutual acceptance" [11]. Thus, not only am I honest, perspicuous, unambiguous, relevant, etc., but you and I *together* as speakers and recipients who trade roles with each speech turn in an ongoing manner minimize our joint work in carrying out our joint communicative goals.

Rather than communication being a matter of simply transmitting "content" from speaker to hearer, then, communicators shape their communication and interpretation of the speech of others according to the principle of least collaborative effort, relying upon the common ground that the interlocutors share, and tacit assumptions about the kinds of inference that the other interlocutor carries out. The mutual inference and activity goes something like this: "I intend that you know something, and so I refer your attention or imagination to some situation (my referential act) in the hope that you will figure out what I intend you to know (my communicative intention). Then you, relying on our common ground (both personal and cultural), hypothesize abductively what my communicative intention might be, given that I want you to attend to this referential situation" [32].

4. Intentionality

4.1 Theory of mind

From Grice and Clark, we have an account of human communication that accords with both the mutualism and the generic conventionalism inherent in dialogicality. Underlying these accounts are assumptions about human cognition that interlocutors make toward one another. The fundamental assumption that human interlocutors make about one another that we will be explicating is often referred to by the term *theory of mind* [22, 36], i.e. that people treat others as having mental activity (such as goals, desires, and plans) similar to themselves. As an assumption about another interlocutor, it is easy not only for interlocutors but for researchers in the social sciences to take these human capabilities for granted. But because computer programmers engage in discursive activity with both the computer and with other programmers—who differ in their cognitive capabilities—it is important to make explicit what it means to have a theory of mind *about other people*, and how this kind of cognition differs from other, weaker forms of cognition.

Tomasello and his colleagues have undertaken considerable research related to the development of human cognition and the associated forms of communication, theorizing from a series of empirical studies with great apes (chimpanzees, bonobos, orangutans, gorillas) and with human infants and children. They have done so both from a phylogenetic perspective concerning the evolution of the human species as distinct yet emergent from that of our nearest primate relatives, as well as from an ontogenetic perspective of the emergence of different communicative and cognitive competencies in individual human development [32, 34, 35]. Since the evidence indicates that the ontogenetic mirrors the phylogenetic, we will tell the phylogenetic story, mentioning only that, ontogenetically, shared intentionality emerges in early infancy and collective intentionality (both defined below) emerges during the fourth year of life (for an extended discussion of the ontogenetic story, see [36–38]). Our primary source for the account we provide here is Tomasello's most recent explication in *A Natural History of Human Thinking* [32], which encapsulates and unifies much of his and colleagues' research over the last three decades. We additionally reference some of Tomasello's prior research and that of others working in related fields.

4.2 Individual intentionality

There is considerable evidence that great apes (and hence the common ancestor to apes and humans from over 6 million years ago) are intentional creatures who mentally represent the world, make inferences from these representations, and self-monitor their thinking and behavior. By intentionality, we mean "the capacity of the mind to represent objects and states of affairs in the world other than itself" [28], i.e. "the way the individual mind is directed at objects by virtue of some mental content that represents them" [15]. The evidence suggests that great apes have intentionality: they mentally represent the world imagistically, schematically, and situationally with respect to their goals ("e.g. that food is present or that a predator is absent" (p28)). Their inferences allow them to make "off-line simulations in which the subject infers or imagines nonfactual situations" (p28) and they have "proto-versions of everything from modus tollens to disjunctive syllogisms" (p28). In sum, the great apes have what Tomasello calls *individual intentionality* since they use their cognitive abilities to counterfactually simulate activity prior to in-the-world activity, thereby letting their thoughts "die in their stead". And although they can reason about social situations, they do so primarily in *competition* rather than cooperation with other individuals of the same species.

4.3 Shared intentionality

Tomasello asserts that the next form of intentionality began somewhere between 2 million and 400,000 years ago, shortly after the emergence of the genus *homo.* This change was due largely to increased pressure for small-scale cooperation in order to procure game animals that are too large for an individual human to reliably kill themselves [35]. With this increased pressure came considerable changes to human physiology, such as increased support for two-legged standing, walking, and (especially) running, and changes to the shape of the face, neck and throat to support articulate speech [20]. Most important, however, were the cognitive changes that led to what Tomasello calls *shared intentionality,* borrowing form notions of shared intentionality hypothesized by philosophers of language [28]. In shared intentionality, two individuals not only have goals and intentions, but they develop the mutual intention to do something *together*. This joint activity has a dual character, in that it involves not only something joint (the goal, the joint action), which must be mentally represented, but it also involves both actors taking separate but complementary roles in carrying out the joint action, and having the capability to take the perspective of the other in order to aid them in completing their role-specific responsibilities. Their mental representations thus had to explicitly include *both* individuals in the joint activity, their inferences had to be socially recursive, in the sense described above (where each in mutual knowledge knows that the other knows, ad infinitum), and they had to self-monitor, not only with respect to their success in their communicative activity, but to ensure that they are achieving their role-specific behavior that they have committed to jointly undertake. Along with these new cognitive capacities came new normative dimensions to human behavior, since undertaking joint activity commits its participants to ongoingly engage in this shared endeavor—and to publicly display this ongoing activity and commitment—in spite of individual incentives to pursue competing individual goals. Part of this normativity concerns uses of language that take account of socially recursive thought, in that "linguistic constructions are created with adaptations for the recipients' knowledge, expectations, and perspective in mind" [32].

4.4 Collective intentionality

Finally, starting approximately 200,000 years ago, humans began a new form of intentionally, largely because of demographic pressure from increased population, and the often lethal *inter*-group competition that arose in competition over finite resources [5, 6]. There were thus a number of cognitive changes that arose for small bands (approximately 200 people) of individuals to identify as a shared collective with a shared culture and group identity. What had earlier been mutualistic normative commitments and forms of communication between pairs of individuals among pre-modern humans gave way to "objective" knowledge, generalized, impersonal group norms and generic, conventionalized forms of communication among modern humans. Not only did individuals have to be able to take the perspective of a cooperative partner, they additionally had to develop abstracted, objective forms of knowledge (e.g. how the entities in the world are classified and named, their properties, their relations). As well they had to adopt institutionalized social practices, i.e. idealized ways of doing things that were not only functional to achieve valued goals, but had corresponding agent-neutral social roles with normative obligations for fulfilling responsibilities associated with each role. Being a member of the group meant "knowing what we all know" and "doing it the way we all do it," which was not only enforced mutually by committed cooperators, but, unlike any other species, enforced by uninvolved third parties [23]. Along with this came social forms of teaching children so that cultural forms of being and doing became common ground, not only among two individuals engaged in a spontaneous form of cooperation in the moment, but to all members of the cultural group [33]. This meant that any subgroup from within the larger tribal band could generate stable expectations for behavior in order to flexibly and spontaneously engage in cooperative activity as demanded by whatever contingency the group faced. This conventionalized form of intentionality is what Tomasello calls *collective intentionality*. "Everything is genericized to fit anyone in the group in an agent-neutral manner, and this results in a kind of collective perspective on things, experienced as a sense of the 'objectivity' of things, even those we have created" [32]. And along with it came the kind of conventionalized, generic forms of communication that Bakhtin speaks of. Conforming to shared cultural forms of speech at the same time signals membership and affinity toward the group, enhances mutual intelligibility, particularly with other members of the cultural community with whom we share no direct common ground, and reproduces the cultural form across time.

4.5 Three forms of intentionality

To summarize, we can talk about three distinct forms of intentionality. The first, individual intentionality, characteristic of the great apes, involves possessing mental representations, making inferences from these to replace, supplant, and/or anticipate action in the world, and to self-monitor with respect to achieving ones goals. Shared intentionality, characterizing the genus *homo*, involves developing joint goals and plans that explicitly include both participants acting in complementary roles, making socially-recursive inferences, along with the second-personal normative commitments associated with undertaking joint goals and plans. With this form of intentionality, interlocutors explicitly create and interpret speech with respect to their communicative goals and what they take to be shared common ground. And finally, collective intentionality, possessed only by modern humans, involves generalized cultural forms of knowing and doing within a group (including generic forms of speech), and generic (rather than mutual) forms of normativity that are common ground among all members of the cultural group, enforced by third parties.

5. DIALOGICALITY, INTENTIONALITY, AND COMPUTER PROGRAMS

Having undertaken this review of intentionality and dialogicality, we can now connect the theoretical framework discussed above to our illustrative research concern within computer science education, that of treating computer programs as communicative objects. In doing so, it is important to recognize that computer programs as utterances have two distinct recipients—1) computers and 2) other programmers—hence two different "interpreters" of these texts and two distinct senses of "meaning" [30].

5.1 The computer interpreter

From an expert perspective, the computer interprets a program in precise ways according to the formal semantics associated with each syntactic construct, as documented in such things as programming language reports and embodied in software interpreters. We also recognize that this interpretation is mediated through various "layers" or "levels," e.g. from a user interface to an Integrated Development Environment, to an interpreter that may use one or more intermediate languages before these are interpreted for the particular hardware that we happen to running, etc. Nonetheless, each level translates through well-defined semantics to the next level below. This view of the meaning of a computer program treats computers as hermetically sealed, so that semantics in this formal sense is only with respect to the ways in which particular operations (e.g. an assignment statement or a conditional) change the computer's internal state. Under this view, such things as conventionalized linguistic genre and style are simply irrelevant as far as the interpreted computer action is concerned: the identifier `Rose` by any other name is just as sweet from the interpreter's point of view. This treats the computer from a point of view that Dennett calls the *design stance* [13], where one can make predictions of its behavior based on an understanding of its functional design properties. To reiterate, this is an expert perspective on the computer. We thus take as an assumption that expert computer scientists and software developers, when programming a computer, take the design stance with respect to the computer, a stance that both authors of this paper share. In doing so, we set aside the larger question of whether computers actually *have* intentionality, one of the most contentious philosophical questions of the latter part of the 20th century [13, 14]

Novice programmers and "just plain folks" (a phrase coined by Jean Lave [18]), on the other hand, may treat the computer as having intentionality, i.e. such cognitive things as beliefs, goals, plans, etc., especially if novices do not have a sufficiently well-developed model of the computer's operation to take the design stance. Such intentional attributions can be seen in statements that any introductory computer science teacher has heard such as "the computer doesn't like me," "it is acting up," "it has a mind of its own," "it just does what it wants" and similar. This is what Dennett calls taking the *intentional stance* or *intentional strategy*. "To a first approximation, the intentional strategy consists of treating the object whose behavior you want to predict as a rational agent with beliefs and desires and other mental stages exhibiting what Brentano and others call *intentionality*" [13].

5.2 The human interpreter

Let us now consider the other recipient of computer programs: all of the programmers who might happen to read the program, human beings who may be charged with having to debug the program, port it to another language, write a unit test suite for it, or any of a countless number of programming-related activities. These program readers consider that the original programmer *is* intentional, and that the computer program is written so as to carry out some humanly-devised goal with respect not only to internal changes of computer state, but how such computer state changes affect the larger state of affairs out in the world. So, for example, what a program statement "means" for a human program reader (and the original programmer, who is self-reflectively also a reader) is not simply that some memory location called "`Order`" receives the value "`Rose`" but that this state change is one part of a larger sociotechnical process that will result in someone having a rose delivered to him or her, such as when someone orders flowers for his or her mother from a florist's website. And so anyone reading the program not only has to internalize the interpreter, so that she can simulate what the program will do in execution, but at the same time has to infer a model of the intentions of the person who programmed the computer. In addition, and particularly for programming teachers when they read student code, the teacher has to infer the student's model of the interpreter that this student has internalized, particularly if there are systematic kinds of errors in the program. Readers are thus socially recursive, in that they have to think about the intentions of the original programmer. Hence, expert programmers are socially recursive in their software development activities, recognizing that their programs will be read by others who will have to infer the program author's intentions. These program authors thus add particular linguistic elements to aid the reader in this task, such as comments, identifier names that suggest larger goals and intentions, unit tests, documentation, and the like. Finally, programmers and readers are collectively intentional, since, as members of programming organizations and communities, they develop and use conventionalized stylistic elements, as well as programming genres in order to "minimize the joint collaborative effort" of programmer and (human program reader), even if neither has ever established direct common ground in the past. That is, experts follow conventionalized ways of expressing programs because of general normative requirements associated with being members of particular cultural programming communities (e.g. put curly braces in these places, put constants before variables before methods, and so on). As a result, program readers can develop stable expectations about how computer programs will be written.

There are a number of empirical research studies on expert/novice differences in reading and comprehending computer programs, such as that of Petre [25], who reports that experts "see" computer programs differently than novices based on different expectations about how programs are written. There is also research on teaching and learning general cultural norms within computing communities [3, 4] But these do not take the perspective that program readers and writers are bound by cooperative *discursive* norms such as those that Grice describes for natural language, nor are such programming norms explored and enumerated. In the only empirical study that we know of related to the cooperative linguistic norms of programmers and program readers, Soloway and Ehrlich [29] identify several such norms, such as "variable names should reflect function" and "don't include code that won't be used." They conclude: "[I]t is not merely a matter of aesthetics that programs should be written in a particular style. Rather there is a psychological basis for writing programs in a conventional manner: programmers have strong *expectations* that other programmers will follow these discourse rules. If the rules are violated, then the utility afforded by the expectations that programmers have built up over time is effectively nullified." To clarify, we are not claiming that programmers and program readers are not bound by discursive norms; on the contrary, we believe that they are. Rather, discursive norms (and the collective intentionality that they imply) have rarely been used as the theoretical framing for empirical studies of programmers, whether expert or novice.

To summarize, to experts, computer programs have two audiences, two distinct readers, one of whom is non-intentional, the other of whom is fully, collectively, socially-recursively intentional. And we can "see" this intentionally inscribed directly in the program: the identifier `Rose` by any other name might make a large difference with respect to the way in which I as the original programmer intend for this program to function in the world and you as the (human) program reader are to understand my intentions. As a result, computer programs embed the dialogicality of human programmers in all of the Bakhtinian senses mentioned above: addressable (to its human readers), heteroglossic (in carrying the "voice" of past programmers in the program code, perhaps in conventionalized programming schemas [21]), and expressed within conventionalized programming genres (which vary within different programming communities, even if they share the same programming language or paradigm).

6. AN EMPIRICAL RESEARCH PROGRAM

Taking programs as utterances with two distinct audiences, one of which is a computer and non-intentional, the other of which is human and collectively intentional, opens a space of empirical investigation with respect to novices learning how to program. We see two ways in which novices might construe programs in ways that differ from the dual-recipient view outlined above. And these misconstruals may serve as considerable hindrances to learning.

6.1 Treating the computer as intentional

In an earlier, study, Pea [24] identified that many students treat computers as intentional, which he termed a "superbug" that has a number of different manifestations. He warns, however, "[i]t is not that students *literally* believe that the computer has a mind, or can think, or can interpret what was not explicitly stated. In our experience, novice programming students are likely to vehemently deny that the computer can think or that it is intelligent. … But students' behaviors when working with programs often contradict their denials; they act *as if* the programming language is more than mechanistic. Their default strategy *for making sense* when encountering difficulties of program interpretation or when writing programs is to resort to the powerful analogy of natural language conversation, to assume a disambiguating mind which can understand" (p33).

Given these results, where might the theoretical borrowings that we explicate in prior sections provide additional insight? One is in studying why some students take the intentional stance and others do not. And, given our distinction of kinds of intentionality, individual, shared, and collective, which kinds are attributed to computers? Are there students who believe that they and the computer they are programming are accumulating common ground based on the *history of interaction* that they are jointly accruing, as a human partner would? Do they believe that the interpreter's linguistic output (such as prompts, auto-completions, dialogs, and error messages) should follow the cooperative maxims and principle of least collaborative effort as human interlocutors are obliged to follow?

Perhaps most importantly, might different kinds of instructional interventions, perhaps some that explicitly link forms of intentionality to communicative competence, help students to take the design stance rather than the intentional stance when programming computers?

6.2 Not fulfilling communicative obligations to anticipated human readers

The second set of empirical investigations concerns the extent to which novices are aware of the intentional, human recipient for their programs. To what extent do they see themselves as participating in human-to-human discourse about computational things expressed in a programming language? What do students believe that human readers require in order to infer the programmer's intentions? That is, what norms do they have about *how* programs should be written so as to be read and understood *by others*? What conventionalized forms of programming discourse (if any) do students believe they should internalize (in addition to internalizing the interpreter)? We conjecture that it is likely that novices (as well as most programming teachers and CS Ed researchers) are unaware of the normative principles (i.e. Gricean cooperative axioms, exploitation of common ground, principle of least collaborative effort) that they use in inter-human communication, just as we ourselves were unaware of such practices prior to reading the research summarized above. And this is because such communicative practices are learned socially at an early age through imitation, implicit use, and correction by experienced others, in the same way that children implicitly learn the grammar of their native languages (a recursive, infinitely generate grammar), and hence become expert users of grammar without any awareness of it. And if this is the case, then novice programmers may see little reason to write their programs in conventionalized ways.

7. IMPLICATIONS OF ADOPTING THE INTENTIONAL AND DIALOGIC PERSPECTIVE

There are several implications of studies concerning the intentional stance of novices toward the computer and of their awareness of human program readers and their normative obligations toward them. *If* some novices believe that the computer is intentional (and we speculate that this is the case), then this suggests that these novices have appropriated cognitive resources that they use in communicative activity with human beings (or perhaps a weaker form such as they might use with pet dogs), and that it will be problematic for them if they persist in taking the intentional stance with the computer. Similarly, *if* a novice does not recognize that there is a strongly intentional human agent that may have to read their code (to figure out what it does, to change it, etc), then it may be problematic for this student to both read and understand programs, particularly with respect to adopting certain genre norms and other linguistic conventions that make computer programs intelligible to other people.

The perspective that we have outlined in this paper, which we shorthand as the *dialogic perspective* (and all that it implies about intentionality) also provides an alternative reading for some prior research on student misconceptions and epistemology. For example, in their oft-cited paper "Epistemological Pluralism," Turkle and Papert [39] lament what they claim to be dominant cultural ways of thinking about computer programs, particularly by computer scientists teaching novices at such places as Harvard. These culturally dominant ways of thinking, they state, privilege the use of black-boxed, computational abstractions rather than code written "from scratch" anew by each programmer, thereby alienating many students—many of whom are women—who prefer a more concrete, bottom-up form of thinking and programming. An anchoring belief that Turkle and Papert express is "*the computer is an expressive medium that different people can make their own in their own way* [emphasis in original]" (p165). In speaking of the computer as an *expressive* medium, but ignoring the *communicative* aspects of computer programs, Turkle and Papert treat individuals as social isolates. Computers are thus naturalized as found objects, rather than bequests from cultural predecessors.

Taking a dialogical perspective on cultural forms for writing and thinking about programs, yields a different insight into Turkle and

Papert's data, and leads to different conclusions. And that is that cultural communities develop programming genres, conventionalized ways of writing programs (e.g. black-box, top-down) that have deontic requirements for members of these communities: in order to be one of us, this is how we write programs. Simply having functionally correct programs is not sufficient; rather, programs have to be expressed in ways that meet the genre norms of the community. Expressive artifacts come with no such obligations, but communicative artifacts do.

Moreover, these conventions, these genres, are not mere caprice, but rather are community wisdom which evolved through much experience and interaction. These norms should not be considered as suppressing students' expression but rather as empowering them to express themselves *within and in relation to the community*. Furthermore, norms constantly evolve as new members of communities introduce new practices, and communities encounter and adapt norms from other communities [26]. For example,, "concrete ways of thinking", such as the *bricolage* that Turkle and Papert [39] advocated, is now more legitimate within the computing education community. In fact, part of being "in the game", we hope, involves challenging its rules. But in order to do so, students should be immersed in the community wisdom, its discourse, and aim at improving it, thereby expressing themselves within their community.

As Tomasello explains: "Communicative conventions thus come to be governed by constitutive norms in the sense that if I do not use them in the conventional way, I am just not in the game. As Wittgenstein (1995) argued so trenchantly, the criteria for conventional use are determined not by the individual but by the community of users. I can rebel, but to what effect? … Anyone not comprehending this communicative convention is just not one of us—which makes it culturally normative" [32].

Though we sympathize that entering new cultural communities, with cultural practices that might be different than what one is familiar with—including discourse practices—may present considerable challenges, we nonetheless believe that lamenting the fact that such conventionalized discourse practices exist and wishing them away is to overlook the fact that cultural practices (including communicative ones) become normative [3, 31].

The debate on genre conventions versus. freedom of expression is timely today, too, given that we want to teach K-12 students computational thinking and that there are a variety of languages today available for their use (e.g., Scratch). What genres emerge as students interact with these languages in the classrooms and outside? What aspects of computational thinking—what discourse genres—are expressed? Further research is required to address these question, future dialogue to discuss the direction(s) this community takes.

8. CONCLUSION

This paper presents the argument that viewing computer programs as linguistic acts, as *utterances* that function with human communities, opens a space for empirical research and theorizing related to the learning of computer programming, surely one of the enduring concerns of computer science education. As utterances, computer programs not only *express* computational ideas that are interpreted mechanically by a digital computer in order to control its operation, but at the same time *communicate* human intentions about the way that the computer will effect change in the ontic world to other human beings. As a result of the human recipient, expert-written computer programs embed dialogicality, in that they are addressable to other programmers who will have to read the program, heteroglossic, in that they reference computer programs written previously by others, and written in conventionalized programming genres specific to programming communities. This dialogicality takes for granted but depends upon uniquely human forms of intentionality, not only involving goals and plans, but *joint intentions* and socially recursive inference within conventionalized and normative programming genres. As a result, we posit that there are two primary ways in which this dialogical perspective can serve as a basis for an empirical research program. On the one hand, because of past language use that novices have with natural language and the relative ease with which people impute intentionality to others, there are a number of research questions related to the intentional stances that novices might take toward the computer that they are attempting to program, and learning about what these stances are is worth understanding. On the other hand, novices may either be unaware of the communicative function of the computer programs that they write with respect to human recipients, or more likely, unaware of their normative obligations to shape their programming utterances so as to facilitate the understanding of their intentions by a program reader from within their programming community, much as they already do so in natural language. Undertaking research to better understand novice intentional stances can then serve as a basis for teaching that is explicitly designed to foster programmer dialogicality.

9. ACKNOWLEDGEMENTS

Initial discussion between the authors started at a workshop supported by the National Science Foundation under Grant No. 1332686. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. We are grateful to Steve Cooper for inviting us to participate in this workshop.

10. REFERENCES

[1] Bakhtin, M. 1986. *Speech genres and other late essays*. University of Texas Press.

[2] Bakhtin, M. 1981. *The dialogic imagination: Four essays by M. M. Bakhtin*. University of Texas Press.

[3] Ben-David Kolikant, Y. 2011. Computer science education as a cultural encounter: a socio-cultural framework for articulating teaching difficulties. *Instructional Science*. 39, (2011), 543–559.

[4] Ben-David Kolikant, Y. and Pollack, S. 2004. Establishing Computer Science Professional Norms Among High-School Students. *Computer Science Education*. 14, 1 (2004), 21–35.

[5] Boehm, C. 2012. *Moral Origins: The Evolution of Virtue, Altruism, and Shame*. Basic Books.

[6] Bowles, S. and Gintis, H. 2011. *A cooperative species: human reciprocity and its evolution*. Princeton University Press.

[7] Chomsky, N. 1965. *Aspects of the Theory of Syntax*. MIT Press.

[8] Cicourel, A. V. 1974. *Cognitive sociology: Language and meaning in social interaction.* Free Press.

[9] Clark, H.H. 1993. *Arenas of language use*. University of Chicago Press.

[10] Clark, H.H. 1996. *Using language*. Cambridge University Press.

[11] Clark, H.H. and Brennan, S.E. 1991. Grounding in communication. *Perspectives on socially shared cognition*. 13, 1991 (1991), 127–149.

[12] Danielak, B. 2014. *How electrical engineering students design computer programs*. University of Maryland, College Park.

[13] Dennett, D. 1987. *The Intentional Stance*. MIT Press.

[14] Dietrich, E. ed. 1994. *Thinking Computers and Virtual Persons: Essays on the Intentionality of Machines*. Academic Press.

[15] Dreyfus, H. 1991. *Being-in-the-World: A Commentary on Heidegger's Being and Time, Division I*. MIT Press.

[16] Garfinkel, H. 1967. *Studies in ethnomethodology*. Prentice Hall.

[17] Grice, P. 1975. Logic and conversation. *Syntax and Semantics, 3: Speech Acts*. P. Cole and J. Morgan, eds. Academic Press.

[18] Lave, J. 1988. *Cognition in Practice: Mind, Mathematics, and Culture in Everyday Life*. Cambridge University Press.

[19] Lewis, C.M. 2012. *Applications of out-of-domain knowledge in students' reasoning about computer program state*. University of California at Berkeley.

[20] Lieberman, D.E. 2013. *The Story of the Human Body: Evolution, Health, and Disease*. Pantheon Books.

[21] Linn, M.C. and Clancy, M.J. 1992. The case for case studies of programming problems. *Communications of the ACM*. 35, 3 (Mar. 1992), 121–132.

[22] Meltzoff, A.N. 2011. Social cognition and the origins of imitation, empathy, and theory of mind. *The Wiley-Blackwell handbook of childhood cognitive development*. U. Goswami, ed. Wiley-Blackwell. 49–75.

[23] Nowak, M.A. and Sigmund, K. 2005. Evolution of indirect reciprocity. *Nature*. 437, (2005), 1291–1298.

[24] Pea, R.D. 1986. Language-independent conceptual "bugs" in novice programming. *Journal of Educational Computing Research*. 2, 1 (1986), 25–36.

[25] Petre, M. 1995. Why looking isn't always seeing: readership skills and graphical programming. *Communications of the ACM*. 38, 6 (Jun. 1995), 33–44.

[26] Rogoff, B. 2003. *The Cultural Nature of Human Development*. Oxford University Press.

[27] Sacks, H. et al. 1974. A simplest systematics for the organization of turn-taking for conversation. *Language*. 50, 4 (1974), 696–735.

[28] Searle, J. 1995. *The Construction of Social Reality*. The Free Press.

[29] Soloway, E. and Ehrlich, K. 1984. Empirical studies of programming knowledge. *IEEE Transactions on Software Engineering*. SE-10, 595-609 (1984).

[30] Tenenberg, J. 2001. On the meaning of computer programs. *Cognitive Technology: Instruments of Mind, Proceedings of the 4th International Conference of Cognitive Technology* (Coventry, UK, 2001).

[31] Tenenberg, J. and Knobelsdorf, M. 2014. Out of our minds: a review of sociocultural cognition theory. *Computer Science Education*. (2014).

[32] Tomasello, M. 2014. *A Natural History of Human Thinking*. Harvard University Press.

[33] Tomasello, M. et al. 1993. Cultural Learning. *Behavioral and Brain Sciences*. 16, (1993), 495–552.

[34] Tomasello, M. 2011. Human culture in evolutionary perspective. *Advances in culture and psychology*. M.J. Gelfand et al., eds. Oxford University Press.

[35] Tomasello, M. et al. 2012. Two Key Steps in the Evolution of Human Cooperation: The Interdependence Hypothesis. *Current Anthropology*. 53, 6 (2012), 673–692.

[36] Tomasello, M. et al. 2005. Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and Brain Sciences*. 28, (2005), 675–691.

[37] Tomasello, M. 2009. *Why we cooperate*. MIT Press.

[38] Tomasello, M. and Vaish, A. 2013. Origins of Human Cooperation and Morality. *Annual review of psychology*. 64, (2013), 231–55.

[39] Turkle, S. and Papert, S. 1991. Epistemological Pluralism and the Revaluation of the Concrete. *Constructionism*. Ablex Publishing Company.

[40] Wertsch, J. 1993. *Voices of the Mind: A Sociocultural Approach to Mediated Action*. Harvard University Press.