# How CS Teachers Change? The Story from Teachers

Lijun Ni
School of Interactive Computing
Georgia Institute of Technology
801 Atlantic Drive
Atlanta, GA, 30332
lijun@cc.gatech.edu

Tom McKlin
CEISMC
Georgia Institute of Technology
760 Spring St. NW
Atlanta GA 30332
tom.mcklin@gatech.edu

## ABSTRACT
In this paper, we present the results of an interview study revealing how Computer Science (CS) teachers create change. We interviewed eight teachers about a year after they attended our workshops on several innovative introductory CS courses. The interview was designed to elicit the extent to which CS teachers have adopted or adapted what they learned from the workshops, and what drives or prevents their efforts to make change. The results of this study revealed that the adoption, adaptation and implementation of CS curriculum innovations in new contexts involve systemic change affecting teachers, departments and institutions as a whole. The findings of this study reveal a list of questions that a CS instructor might ask before committing to any new innovation. Our findings further suggest several recommendations directed towards more effective dissemination of computing education innovations.

## Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education–*computer science education.*

## General Terms
Design, Experimentation, Theory.

## Keywords
CS Teacher, Adoption, Change, Professional Development.

## 1. INTRODUCTION
In recent years, one major challenge the CS education community facing is the declining enrollment and interest in CS [12]. Responding to the current challenge, researchers have been contributing great efforts to this field, with a variety of significant innovations invented, such as new curricula [3], and creative programming languages and environments [1]. To make those innovations or best practices emerging from CS education research have real impact on teaching practices (and thereby on student learning), we eventually need CS teachers to bring those innovations and integrate them into their own classrooms. Therefore, it is critical to understand how teachers actually adopt, adapt and implement computing education innovations.

One major channel to disseminate educational innovations is through offering Professional Development (PD) opportunities for teachers, such as workshops, conference sessions and other training opportunities. Related research indicates that faculty development participation has strong, statistically significant impact on faculty's adoption of innovative teaching practices [9]. However, participating in PD related to an innovation does not necessarily lead to teachers' actual adoption. The actual impact of PD is usually *diluted* by all kinds of other factors that support or hinder teachers from making change [8, 10]. A variety of variables have been identified as those factors beyond PD, such as teachers' motivation for professional development, teachers' self-efficacy, and their access to preparation time [10]. For example, teachers with a strong need to learn an educational innovation usually demonstrate more change in their knowledge and action after participating in PD activities [11].

In this paper, we attempt to understand how a CS teacher creates change. In other words, we wonder how a CS teacher actually adopts a computing education innovation into his/her local context after attending PD activities related to that innovation. This paper is aimed at starting to explore this question through contacting CS teachers.

We interviewed eight teachers about a year after they attended our workshops on several innovative introductory CS courses. This interview was designed to elicit the extent to which CS teachers have adopted or adapted what they learned from the workshops. Furthermore, we attempted to understand what contributes to teachers' actual decision on whether to make change within local contexts. The results of the interview can provide researchers and developers of computing curriculum innovations as well as other related stakeholders with guidance in the successful dissemination of computing education innovations.

In the following section, we first describe the interview study we conducted to explore how CS teachers make change. Then, we present the results from the study and discuss the findings in the end of this paper.

## 2. INTERVIEW STUDY
From April to May, 2008, one author conducted semi-structured interviews with eight CS faculty members who had attended at least one summer workshop on several innovative introductory CS courses for undergraduates.

## 2.1 Interview Population

The teachers we interviewed attended at least one of the three workshops offered in Summer 2007. These workshops introduced several contextualized computing courses offered for undergraduates to encourage diversity and to improve the enrollment and retention of students in CS [4, 5]. These contextualized courses included Introduction to Media Computation in Java or Python (both CS1), Media Computation Data Structures in Java (CS2), Engineering in MATLAB (CS1), and Robotics in Python (CS1). These courses emphasize the use of a specific context or theme (e.g. media computation or robotics) throughout the whole course, which students recognize as being authentic and relevant for computing [5].

Interview participants were selected based on a previous survey administered in the Fall semester after the summer workshops to collect workshop participants' adoption decisions. We identified those likely to adopt any of those courses introduced in the workshops and those unlikely to adopt based on the teachers' responses. Four *adopter* and four *non-adopters* were interviewed at the end of Spring 2008 semester, about a year after they attended the workshops.

## 2.2 Interview Questions

The researchers used a semi-structured interview protocol designed to elicit the extent to which the CS teachers had adopted or adapted the contextualized computing courses. The interview protocol is based on Guskey's [2] five levels of professional learning which mostly parallels Kirkpatrick's[7] four levels of evaluating training programs. Specifically, the participants were asked what content and/or pedagogical techniques they recalled from the workshop and how they apply those techniques. They were then asked what action they had taken as a result of their workshop participation, whether their participation resulted in organizational changes to their department, curriculum, instruction, etc. They were also asked to describe any student success they had observed related to using the innovation. Finally, they were also asked about their plans to adopt or adapt any of the new courses.

## 2.3 Analysis

Each interview was recorded and transcribed. We performed a three-step process to increase coding reliability and fidelity. We first coded four separate interview transcripts each. From that, we created a code book, a list of codes with a definition and an example of each code. One of us then went back to the transcripts and coded all eight transcripts. Afterward, the other author edited the first researcher's coding categories. During this stage, we discussed the discrepancies until we had agreed upon each code/quote combination.

## 3. RESULTS/FINDINGS

In this section we outline how the participants actually made or did not make any changes. We first summarize participants' reasons for considering curricular and pedagogical changes to teaching CS. We then broadly describe their most pressing concern: making the innovation fit and work in a CS department with multiple competing needs. Finally, we derive a set of questions from the interview transcripts that teachers contemplating a computing education innovation may consider.

## 3.1 Reasons for Considering Change

The major motivation for teachers to look for curriculum innovations comes from a need to change the current situation: lackluster student motivation, steadily declining enrollment and retention of students in CS, and shrinking participation among women and under-represented minorities. Below we asked teachers what drives their intent to adopt an innovation.

> "Enrollment really dropped. That is one reason why we tried to look at other ways and make changes in the intro classes."

> "We have a retention rate of like 30%, and in our department, it was like 10%, so there are serious problems there. We had to think about what was wrong and the main problem was the intro class… so we tried to make it more interesting to get them to stay in the program."

> "[T]he enrollment of women in computer science has fallen off precipitously, and that started 20 years ago…. I [have]… about one [woman] in one class. I've got three in another class. And that's not very many."

## 3.2 Adoption Status & Organizational Change

While the teachers we interviewed were motivated to make changes and interested in adopting the innovation, all faced roadblocks to making the changes occur. The extent and nature of the roadblocks *determined* the status of their adoption.

- Currently, one participant can be described as a full adopter. He has adopted one Introductory to Media Computation course *successfully* for two semesters, and then his department has decided to adopt it in the whole department in the coming Fall semester.
- Another teacher tried one Media Computation course last Fall. He has decided to teach it again this coming fall when will be his turn to teach the CS1 course in his department.
- Another teacher has chosen to adopt one specific element of the workshop, a new IDE—Dr.Java into his own CS1 course. And now he and his colleagues who also taught introductory programming course in Java are using this new IDE.
- One participant is hoping to adopt in the near future. She is waiting for a textbook for the new Media Computation Data Structures course to be available.
- The remaining four teachers has expressed interest in adopting in the future; none has said that they has given up hope of adopting the innovation altogether.

Organizational change is operationalized as a comment that describes the process of getting members of the CS department to do or support an innovation. Most of the conversations on the topic of organizational change occurred among the non-adopters, and while many described the flexibility they enjoy in changing their own teaching strategies, a majority of these comments focus on the difficulty of getting one other person in the department to change in order to introduce both the content and teaching strategies inherent in the innovation.

> "[There] are four of us that are willing to try things and there is one of us that absolutely would not."

> "I think [the innovation] has possibilities for our CS0 course, but we haven't acted on that yet because the person who is in charge of that course did not leave which was what we were anticipating."

While many claimed that the root of their adoption issues lied in getting one other person to change, the real issue might not lie with that one person but with the organizational system supporting that one person's behavior. The difficulties inherent in organizational change are revealed in our analysis through the code, "Barrier to change." This coding category describes any barrier that prevents the original adopter from introducing an innovation to a CS department. Interestingly, this coding category was used more often than any other coding category.

### *3.2.1 Barriers to Change*

The implementation of an innovation will face numerous barriers which can either slow down or completely stall a departmental change despite the overwhelming need to try something different. Below, we outline some of the common barriers from the transcripts.

- **Little Freedom for Content Change**

Many respondents said that they enjoyed the freedom to teach CS content in whatever way they wish provided that the material was covered and the students were well-prepared for the next level. However, that freedom did not extend to changing the curriculum or course content.

> "There are other faculty [members] teaching the course, so I have to use the course that is already developed."

> "I think that is across the board for the department, so there are certain things that we have to have, and we might have the same labs, but other than that, lectures can completely be different but making sure we cover the same types of material and the same material."

- **Demand for Faculty Development**

A curricular/pedagogical innovation competes with other faculty demands. In the following example, one participant expressed that the innovation required training other faculty, a commitment that might over-burden stressed faculty members.

> "Some of the other issues that are important are faculty development. I have been to these workshops. I was familiar with the [innovation] before I went, so this was just fleshing out the skeleton I already had to work from. Um, how are we going to do the faculty development? We made some changes before, when we moved to an object-oriented approach to teaching our first course…and that was a lengthy transition."

> "The entire faculty is really overloaded with trying to meet teaching loads, advising loads and committee loads…So for the younger faculty members that is going to be a real issue."

- **Poor Ability and Background of Students**

Teachers might perceive that the change simply will not work within their own context. One way that manifests itself is within students' ability and background. Some participants perceived that the students at the innovating institute might more easily handle the innovation.

> "This is too difficult for our kind of students. It might work well for students at [the innovating university], but I think our students are a little different. I don't think I would be able to incorporate that."

> "I think it's probably not a good idea for our kind of students, at least for the data structures class… Usually they are weak in foundations even if they went to CS0, CS1, CS2. When they come in my class I mean, some of them have a hard time to write even simple programs or reading…One really cannot expect that they have any major math skills."

- **Different Approaches (Innovation vs. Tradition)**

One participant explicitly questioned the innovation because it strays from traditional approach to teaching CS. This participant makes the case that there is not only disagreement among faculty but that a non-traditional approach would attract the *wrong* students and possibly set them up for failure later in the major.

> "Generally some people want to push for less programming and others want to keep the status quo. There is a lot of programming in [our CS] program, so there is… disagreement in which way to go."

> "[T]here are other pieces to computer science. There is a lot of it that is related to applied mathematics. There's computer theory, assembly language and machine organization…. In fact the [content]… is going to be rigorous and very formal, …[requiring]… analytical skills that some people don't want to bother [with]. They don't want to sit down and focus like that. So I am not sure if it would attract the right people for the right reasons."

- **Near and Far Ripple Effect of Change**

The participants also mentioned that any sort of change will send out cascading and possibly unexpected changes to both the courses in which the innovation is adopted and courses later in the sequence. Participant response drives us to think about what we currently value in a course, and the affect of the innovation on subsequent courses.

> "It's not just the [innovation], it's any sort of change. The [innovation] really requires a rethinking of two things: 1) how do we teach fundamentally the same material? 2) what's important? What are the really important concepts here? [If we change, it means that] some of what we used to do, we no longer do, and some of what we are doing now we probably will be deemphasizing if we take the [innovation] approach. Is that good? Is that bad? It takes discussions and different faculty will react differently to this."

> "[I]t does have an effect not only in the beginning courses…. What can we assume that the students know in subsequent courses? So there is a ripple effect throughout several courses in the curriculum."

### *3.2.2 Accelerators to Change*

Just as there are barriers to adopting an innovation, we also see systemic conditions that not only enable the innovation but that increase the rate at which that innovation may be adopted. We call the latter "accelerators."

- **Sense of Urgency**

One accelerator is a strong sense of urgency, and many of these were outlined in the section above "Reasons for Considering Change." Many participants are troubled by the declining enrollments, low retention rates in CS, and homogenous student body. Some perceive this as a reason enough to try an innovation whereas others may not be as affected by these trends and may be less likely to adopt. One participant sums it up by saying, "I think what we're finding here is that having a real traditional CS curriculum is not working."

- **Perceived Benefits for Students**

The adopters among our interview group believed that motivating students is essential for increased enrollment and retention, and the curriculum innovation would help to motivate the students.

> "I think they would enjoy [the innovation]…, and I think it would be valuable for them. [It] would help them understand computing a little bit, and it would be an interesting introduction for that purpose. Students seem to be more interested because they have a sense of ownership to the objects they are manipulating."

- **Successful Experiment**

One participant conducted an experiment comparing two courses, one using the innovation and the other taught in a traditional way. The *successful* initial adoption convinced him to further adopt this innovative course and even drives a full adoption in the whole department.

> "We did an experiment this year, I was teaching the course using [the new approach] and … my colleague was teaching the course in the way we've always been teaching it, just the traditional way of teaching computer programming. [And] we've discovered that the students seem to be responding better and they are more excited about the [new approach]…We've discovered that the performance [of students taking the innovative CS1 course] in the next course [CS2] is on a par with the students who are coming out of the other courses. The plus side of all this is we've got more students finishing the [innovative CS1] course successfully with an A, B, or C than in the traditional course…So I think I have won over my colleagues and we are going to be, everybody in the department will be teaching it using [the new approach] in the fall."

- **Recommending Pedagogy**

The participants mentioned earlier that they usually did not have the flexibility to change course content, which prevent their adoption of the curriculum innovation. The contextualized courses include the pedagogy of contextualization, suggesting particular ways of presenting CS concepts, or student assignments to complete, which seems to bring some kind of *restrictions* for the teachers. However, the teachers positively perceived the *restrictions* from this innovation.

> "Actually I guess it's probably best articulated by one of my colleagues who teaches the beginning programming course. 'Traditionally we've been able to teach what we want, just in the constraints of language that we're using. Now you are telling us how we are going to teach the course.' Ands that is a shift in the way we have done things. But he was convinced this was the right thing to do."

> "When we switched from C++ to Java a couple years back, there was no discussion about how to teach the course, just here is a different language to teach it… So the faculty members still had the luxury of picking their own assignments, picking the order of the topics that they were doing, pretty much designing their own approach to teaching the course. With [the innovation] there is a little bit more constraint…. So there is a little bit more focus on the types of assignments that the students are going to be given even though there is latitude in picking the exact assignments. "

Teachers might enjoy the freedom to choose the way to teach as long as the material is covered. However, under this situation, they may be inclined to focus more on covering the required material than on determining whether the students have learned the material. Otherwise, when the innovation brings corresponding pedagogy as well, teachers would likely implement a wholly change involving pedagogical innovation.

## 3.3 Critical Issues in Making Change

We derived a set of questions from the interview transcripts that a potential adopter may ask consider before committing to an innovation. These questions were harvested from all interviews and reflect questions that were most asked while considering adopting media computation. These questions are:

- What's really the important material for the course, and how will the innovation highlight it?
- What actual changes will I have to make to implement the innovation?
- How will we ensure that students learn the same material?
- How might current and incoming students respond to the innovation?
- How will the innovation help us maintain quality?
- How will the innovation motivate students to enroll and persist in the major?
- How will the innovation assure us that students are well-prepared for upper-level courses?
- How will this change affect other stakeholders?

Some of those stakeholders could be lab instructors, teaching assistants, colleagues teaching the same course, upper-level instructors, and instructors in other departments participating in a joined program. For example, a CS teacher might consider how the change might influence upper-level instructors: How will this curricular or pedagogical change prepare students for upper-level courses? Will upper-level instructors have to cover material that is not learned earlier? Will they have to alter their teaching strategies? Or considering the lab instructor, the teacher might ask: will the change affect what happens in the labs?

These questions indicate different types and stages of concerns from teachers considering and experiencing change: *personal concerns* about how change will affect them, t*ask concerns* about how to manage new practice, and finally *impact concerns* about how the new practice will affect students and other stakeholders [6, 10]. In general, early concerns are more self-oriented, e.g. what the innovation is? How will it affect me? Our participants reported mainly the task concerns and impact concerns. One possible explanation might be: participating related PD helped to solve those early concerns. In other words, the teachers gained some knowledge about the innovation from the workshop which helped to answer their early concerns.

## 4. DISCUSSION AND CONCLUSIONS

This interview study has provided us an initial source of data to explore our research question: How do CS teachers actually create change? First, the results of this study indicate that the adoption, adaptation and implementation of an innovative curriculum in new contexts is not just one individual teacher's business. Adopting a contextualized computing course is a major change involving systemic change affecting teachers, departments and sometimes other departments or even the institution as a whole.

Second, as we presented barriers and accelerators to change, we have seen multiple competing needs influencing how teachers actually make a computing education innovation fit and work in a CS department. In other words, the results of this study indicate the complexity of implementing change in new contexts. Although the teachers we studied wanted to create change, and had attended useful PD activities, they still experienced a variety of challenges when considering actual adoption.

Meanwhile, the findings of this study also lead to a few recommendations directed towards more effective dissemination of computing education innovations. Particularly, this paper is intended for three kinds of audiences: CS education researchers and curriculum developer who develop innovative curricula, tools and theories for CS education; PD developers and facilitators who design and organize PD activities to disseminate innovations; and policy-makers leading either curricular or pedagogical changes in their CS departments.

- **Suggestions for Curriculum Developers/ Researchers**

First, a new computing curriculum should cover the majority of current course content, and easily mesh with lower and upper-level courses. Furthermore, a new curriculum might be more likely to succeed if it were also accompanied by pedagogical recommendations and examples.

- **Suggestions For PD Developers and Facilitators**

Professional developers and facilitators need to understand what concerns PD participants face, and try to address those issues (example questions are listed in section 3.3). Particularly, when designing PD activities, PD developers and facilitators should consider how the PD could help motivate teachers to change and solve early stage concerns.

- **Suggestions for Policy-makers**

Department chairs and other policy-makers might use their unique roles to actively facilitate change where it's needed and offer support for instructors to experiment with CS education innovations. One strategy recommended is to send the *right* people who would be able to enact change into PD. Examples of the *right* people are teachers who are interested in a new curriculum and willing to lead the adoption, or those teachers responsible for the same level courses in a department.

Moreover, our findings of this study to some extent suggest potential research questions that require further understanding. For example, those critical issues/questions in section 3.3 actually draw out a handful of interesting research questions to explore for each specific computing education innovation. Meanwhile, longer term study tracking on teachers' change process would help us further understand how CS teachers create change, and thereby offer more insights for related stakeholders to better disseminate innovations from the CS education community.

## 5. ACKNOWLEDGMENTS

This research is supported by grants from the NSF #0618674 and #0634629. We would like to thank all of the teachers who attended our workshops. Also thanks to GaComputes! group and the CSL lab for their comments on earlier drafts of this paper.

## 6. REFERENCES

[1] Dann;, W.P., Cooper;, S. and Pausch;, R. L*earning To Program with Alice*. Prentice Hall, 2008.

[2] Guskey, T.R. and Sparks, D. *Evaluating Professional Development*. Corwin Press, 2000.

[3] Guzdial, M. *Introduction to Computing and Programming in Python: A Multimedia Approach*. Prentice-Hall, 2004.

[4] Guzdial, M., & Forte, A. Design Process for a Non-majors Computing Course. *Proceedings of SIGCSE 2005*, 361 -365.

[5] Guzdial, M. and Tew, A.E. Imagineering Inauthentic Legitimate peripheral participation: an instructional design approach for motivating computing education. Proceedings of ICER '06, ACM Press, 2006, 51-58.

[6] Hall, G.E. and Hord, S.M. *Implementing Change: Patterns, Principles, and Potholes*. Allyn and Bacon, Boston, 2001.

[7] Kirkpatrick, D.L. Another look at evaluating training programs, Alexandria, VA: American Society for Training & Development, 1998.

[8] Kubitskey, B. and Fishman, B.J. Untangling the relationship(s) between professional development, practice, student learning and teacher learning. Annual Meeting of the AERA, Montreal, Canada, 2005.

[9] Matney, M.M. Institutional and Departmental Factors Influencing Faculty Adoption of Innovative Teaching Practice. PhD dissertation. University of Michigan, 2001.

[10] Smith, C. and Gillespie, M. Research on Professional Development and Teacher Change: Implications for Adult Basic Education. NCSALL, 2007, 226-234.

[11] Smith, C., Hofer, J., Gillespie, M., Solomon, M. and Rowe, K. How Teachers Change: A Study of Professional Development in Adult Education, National Center for the Study of Adult Learning and Literacy, Boston, 2003.

[12] Vegso, J. Interest in CS as a Major Drops among Incoming Freshmen. *Computing Research News*, 17 (3).