Increasing Female Students Declaring a Computer Science Major from 11% to 46% via Pair Programming

Host: Donna Milgram, Executive Director, Institute for Women in Trades, Technology & Science

Presenter: Dr. Charlie McDowell, Ph.D., Professor of Computer Science, University of California, Santa Cruz

Interview Transcript:

Donna: Hello and welcome. My name is Donna Milgram, Executive Director of the Institute for Women in Trades, Technology and Science. And I'm so excited to welcome you to this session of the *STEM Success for Women Telesummit*, funded by the National Science Foundation.

We have an interview with a very special guest. And then at the end of the session, you'll have an opportunity to ask our guest questions.

Our guest today is Dr. Charlie McDowell, professor of computer science at the University of California, Santa Cruz. Dr. McDowell has dramatically increased retention of female and male students in computer science courses using pair programming.

He also organizes the annual Bay Area Aspirations Award, which recognizes 50 technical high school girls with a gala event at the San Jose Tech Museum. He's established a $16,000 scholarship for women in computer science at UC Santa Cruz and facilitated many other California schools following suit. Nationally, he co-chairs the Academic Alliance of NCWIT.

Welcome, Charlie, and thank you for joining me for the STEM Success for Women Telesummit.

Charlie: Thank you, Donna. I'm happy to be here and share.

Donna: Although we have never met, I really feel like I know you because I have been talking about your work and pair programming for many years. It's an important part of our WomenTech Educators Training. I can't tell you how many educators, after hearing your statistics and the difference that it made, plus you have a control group, and there are very few studies that have control groups, and you have a large "n" of 502 – they have ended up deciding to do pair programming.

I have to tell you, it's such a privilege to have you as part of our Telesummit. Maybe you could start out by just talking a little bit about what pair programming is for those who are not familiar with it. And then maybe talk about your numbers, your study and its impact.

So I'll let you start by defining pair programming.

Charlie: Thanks, Donna, and thanks for sharing the word about pair programming with all the people that you talk to. I'd like to start by saying I didn't do this alone. In fact, the initial impetus for the work that we're talking about today came from my colleague, Linda Werner.

The initial study was her idea. I was just this guy teaching a big introductory programming class that she wanted to study and it was her that brought us together. We brought on two psychologists from UC Santa Cruz. I think it was all of that together that helped make a really credible study.

Pair programming originated as a practice in industry as part of an agile development process called extreme programming. It is two people working together at a single computer – one mouse, one keyboard. The person holding the mouse and the keyboard is usually referred to as the driver. The other person is referred to as the navigator.

The navigator can suggest directions for action, catch errors, ask questions. There's really no constraint other than that they are both fully engaged in a conversation about what they're doing. They maintain shared ownership and equal contribution, both in the professional world and when we use it in education. It's important that the partners switch roles periodically, typically every 15 to 20 minutes, but it could be more frequent.

Although we believe all types of project-based collaboration work well in this environment, it's important to realize that pair programming is not divide-and-conquer. There's nothing wrong with divide-and-conquer – two people working on a problem — but that's not pair programming.

In divide-and-conquer, one person's working on one part alone. The other person's working on another part alone. And then they put those parts together. In pair programming, two people are working together at the same time. That togetherness I think is really important in what makes pair programming special and part of the impact I think that it has.

Donna: Let's go back before I have you reveal your numbers – and I think they're actually a little bit larger than I mentioned. It's sort of fitting that you did this with other people because it's sort of the principle of working together.

So the professor who you mentioned wanted to study your class, what made her come up with this idea? You said it was done in the industry. Had she seen it done in the industry and she wanted to apply it? What was the impetus in the first place?

Charlie: Yeah, she heard about pair programming at about the same time she read a report from the American Association of University Women that hypothesized some reasons why they thought fewer women were going to be in computer science. Those reasons were that women felt a career in computing was not well-rounded and conducive to family life. They felt that working in this field is competitive and not collaborative, that it's a solitary kind of activity.

In particular, women students had concerns, personal safety concerns, about working in a lab late at night by themselves. She saw those two things together. Here's pair programming; here's some stuff that's going on. Gee, maybe we could use pair programming and have it addressed.

It addresses the solidarity; the competitive nature is somewhat diminished and it becomes collaborative. If you're meeting, if you're going to working in a lab late at night, you have somebody else there. Three of those four things got hit by pair programming. I think it was those things together that caused her to say, "Hey, let's see what happens if we use pair programming and see if it will make the students want to stick around."

Donna: Interesting. I am going to guess that that study was probably AAUW's landmark *Tech-Savvy: Educating Girls in the Computer Age* study. What year was this about, more or less?

Charlie: This is in 2000.

Donna: Yep, that was right after that study had come out. So she was actually looking at this as a potential remedy for those things that made computer science unattractive that were described in the study.

Charlie: Exactly.

Donna: That is so interesting. So tell us what happened in some detail and how you ended up applying this pair programming to your big introductory computer science lecture course. Talk about how many students, etc.

Charlie: Well, let's see, so she decided that we should do this study. We got together with a psychologist and we really wanted to try to control as many factors as we could. We ended up using three classes. I was teaching two of them. Another person was teaching the third section. Some of the sections we were using pair programming and then in another one, we made the students work independently.

I think, boy, the numbers – this was so long ago. It was something over 500 students that we had in the total study. We looked at a number of factors. Initially, the sort of key things that we were looking at were how well did they do? We gave them a lot of questions about did they like what they were doing? Were they confident in their solution? We looked at the programs they produced. We looked at their scores on the exams. And we did exactly the same thing for a large class with students pairing, a large class with students working individually.

Donna: It had a control group.

Charlie: Yes. And actually, so we had some very strong positive initial results from that, right from the beginning, in terms of retention, because we looked to see if they enrolled in the next class.

So we want to know after they finished this class did they decide, "I don't want any more of that," or did they go into the next class? We also then continued to follow these students for a year. Then we looked a year later to find out if they had decided to declare a computer science major.

Donna: Well, what did you find?

Charlie: As I think you alluded to in some of your advertising for this event, it was quite dramatic. In fact, we were rather surprised. I think one of the big numbers that sort of jumps out at you is when we look at the percentage of students that were still at UCSC one year later. If they decided to leave the university, we didn't count them. Of those students that took the classes were still at UCSC one year later, and found out how many of them had declared a computer science major.

From the non-pairing class, 11% of the women that were in the non-pairing class had declared a computer science major; 46% of the women from the pairing class had declared a computer science major.

Donna: That's huge.

Charlie: We saw similar results for those continuing onto the next class. It was 44% of the women went onto the next class in the solo section, and 68% went on from the pairing class.

Those numbers seemed really strong, so we said, "Well, wait a minute, maybe it's just the number of computer science majors." So even when we restrict that – because this introductory class has lots of students from non-computer science majors – when we restricted just to those students that at the time they took the introductory course were listed as proposed computer science major. At Santa Cruz, they haven't necessarily declared their major at that time.

So if they were proposed computer science majors when they take the class, 22% of the women in the solo class went on and declared, i.e., they turned that proposed into declared; 66% of the women that worked in pairs that were proposed went on and declared.

Donna: Huge if they were paired.

Charlie: Right.

Donna: If they declared, they had declared in advance that they were considering CS. What about if they weren't? Because I remember that the females in the first year, your two introductory classes, also persisted at a much higher rate and made it through those introductory courses if they were paired versus if they weren't.

And I would guess that you may have actually got some of the females who were not planning on pursuing CS as a major to consider it as a major. Let's see if I'm right.

Charlie: Certainly there were women that were not considering computer science as a major at the time they took the class but then went on. That was more in the pairing class, although I don't have that number at my fingertips. If we look at the data from our main communications with the ACM article, we could certainly derive from that, "Gee, what percentage of the students? Let's see, there must've been that many that were not declared." I don't have that at my fingertips.

Donna: Okay, okay. So once you had identified this, did you start looking at the male data as well and what did the male data tell you about how pairing worked for them?

Charlie: Certainly. We certainly looked at all of the students. In fact, pairing helps everybody. So this was a good thing. It would have been problematic if it had somehow been contrary to male interest.

So everybody's went up. But we like to describe it as closing one gender gap, because the women benefitted proportionately more than the men did so that, in terms of any of factors, they get closer together.

Certainly in the case of men, to compare one of the numbers I gave you, I had said that for computer science majors that they had proposed it at the time and they were still there one year later. For the women, it went from 22% to 60%. For the men, it went from 47% to 74%. So there was still an increase, but the increase was not as much as it was for the women.

Consequently, the percentage of students that persisted was closer in the pairing sections than it was in the solo sections.

Donna: Yes. By the way, we got permission from UC Santa Cruz and from yourself and the other authors to include your study in our Proven Practices collection on our website. If our listeners want to see the study, you have tons of numbers and details in there and you have ends and percentages. It's called *Pair Programming Improves Student Retention, Confidence, and Program Quality*. You can just do a search for it in the retention portion of our Proven Practices collection.

So I want to go back and talk in some more detail about the pair programming itself. You described it as working collaboratively and that the students would switch off on the driver and the navigator technique.

The research shows when you pair male and female students together in a lab setting - any kind of lab setting - that the males will end up taking over the equipment and the females will take notes. So I'm wondering if this came up in the pair programming and if you had to structure it more, if you paired females only with females. Talk a little bit about this dynamic and the pair programming aspect, how it relates.

Charlie: It's interesting you shared that. It really made me smile. My colleague Linda Werner often talks about this - I think it's the same result you're referring to. I'm sorry I can't quote it. I don't remember, and maybe you can even fill that in. It was with young students. I don't remember for sure, somewhere in the K-12 group.

They studied boys and girls. And girls have what's called a give protocol. That is, in pair programming, the girls would give the mouse and control to the other girl. And the boys have what's called the take protocol. So the boys said, "Give me the mouse," and they would take it away. Consequently, the boy/girl pair-up, as you pointed out, is not a great thing to do.

In fact, in my classes I now — I didn't do this at the time — but for the last several years, I preferentially pair women with women and men with men. Recently, I got some flak for that from one student, a female student, who in my end-of-quarter evaluations complained and I pointed out that I was observed that I was doing this, and described me in kind of strong terms as being sexist and it was sexist activity, which really surprised me.

I've actually changed. I've modified that slightly, which is that I now make a point of announcing at the beginning of my class that this is what I do, that when I assign partners, I will preferentially sign women with women and men with men. And if any of the women object to that, they can simply let me know and I will put them in the group when pairing the men.

I've only had the chance to do this once. This was relatively recently. I did it once. It was a large class. I had zero takers, so I actually think it's okay and they mostly like it.

Donna: Yeah, and it's so interesting because yesterday, I started off our Telesummit with a session on how to ensure female students are successful in the lab. I talked about this dynamic. By the way, I don't know that study and I would love if you could put me in touch with the person who would know. I would love to get my hands on that study because it gives strong evidence why it can make sense to do what you're doing.

And I give a number of different strategies to avoid having this happen and one of them is to pair females and males together separately, at least some of the beginning classes until the females gain more confidence. Also, overall as a group, they tend to come less prepared because they have less informal experience. I have a whole hour on why that is.

Those in initial courses, in particular, are having less confidence. Then there's this giving/taking dynamic that you're talking about. So actually having them pair with other females controls for that, but not totally, which is why you still need to have some structure to it because you can also have female students who are taking the mouse.

It's interesting that what you've done is single-sex pairing. I mean, this is sort of a practical question, but what do you do when you have uneven number of students and now you have to pair a male and female together? How do you handle that?

Charlie: Well, that's certainly fine. I actually have large classes and if there's an odd number of women, one of them is going to get paired with a man. But, at least, she's not going to be paired with a man every time.

I also do have this situation where if I have an odd number of students in the class, what do you do about that? There are careful studies about this and I have to admit, I waffle back and forth. I don't like three-person teams because it's kind of hard to really do proper pair programming with three, so I resist that. I'll admit I've let it happen a few times when I had three students that all came to me together and really wanted to do it, so I've let that go.

In general, I will say odd-man-out, that odd-person-out, the odd-woman-out, whoever it happens to be, is sometimes quite happy to just work alone for that assignment. I always tell them, "Hey, what happened here is, it turns out, your partner's going to be me. So you get to work with a partner. You get extra time in my office. You can come by and I'll be your partner."

If they're unhappy about being left solo, I just explain to them, "We'll make sure you get any extra help, so feel free to come to the head of the line."

Donna: That's great. How often do you switch? You mentioned that every 15 minutes, they should switch their roles, but how often do you switch the students in terms of the pairing? Like do they have the same person all semester or does it change per project?

Charlie: Sure, sure. In fact, in my classes, we have what are called open labs. Much of the time that they're working together, I'm not able to monitor them because they need to spend time on programming assignments outside of the lab. So they do have to schedule time together.

It's easier to do in closed-lab situations. In many places, they do have closed labs. So the students are doing most of the work in a lab, which is great. In fact, in those sort of structured situations, some people literally have a timer and every 20 minutes, ding! The bell rings, and it's time to switch partners and you change roles. I think that's particularly important when working with young people.

Donna: How often do they switch partners? Like do they get the same partner the whole semester?

Charlie: Sure, right. In the study that we did, we actually had them stay with the same partner. For that study, every student gave us three names and we then managed to pair almost everybody with somebody that was on their list.

Since then, I've evolved. And there's this tension when you have the partnerships that have to stay together for the whole quarter. If you have a dysfunctional partnership or a person that really feels like their partner is slacking, that's problematic in that they're stuck with this person for the whole term.

So you do have to try to figure out how to change those. My response to that has been to actually migrate to a fairly hybrid system that I'm quite happy with: I assign partners. I change them for each assignment, which is about two weeks. They're longer, two-week-time assignment periods.

However, students can indicate a preference to work with a specific partner. I have a large class. They do this through an online system that's not very user-friendly that I've sort of developed for myself. They can request a partner. Both partners have to do it. So I try to make sure that one person is not feeling pressured. If they don't both do it online, the person can make an excuse. "Oh, I forgot" or whatever. The bottom line is they're not supposed to work together as a pair if they haven't both requested it.

Donna: Sure.

Charlie: So they can request a partner. If they don't request a partner, then I pair women with women and men with men. I also do bring in a factor to try to pair students of similar ability because there have been a number of students that suggest that that's important. You don't want a really strong programmer working with someone that's really struggled. That difference gap can be problematic for both partners. You want them to be relatively close together in ability.

I'll bring in something like class exam scores. For the first assignment, when I don't have any exam scores in my large intro class, I tend to group them by major. So I've got my proposed computer science majors sort of grouped together and I've got the students that are taking this for who knows whatever reason. From arts or humanities, I've got them paired together. So it's more likely that they're going have similar kinds of experience when they come in.

Donna: Did you actually take a computer program to sort of do that kind of sorting?

Charlie: Yeah, it's a really kind of messy bunch of Python scripts. It's basically you would think of as a small web app that pairs the students and manages those requested for partnerships.

Donna: How many students do you have in a class?

Charlie: Too many. Most recently, my intro course could have over 300 students in it.

Donna: Oh my gosh. How many sections do you teach?

Charlie: It's one big lecture that I try to keep interactive. And then they meet in sections with 20 to 40 students in a section. In that section, there'll be a graduate student TA, and depending on the size of the section, one or two undergrad tutors to help them out as well.

Donna: Well, that is a really large class. So I'm just curious — and I know UC Santa Cruz doesn't grade the same generally as the rest of colleges do — are your classes on a pass/fail system or a grade system? I'm wondering if these students are graded as a pair.

Charlie: Well, this is great, another chance for me to get rid of some old data. Santa Cruz has a more or less conventional grading system at this point.

Donna: Ah, okay.

Charlie: When I first came here, we didn't give any grades and we only wrote narrative evaluations. But at this point, we have a pretty standard grading system with A-pluses and minuses, and A, B, C, and F's, so pretty normal.

Donna: Oh, wow. Okay.

Charlie: The students grade us – only a small portion of their grade is based on the programming assignments. That was true even before I started using pair programming because quite frankly, one of the concerns that people have about pair programming is like, "Oh, if they're working together, how do I know who really did the work?"

And the bottom line is you don't, but you don't know that when you tell them to do it alone either. Nowadays, they've got the Internet, they've got their roommate, and they've got another classmate. You'd be surprised at how much code a student can milk out of a TA without the TA realizing what's going on in terms of how much they helped them.

So that helps them regardless. That's just not a reason to not let them gain all the benefits of pair programming because some students maybe "didn't do the work." Some students aren't going to do the work no matter how you approach working together.

Donna: Okay. How is the majority of the grade? What is the majority of the grade focusing on then?

Charlie: Yeah, what I do give? So the programming assignment, part of the grade is maybe 20%. I can't remember what I've used most recently. The rest of that is going to be primarily on exams. The bulk of that is sort of split between a final exam and in my particular course, some people give a midterm. I actually give a short exam every two weeks just to try to keep students on top of things.

Those exam scores are done in a conventional way, with individual students. So they have to complete the exams. I sort of point out to them if you shortchange the programming assignments for whatever reason, it's going to show up on the exams. I have to believe that my exams are a good measure of how much they have achieved. If they do the assignments, they should do well on the exams, and I believe that's true.

Donna: So that is so great that you're giving them lots of mini-exams along the way so that they get an indicator. The actual exams — are they hands-on? Are they actually doing programming, writing a program for the exam?

Charlie: They have to write some code. It's not a perfect situation. I'd love to be able to test my students by having them sit with an editor and a compiler and run and debug a little piece of code. That doesn't happen. They actually have to write code out with pencil and paper, and we have to grade those things by hand. So they do write some code for me.

Donna: Okay, okay, so I guess what I'm wondering is has there been any follow-up studies, either of your original students that you studied, or have you done subsequent similar studies with pair programming? Are there other students that you know of? How widespread is this in the research world?

Charlie: Well, there have been a lot of studies done since ours. We did a little bit. I'll maybe share, in an attempt what we were doing, we pretty much tried to do the same kind of study with the same questionnaires at a community college and at San Jose State at a four-year, non-research university.

The San Jose State study was interesting. We had two faculty there that agreed to participate. They teach the same class every quarter. So one of them, one quarter was going to use pair programming, and the following quarter was going to use solo. The other one was going to do solo the first quarter and pair programming the second to sort of try to control for different things. The order that you have the control when you do the intervention is maybe a factor because the teacher changes somehow.

It was interesting because the guy that taught the paired programming section actually sort of refused to be part of the study after because he said, "There's no way I'm going back to doing solo programming." From his perspective and everything he observed and what the students were doing in class and how they behaved, – and this is of course a much different situation than mine because this was a class of 25 —, so this is about an instructor with 25 students.

And he said, "There's just no way I'm going to not do pair programming from here on out." So he was completely bad from the point of our study. But was good for the students that got to do that.

There also have been a lot of people doing work. In sort of preparing for this interview, I went and checked. If someone really wanted to see what's happening in the literature, I think one of the best ones that I found was in 2011 in the IEEE Transactions on Software Engineering.

There's a study called *Empirical Studies of Pair Programming for CSSE Teaching and Higher Education: A Systematic Literature Review*. They cover 74 studies, including ours, most of them that happened since ours, that are looking at lots of different aspects of this.

We certainly don't have time. I mean, that's a whole interview to just talk about all the things that they found. But nothing in there negates anything that I've told you today.

Donna: So it's similar findings.

Charlie: Similar findings and talking about, again, these situations of does it matter how you make the pairs, closed-lab situations, is the quality of the program – are the programs better? Do they get them done in less time? There's a whole host of other things that you can look at and they talk about. They do a nice job of trying to summarize a huge body of work.

Donna: Well, I will take a look at that. I have to say that what you described a bit earlier in a closed-lab situation, which I know is not your situation, where you call time and switch, is also what I recommend in addition to the pairing, to have it structured in that way.

I realize it's not possible in your class. And the other thing I'm wondering – and I started to try to get at it when I was asking you about the grading –are the grades better? Did you look at that in the original study? Do you have any sense of that from your class versus your colleagues that don't pair?

Charlie: Well, sure. First of all, the grades, as we saw them, were about the same. But the number of students that actually persisted to the final is greater. So popping back and looking at this table from the ACM article I mentioned that we published:

In the solo class, 80% of the students made it to the final. I had 20% voluntary attrition from before the final. In the pairing class, it was 91% and that was for all students. With women, it went from 80% to 88%. Now, that wasn't a significant result and that was partly due just to the lack of the power, because the number of women wasn't quite large enough. I mean, this is a substantial difference. It's a nine-percentage difference in terms of the number of students that stayed to the end.

The grades were about the same, but there were more students that were getting those grades. So I would say overall, there was more learning taking place.

Donna: Absolutely. I know you that you mentioned that pair programming is also done in industry. Has there been any study that looked into if you get better code, better computer programs when it's done by a pair?

Charlie: Yes, I think it's a little bit mixed, but most of the studies indeed find that the programs that are produced have fewer errors in them. So that's a notion of “better”. They tend to take about the same amount of total programmer time. The wall clock time would be a little bit less, but it's not 50% less, so it ends up being about the same amount of effort. But they end up with higher quality programs, with fewer bugs that show up later.

Donna: Okay, good. So that's another plus for the pairing. Now, you mentioned that you also did a study with two-year colleges. A lot of our listeners are from two-year colleges. Would you share with us about that study?

Charlie: Boy, I can't –

Donna: Can't remember any specifics?

Charlie: If I recall, we did that a long time ago. I'm going to have to say there was nothing surprising that came out of that finding, and probably because we lost the San Jose State folks. I'm not sure we published much about that. It was sort of part of the NSF study work we were doing and probably showed up in an NSF final report. But I honestly don't remember. I clearly remember that anecdotally, everybody likes it and certainly the community college where we did it, the folks there use pair programming in their classes now.

Donna: Sure. Now, what about other types of career pathways and other types of labs? Have there been studies about how this would work in an engineering lab or even an auto technology lab? I'm wondering if this is applicable to other program areas.

Charlie: It's certainly my understanding – and I haven't followed it closely, but it certainly comes up from time to time – that project-based work and by that, I mean team project work, whether it's two people or small teams of three, four, or five students – is really important. It's beneficial from an educational point of view and is strongly being encouraged to be introduced as early as possible in a curriculum, particularly in that first year, to help students for some of the same reasons. It helps students if this is a collaborative activity.

Certainly in all of the engineering disciplines, when they get out in the real world, collaborating and working with the team are important. You don't sit in a cubby by yourself and design a bridge or whatever you're going to build.

I'm not aware of studies that have specifically looked at the kinds of questions we were looking at to say if you used some kind of two-person team activity in an engineering course where you used to make them do all their projects by themselves, you'll see higher retention.

But in terms of the things that the reasons why I think pair programming is important, I would think that they would apply there. Those reasons being you feel it's a collaborative activity. I think particularly in the case of women in engineering and in computing, and in computer science in particular, the women tend to come to it with less prior experience. I think 19% of computer science high school AP exam takers are women.

Donna: Yes.

Charlie: So there are five times as many men that come to the university that already have computer science experience. I suspect that may be the case in other engineering areas, that there may be more guys that have been tinkering in a garage or with their car or something before they get to university and decide to pursue some engineering activity.

When you let them work together, especially now after you've paired them with a same-gendered person, that woman is now talking to another woman that they can share ideas and say, "Oh, well, gee, I guess I'm not the only one who hasn't done this before."

The problem is the woman is in a class; she's not talking to other women. She sees the guy raising his hand, doing his alpha male thing and showing off by answering a question that's way beyond what should even be in the class. She says, "Oh, gee, I guess everybody knows this. I don't belong here."

Now she's talking to these other women through the pair activity that they're doing and that other person is more likely to also be in the same situation and they realize, like, "Wait a minute, I'm not the only one here." They also are building a community and a little awareness of it.

So it seems like it's going to help. The reasons that I think it helped in pair programming would help in any engineering discipline.

Donna: I think so, too. I do remember in your study that you also looked at confidence and that confidence really went up for women – and also for men, but especially for women – when they did the pair programming.

Charlie: Yeah.

Donna: You're absolutely right on the mark in terms of the less experienced. If you're paired with someone who already has tons of informal experience, then they're going to dominate, especially if they're the mouse-taker. Having that separately by gender really makes that less likely.

Gosh, I just went and looked at our question-and-answer screen and we have tons of questions. So I think what I'm going to do is ask you a final question and then I have some information for everyone, and then we'll go to the Q&A, because there are tons of questions as I thought there would be.

For those who are listening and are interested in trying out pair programming, what's your advice to them who want to give it a try?

Charlie: Just do it. I firmly believe any pair programming method that you do is better than making them work alone, live in fear of being accused of copying if they talked to someone about their program. You don't need a complicated system. At the very least, make it an option and let those that want to work with a partner work with a partner. Better yet, make them do it at least for one or two assignments.

There is research that shows that students initially don't want to do it. Oddly enough, they feel like, "Oh, I have to get in there and do it myself." But once they do it, they actually enjoy it. So it doesn't have to be complicated and formal. You don't have to have a fancy system for doing the pairing. It can be better if you do that, but just let them pair. Let them pick their own partners. That's better from my view. That's better than making them just go and do it by yourself and if you talk to anybody, that's cheating.

Donna: Well, that is some really fabulous advice. Move into imperfect action is what you're recommending.

Charlie: Yes.

Donna: I also hear you saying something very important to remind our listeners: initially, they're not going to like it. They won't like it the first couple of times. They're not used to doing it. But then after that, things will change.

Thank you so much, Charlie, for joining us for the STEM Success for Women Telesummit and talking about your groundbreaking work in pair programming. We are going to go to our question-and-answer in a moment.

Let's move to our questions and answers. First question for Charlie is from Peggy in Lexington. She asks, "Did the paired groups remain the same throughout the class or did the pairs gets mixed up through the course?" Okay, so we actually answered that one. It was a question I had.

And the next one is from Linda Werner. She says ink, pen, and claw and others – and she's from Santa Cruz – so she's one of the people you mentioned – studied young children working in a museum environment and wrote about how they perceived the collaborations to be. They named the protocol "take" typically boys and "give" typically girls.

Great. Thank you so much, Linda Werner. All of us can go take a look at that study. She indicated that it is from 1995 and it's called *Give and Take: Children Collaborating on One Computer*. I can't wait to take a look at that study. I know it's from '95 but I think that all of that probably still is occurring today. I don't think there's been any change.

So next question –

Charlie: Kids are still kids.

Donna: Kids are still kids. But I have to tell you, this also happens with adults. In my session about *How To Be Successful in the Lab*, I give an anecdote myself where I was in a bike repair class, how to change your tire if you get a flat. I brought my bike and I was paired with a male student who did not bring a bike, but also had a lot more background than I did in bike repair. During the lecture portion, he was explaining parts of the terminology for me that I didn't know and the instructor wasn't breaking down.

Then when it came time to turn over my bike, he immediately put his hands on it and started taking the – and I had to tell him three times, diplomatically, of course, that I really needed to do it myself — and could he just sort of stand back and be my coach when I asked him for help? It took three whole times. So I would say even with adults.

Charlie: Actually, I'm going to jump in there and just comment. I give one piece of advice, of instructions for my students that work in the lab — the teaching assistants and the undergrad assistants – there's one thing that I emphasize every time. The first thing that I tell them is when they're in the lab, they must never touch a mouse and a keyboard. It's sort of the same thing, particularly if more of your teaching assistants are men. Maybe they're more likely to do that. You just don't touch. You have to talk them through it. You can't touch it.

Donna: That is very, very important: not to touch and to, as you say, talk the students through it. I'm so glad that you mentioned that.

Now, Andrea Calloway from New York at PACE asks, "How does the paired programming experience in the classroom translate to the real-world programming environment? Is that identified to the students as part of a reason to approach the course this way?" Great question.

Charlie: Well, certainly, as I think I mentioned, it is part of more than that. Certainly, it's at least part of this agile software development process called extreme programming. And yes, we tell them about that, that this isn't just something that I we do for educational reasons, but this is a really great process to use when you go out and you're working in the real world.

Not every company necessarily embraces it, but even if it's not pair programming, there are certainly many times in the day that this is done. I was reading something that someone was commenting about how much time. They claimed that they spend about two-thirds of their time during the day is spent collaborating with someone else and about one-third of their time is spent working alone.

So if it's not pair programming, it's certainly working with someone else on code in various kinds of collaborative ways.

Donna: And I think that there are more and more research and sentiment that says you develop better products when you work in teams and it's not one person working on their own. The culture around that has changed so that it's increasingly valued. That's my sense of things.

Charlie: Yes. There are plenty of studies that are reporting diversity improves products and companies and diversity in many regards, gender being one of them.

Donna: So Brian from San Francisco asks, "Have studies shown that pair programming is similarly beneficial for other groups of traditionally underrepresented students, particularly students of color? Also, what, if anything, is different?" So do you know of any studies in that area?

Charlie: I don't. The first thing that I would do is I'd go quickly and rescan that one paper I mentioned to see if they have any that were specifically looking at other characteristics. I can't imagine why it wouldn't. Many of the same benefits would occur there. I guess that might raise the question should you, for example, consider bringing ethnicity into the pairing. I don't know. That's a good question; maybe somebody might have looked at it.

Donna: That's a further study.

Charlie: Yeah.

Donna: What about your classes? Are they ethnically diverse? Are there many students who are underrepresented?

Charlie: Not as many as we'd like. I think we have a reasonable population of Hispanic students in my classes here at Santa Cruz. The African-American population is pretty small, so they're definitely in a very small number. I have a class of 300 and I might have at most a handful of African-American students in there.

They tend to find each other, but I probably could do things. It would be interesting to sort of maybe ask them how they would feel about that if I actively paired them together.

Charlie: Again, as I pointed out, they can choose to pair with themselves if they'd like to do that.

Donna: That's one of the factors. Do you have a substantial percentage of Hispanic students?

Charlie: Yeah. In fact, Santa Cruz has recently become a Hispanic-serving institution, which I think means as a campus we're 25% Hispanic at this point. We probably don't have 25% Hispanics in my engineering classes, but I haven't looked carefully at that, so I don't know for sure.

Donna: But that could be a future study.

Charlie: Yeah.

Donna: The only thing is that I know that from what you told me, nobody's going to want to do the control group on that.

Charlie: Right.

Donna: But that could be a future study.

Charlie: They could consider whether I should be taking that as helping them even more in terms of fighting the community. By the way, I do the pairing. I'm certainly not going to make them work alone, that's right.

Donna: Yeah. Well, I have one more question that I didn't ask you before in our Q&A. Here we are, it's 15 years later after you've done this work and had your paper published. How come colleges – and not just colleges, high schools, too – how come they're not all doing pair programming? I mean, it works better for both women and men.

Charlie: Well, as you've apparently indicated, many people maybe just don't know about it yet. We're working really hard to get the word out there more. I think for people that find out about it, maybe they haven't read the studies; maybe they have. I think there are two primary impediments why people tend to not do it.

I think one of them, as I alluded to earlier, is the assessment factor. If the students are writing the program with a partner, how do I know who really wrote the code? As I said before, I think that's a legitimate question to ask even when you've extensively told them to do it individually. So I don't think that's a reason for not doing it.

And even if it turns out you do have some slackers that take advantage of the pair programming aspect of the class, you're going to have them anyway and I don't see any reason for depriving the majority of the students of this benefit to try to prevent a small minority that might abuse it.

The other thing is the administrative aspect of it. My answer there is do whatever's easy. It doesn't have to be an administrative burden. In fact, you get some gains. The fact that you have pair programming, I will point out very quickly, you can do the math. You have half as many assignments to grade.

So for instructors in small classes where they have to grade the assignments themselves, guess what? Your class of 24 now has 12 programs, not 24 programs. So there's the win right there in terms of your own personal time.

Donna: Well, I have to tell you, I am on a personal mission to share your important work and those of your collaborators as well with as many educators as possible. It's been my experience when I do the WomenTech Educators Training and I share your numbers and success, that I have people immediately say, "Oh, wow. I mean, I did not know about this. I want to do it." And they put it in their retention plans immediately.

I really believe that there are many, many, many more educators that would definitely do pair programming if they had the information about the difference it can make on not only retention, but declaration of a computer science major and that they would totally adopt it.

I will, right now here in this Telesummit, commit to doing everything I can to bringing this information much further. And again, I just want to thank you so much for your participation and for your work in this area and for your commitment to improve retention and improve the quality of the computer science introductory experience for your students. Thank you so much.

Charlie: Well, thanks for helping spread the word about this best practice.