
Summary
The investigators propose to develop an early computer science (CS) curriculum for a semester-long secondary school course, and assess its efficacy. The course, “Introduction to Computer Science with Explicit Modeling (ICSEM),” immerses students in a constructivist learning experience as they explore the field of computer science through an intensive study of object-oriented, explicit modeling. Students’ early introduction to sophisticated CS competencies is enabled by a novel software tool called “Objektgraph,” which has already been developed and demonstrated as a prototype. Objektgraph and the ICSEM course are designed as effective methods of instruction for all learners. However, they have been strategically designed to address the needs of girls and students from other underrepresented populations whose social and intellectual learning needs are often unrecognized and unsupported by traditional CS courses and curricula. Students who are relational thinkers, favor social learning experiences, benefit from discussion, and require practical motivation will benefit from powerful accommodations that promote their learning. This project requests funding for the refinement of the Objektgraph tool, the development of the ICSEM course, and their implementation in secondary classrooms. It also seeks funding that will contribute to more productive recruitment of girls in the course and the design of interventions that will increase their retention in the CS field.

The central hypothesis is that participation in the ICSEM course will positively affect the participants’ (1) performance of CS competencies, (2) sense of CS efficacy, and (3) interest in pursuing further CS education. The extent to which planned supports contribute to these for girls will be the focus of qualitative study. It is expected that the dominant thinking style of girls (i.e., relational thinking) will be validated by using Objektgraph to perform explicit modeling activities. This is likely to influence all three measures. It is also anticipated that learning in a positive, social environment will result in girls having a more favorable opinion of a CS career—thus influencing their plans to pursue further CS education. Being able to use Objektgraph to create concrete work products that can be readily shared and discussed will enhance both girls’ motivation to learn and their feeling of inclusion in a community of practice. We assert that early, supported exposure to the meaningful aspects of computer science practice, in an environment that successfully mitigates the impacts of isolation and barriers of traditional CS culture, will help girls realize their innate potential for success in the field.

Intellectual Merit: (1) Teaching explicit modeling using Objektgraph is expected to privilege the relational thinking style dominant among girls (and perhaps common among students in other underrepresented populations) and influence their experience in other positive ways. If so, more widespread exposure to early modeling experiences could broaden participation in CS and provide insights for CS education reform. (2) The ICSEM environment supports and encourages the construction of software using highly desirable software engineering practices. This is expected to result in the identification of promising new approaches for CS education. (3) The ICSEM offers a radically different and potentially more desirable approach for introductory high school CS courses—one that provides greater depth without sacrificing the breadth of traditional courses. It promotes exploration of new pedagogies.

Broader Impacts: If successful, wider adoption of the ICSEM and Objektgraph has the potential to increase both the quantity and quality of students entering the CS field—especially female students. For women, increasing gender representation in turn creates environments that are more inclusive, supportive, and attractive to more women. Increased representation of women in the CS field promises to result in innovation, expand markets and create new services that fuel the economy. Development of strategies for recruiting and retaining girls in computer science refined in the project will also be of use for other CS education efforts and in other STEM fields.
Introduction
Secondary and post-secondary education providers throughout the United States play a critical role in and building a workforce to power a technology-driven economy. To maintain its competitive advantage in the global marketplace, the US must produce an increasing number of competent, high-tech workers. However, the nation’s “production” of high-tech workers does not keep up with projected demand. Since 2003 the number of computer science and information technology graduates has declined by 11% while the demand for such graduates has risen 13%. [34] A more disturbing aspect of this decline of is that the proportion of women and members of other underrepresented groups in this general population is at an unprecedented low. This represents a critical area for inquiry and intervention.

Project Goals and Outcomes
The situation described above is the raison d'etre of the STEM-CP CE21 RFP. If women and other underrepresented groups were to enter the computer science field at the same rate as men, the number of students that would pursue degrees could nearly double, contributing to an adequate high-tech workforce. The first goal of this project is to promote the development of more computer scientists through strategic recruitment and retention of girls into computer science education. The second goal of the project is to promote the development of more competent computer scientists by improving the educational preparation of both male and female students to be successful in subsequent studies of CS.

This project proposes to further these goals by introducing computer science through explicit object-oriented modeling. This approach will facilitate the development of an introductory course of study with characteristics that support the goals of inclusivity and improved success in subsequent CS courses. This approach is a radical departure from extant “objects-early” curricula, and should not be confused with them. However, it shares an outcome objective that is implicit in those curricula: that the student be able to create effective models, recognize their primacy over procedures, and value them as an important part of computer science. These outcomes will be achieved by more students using the approach advocated here because it better supports students with diverse intellectual styles, most notably the relational style, which is more commonly associated with girls. An additional outcome objective here is that students gain insight into the characteristics of careers in computer science-related fields. This will be facilitated by students having experiences developing complete software applications in complex domains.

Implementation Plan
To realize these goals and outcomes, this proposal requests support for the development and implementation of the introductory, semester-long high school computer science course: “Introduction to Computer Science with Explicit Modeling (ICSEM).” This course immerses students in a constructivist learning experience as they explore the field of computer science through an intensive study of object-oriented, explicit modeling. Object-oriented explicit modeling as defined here refers to creating a model where meaning arises primarily from associations (rather than attributes) which make the model visible diagrammatically. Explicit modeling is related to natural language (that of everyday speech) with nouns and verbs expressed in diagrams rather than textual syntax. [3] Students’ early introduction to sophisticated CS competencies is enabled by a novel software tool called “Objektgraph,” which has already been developed and demonstrated as a prototype by Duane Buck (the grant PI). [2]. Objektgraph and the ICSEM course are designed as effective methods for the instruction of all learners. However, they have been strategically designed to address the needs of girls and students from other underrepresented populations whose social and intellectual learning needs are often unrecognized and unsupported by traditional CS curricula and courses. Such efforts require an intentional and radical departure from tradition. They require a recognition of the extant barriers faced by students from non-dominant groups when experiencing existing curricula, courses and programs. This project proposes an innovative approach that leverages research-based instructional approaches, powerful software, and quality teaching to improve the opportunities and experiences for many. This course will be developed and evaluated by a cross-disciplinary team of experts representing five institutions of higher education and will involve students in 12 schools, in 3 location, and in 2 countries in its evaluation. The project’s inputs, activities, outputs and outcomes are summarized in Figure 1.
**Background and Motivation**

Students who are relational thinkers, favor social learning experiences, benefit from discussion, and require practical motivation will benefit from the powerful accommodations that facilitate their learning in the ICSEM course proposed here. This project requests funding for the refinement of the Objektgraph tool, the development of the ICSEM course, and their implementation in secondary classrooms. It also seeks funding that will contribute to more productive recruitment of girls into the course, taking into account its unique approach, and the design of interventions that will increase their retention in the CS field. Students who participate in ICSEM will be able to engage in sophisticated, computer science related thinking that has been heretofore impossible in an introductory course. They will have a deep and meaningful experience that will provide them both enjoyment while learning and an authentic motivation for pursuing a career in a computer science.

This proposal is part of a larger initiative to exploit the potential of explicit modeling in computer science education, enabled by Objektgraph. The Objektgraph tool—one of the most ambitious computer-aided software engineering (CASE) tools ever undertaken—has already been developed as a prototype with support from Otterbein University and the University of Oldenburg, Germany, and was demonstrated at SPLASH’13. [2] If proven successful, future anticipated projects will develop curricula using the approach for middle school and postsecondary CS education. Far from a static and final product, the Objektgraph software is robust and evolving. As a component of this project the tool will be refined and its functionality will be expanded to enable the deployment of applications on smartphones.

**Project Background**

The extant barriers for women in CS, the characteristics of women, and what can be surmised about their needs, inform this effort to broaden participation by including this underrepresented population in computing education.
**Barriers to Women in Post-secondary Computer Science Education.** For CS students from underrepresented populations, success is hindered by numerous barriers. In their synthesis of research on women’s participation in post-secondary computing education, Cohoon and Aspray [4] identify ten factors that have received attention as possible explanations for the underrepresentation of women in computer science including the culture of computing, experience, barriers to entry, role models, mentoring, student-faculty interaction, peer support, curricula, pedagogy, and student characteristics. These factors have great relevance to our work. They provide a framework for the project we propose and direct our efforts to recruit, support, and retain the students who participate. Here we share descriptions of a few of these factors with particular relevance.

*Culture of computing.* Culture involves the language, beliefs, customs conventions, values, and artifacts. Because CS is predominantly male, CS culture often overlaps and takes on the characteristics of masculine culture [30]. The key characteristics of this culture are aggression, ambition, competitiveness, dominance, and forcefulness. Perhaps this information about the culture of computing explains another barrier for women getting into post-secondary CS education: having less experience.

*Experience.* Precollage experience with computing includes the informal experience with computer games and applications as well as formal experience with computing classes. Research suggests that girls have less exposure to these earlier experiences than boys. Although recent reports suggest that informal experiences with technology among girls are increasing [7], girls are significantly underrepresented in formal educational experiences.

*Role Models.* Role models serve as an example of the values, attitudes, and behaviors associated with a role. Girls have a noted lack of role models in CS because there are fewer women in the field. Given the axiom “You have to see it to be it,” it is not surprising that significant investments in broadening the participation of women have done little to change the status quo. [8]

*Student-Faculty Interactions.* Student-faculty interactions describe the nature and variety of experiences students have with their teachers. Of interest to researchers are whether such interactions are supportive or not, who initiates them, and more. Faculty attitudes, values, and behaviors all have important effects when broadening participation is the goal.

**Gender Differences in Learning**

Interest in the impact gender has on learning has been longstanding among researchers. Over time, the following truth has emerged: girls and boys are equally capable of learning but do it differently. A comparison of cognitive ability, attitude, and behavior across genders has identified some differences that have relevance to our project.

Girls have long been recognized as having superior language abilities [28]. Girls perform better on timed perceptual tasks such as matching stimuli, scanning visual arrays, and copying forms [15]. Gender differences also exist in memory. These include the speed of memory recall and the degree of functioning for short-term and working memory. Women are faster at memory tasks [29], have better short-term memories [20], and have larger working memories [17] than men. Halpern [16] divided visual-spatial abilities into five types which prove to be useful in thinking about gender differences: spatial perception, mental rotation, spatial visualization, spatiotemporal ability, and generation and maintenance of a visual image. In comparing male and females on tests related to these tasks, males were more proficient in numerous areas including spatial rotation and mental rotation tasks. On visualization tasks, men were faster, but women were more accurate.

**Girls and Relational Thinking.**

The barriers girls experience in post-secondary CS education have numerous sources. Biology and socialization are two important ones. Both are likely causes for another important difference that distinguishes women: their dominant thinking style. Turkle and Papert [32] in their study of students in their programming courses identified an important difference in the way male and female students approached computer programming. Their approaches were reflective of the innate thinking styles they used. Women tended to practice relational thinking—which considers more pieces of information
simultaneously—and look for connections following identified patterns. This thinking style was described as concrete, “bottom-up,” or “soft.” Men, on the other hand, tended to practice more hierarchical thinking styles which connect information in an axiomatic progression. This style was also referred to as “top down” or procedural. Among other noted differences, difference in thinking style had a significant impact on the degree of success and enjoyment women had while participating in the course. Led by a male instructor, who unconsciously promoted only the thinking style most compatible with his own, the girls were pushed to conform to an inorganic approach to programming. Those girls were less successful, and less likely to continue CS education.

Williams [33] also observes the difference in thinking styles observed by Turkle and Papert and acknowledges the dominance of the “axiomatic” over the relational in STEM fields outside of CS. She makes a case for broadening participation in STEM and encourages the inclusion of more diverse and more numerous thinking styles in scientific fields saying, “academic tradition in science favors axiomatic communication and pedagogy, but the relational style is an agent in creativity, and scientific validity is established through a diversity of styles.” She further points out that diversity promotes “innovation and proposes that new science and technology will rely increasingly on relational thinking.”

This background information on women in computer science suggests both reasons and methods for reforming course curricula, instructional methods, educator preparation, and other support to promote greater inclusion of women. The development of the ICSEM course, refinement of the Objektgraph tool, and creation of recruitment and retention strategies proposed here—if armed with these insights—can strategically address and broaden the participation of women in computer science. Based on the notion that what is good for females will also benefit males, we propose that early introduction of object-oriented, explicit modeling is an appropriate intervention to produce both more and more competent computer scientists. In the remaining sections which detail the project background, we first describe why early introduction to modeling is important. Then we make a case why teaching explicit modeling may provide needed support for girls’ relational thinking styles.

**Interpreting the Work of Turkle and Papert to Explain Explicit Modeling’s Advantage for Girls**

One argument in favor of the hypothesis that explicit modeling differentially supports girls may be derived from the work of Turkle and Papert [32]. In their study, they found that girls tended to think about computer programming more relationally than boys. This is expressed as closeness to the objects and their arrangement and as identification with the objects. One way to characterize this style is “bottom-up,” or “soft.” The latter term as used here has no pejorative implication. On the other hand a top-down (axiomatic) approach is more indicative of boys, where the preferred style of thinking is in terms of procedures, and they may not identify with the computational objects. Turkle and Papert give as an example of the contemptuous comment of one fourth-grade boy who overheard a classmate talking about "being a sprite" when programming. "That's baby talk," he said. "I am not in the computer. I'm just making things happen there." Turkle and Papert also make arguments that anticipate the efficacy of the approach proposed here. First they consider the power of the computer to give form to ideas:

“The conventional route into formal systems, through the manipulation of abstract symbols, closes doors that the computer can open. The computer, with its graphics, its sounds, its text and animation, can provide a port of entry for people whose chief ways of relating to the world are through movement, intuition, visual impression, the power of words and associations. And it can provide a privileged point of entry for people whose mode of approach is through a close, bodily identification with the world of ideas or those who appropriate through anthropomorphization. The computational object, on the border between the idea and a physical object, offers new possibilities.”

The people they are talking about in their study are predominantly girls. Elsewhere in the paper they refer to the approach as “soft,” but argue that it is just as valid as the “hard” approach of the predominate computer culture. Later, as they look to the future, they go on to say:

"The icons in the Macintosh reflect something deeper, a philosophy of "object-oriented programming."

In the traditional concept of a program the unit of thought is an instruction to the computer to do
something. In object-oriented programming the unit of thought is creating and modifying interactive agents within a program for which the natural metaphors are biological and social rather than algebraic. The elements of the program interact as would actors on a stage. This style of programming is not only more congenial to those who favor soft approaches, but puts an intellectual value on a way of thinking that is resonant with their own.

Turkle and Papert are accurate in their vision of the potential of the object-oriented programming, but perhaps overly optimistic regarding the difficulty of the realization of their vision. Subsequently, with the different goal of improving the modeling skills of graduates, new computer science curricula have been developed that introduce object-oriented programming languages at the beginning. However, that has proved insufficient to realize the vision of Turkle and Papert, and perhaps was counterproductive. Although object-oriented programming supports the implementation of an explicit model, it does not require one. For many students, learning an object-oriented programming language simply added more topics to an already crowded first course. Such a course may only be effective for elite “hard” thinkers who enjoy ever more abstract challenges. Even some elite schools have reverted back to procedurally-oriented programming in the first course. [24, 25]

The approach here does not use an object-oriented programming language, but instead uses diagrams that immediately enable students to model concrete objects, manipulate and associate them relationally, and create their behavior in terms of their relationships. The Objektgraph tool creates exactly the type of environment Turkle and Papert anticipated would resonate with those who favor relational thinking styles.

**Contrasts with Current High School CS Offerings**

To understand the value of the course we propose, it helps to understand how it fills a crucial void in curricular offerings. This course is fundamentally different from other computer science offerings at the high school level. It targets the competencies in Levels III and IV of the ACM Model Curriculum for K-12. However, it differs from other courses targeting those levels because its focus on explicit object-oriented modeling with the Objektgraph tool supports a much broader range of intellectual styles; existing courses have failed to increase the participation of girls and other underrepresented groups because the courses favor an axiomatic thinking style. The new offering, Principles of Computer Science, [5] although apparently targeting increased diversity, does not significantly address this issue. It seemingly attempts to make CS more accessible by eliminating much of the coding content and substituting foundational CS content. However, much of the new content (the principles) is theory based, and continues to privilege learners with axiomatic thinking styles. In addition the principles course lessens the experience which is the primary reason people are drawn to computer science: giving life to ideas. Another offering targeting inclusiveness is Exploring Computer Science, [12], and it appears to offer much more to the relational thinker. However, it targets lower levels in the ACM Model Curriculum for K-12.

**The Explicit Modeling Curriculum and its Enabling Pedagogical Tool**

The curriculum proposed here is contingent upon a pedagogical software tool that supports it. The tool has been named Objektgraph, recognizing its German heritage. Objektgraph builds on the human mind’s natural solutions for understanding the world. The pedagogy supported by the Objektgraph software environment represents a radical departure from that employed by extant computer science curricula. Objektgraph lowers the level of frustration experienced by students. At the same time, it raises the level of abstraction in the early curriculum and raises students’ awareness of what real computer science practice is really like. We believe using Objektgraph to do modeling may permit novices to experience what it is like to have a career in software development.

An example of the context in which Objektgraph will be employed is described by Diethelm et al. [9, 11] First students are introduced to concrete object models within familiar domains expressed as UML object diagrams. Next, they explore new domains, which are often board games, and are asked to create object models. There is not a single correct model, and there may be fruitful discussion over their relative merits. As they create objects using Objektgraph, they are implicitly encouraged to classify objects, if for no other reason than to save time because a new object is “like” a previous one. However, classifying
objects is natural in any case, and students find reasoning with the diagrams not too different from using language descriptions.

Later, students learn the concept of a class diagram, which identifies the types of objects that may be used in an object model and how they may be associated with other objects, which is analogous to a grammar. Objektgraph facilitates this because the class diagram is created in the background as they develop object diagrams. When the students first view the class diagram, the abstract concepts are readily understood because they are already familiar with concrete instances it describes.

**Story Diagrams**

Writing story diagrams is equivalent to writing methods in an object-oriented programming language. In contrast with a traditional implementation that requires numerous methods dealing with both the implementation and problem domains, fortunately Objektgraph only requires story diagrams related to domain events, which query and update the domain object model.

Pedagogy for introducing story diagrams has already been developed, and it has met with success. [10] The first step is to introduce scenarios for a particular type of event, for instance a “move” in a game, and assemble sequential “before and after” object diagrams into a “storyboard.” Then, the student is ready to generalize from the storyboard and describe what it means to “move” using rules. They begin by identifying the changes made from one object diagram to the next. This usually involves deleting and/or creating links between objects. A special language provided by the tool is called a “story diagram.” It provides a way to specify a pattern that can be used to match a particular type of situation in which an object may be involved, and then identify what should be done in response. The story diagram includes pattern nodes, based on a “graph-grammar,” that specify object and link patterns. Story pattern nodes are tied together using a representation loosely based on the syntax of UML activity diagrams, which sounds complex, but is quite intuitive. We have developed a pattern node dialect specifically for this pedagogy, that derives from the one presented by Fischer et al. [13]

Story diagrams have proven the most difficult part of explicit modeling. Up until now, the student has been viewing objects and their links to other objects from the “outside.” However, they must now conceptualize that a story initially has knowledge of only the object upon which the story focuses. (Coverage of collaboration introduced through story parameters is deferred.) The student learns to develop a story pattern by following the associations of the object for a situation to be matched. Because a story pattern is similar to an object diagram, once they comprehend that they are asking rather than telling about other objects, they are able to specify patterns. The latest improvement to the pedagogy is having the student first write a natural language description of the story in the application domain, translate it to a natural language description that refers to the elements of the model domain, and finally translate this to a story diagram. Even a very complex story typically usually has only a few story patterns.

A story diagram has one entry point, and story patterns are linked together with flow lines. The fact that the associated objects are bound to names during the matching (binding) process provides generality and allows the use of the named (bound) objects in downstream story patterns, much like a variable holds a value in a traditional program. Some story patterns may query, others may update, and a single story pattern may do both. Transformations (updates) may be specified as needed by story pattern annotations. Parts of the pattern may be flagged as optional, and downstream decision nodes may query their presence to direct flow to alternative actions. Stories are simplified because in a given context, most or all of a pattern will unquestionably be matched, and subsequently need no alternative processing. At runtime, inconsistencies between what the student assumes will match (pattern components not flagged as optional), and what is actually the case, trigger entry into the visual debugging environment. The resulting flowchart does not turn into “spaghetti code” for two reasons: (1) because story patterns may notate iterated traversal of links, story diagrams most often require no loops, and (2) the system prevents construction of flowcharts that do not exhibit the same degree of structure imposed on program coded in a traditional structured language, like Java, by examining its structural properties with a directed graph algorithm.
Figure 2 is a story diagram being edited in the Objektgraph tool. The story diagram is at the upper left, the selected story pattern is at the bottom left, and the runtime object diagram being used for a test is on the right.

**Figure 2:**

The Introduction to Computer Science Course (ICSEM) Curriculum

General CS student learning outcomes in the ICSEM course will be developed in consultation with The National Educational Technology Standards for Students (NETS-S) [x6] developed by the International Society for Technology in Education (ISTE), and the standards from the Model Curriculum for K-12 Computer Science [31] developed by the Association for Computer Machinery (ACM). These widely respected standards will inform decisions about the scope of curriculum addressed within the ICSEM and enable it to be adopted widely as an introductory-level secondary course across the US. The curriculum standards addressed in the Objektgraph modules address cognitive and affective learning goals enumerated below.

**Cognitive Domain**

1. The student is able, based on a class diagram and natural language description of a situation in a familiar domain, to translate the description into an object diagram.
2. The student is able to examine an object diagram in a domain they are familiar with, and state its meaning in natural language.
3. The student is able to examine a class diagram and object diagram, and determine if the object diagram is consistent with the class diagram.
4. The student is able to design a class diagram by generalizing from a representative set of object diagrams representing concrete situations in a domain. The class diagram will be consistent with all of the object diagrams.
5. Given a class diagram in a domain with which they are familiar and a natural language situation description, the student is able to generate an object diagram corresponding to the situation and consistent with the class diagram.

6. Given a class diagram, an object diagram consistent with it that represents a situation in a domain with which the student is familiar, and a natural language description of an event type, the student is able to create the object diagram resulting from the event.

7. The student is able to work using a development process, rather than chaotically.

Affective Domain

8. The student values object-oriented modeling as a way to create a representation that can easily be understood by both computers and humans.

9. The student values teamwork in problem solving.

10. The student values looking at career choices holistically.

11. The student values creating software artifacts based on the viewpoint, thinking styles and esthetics of the community that it will serve.

Social skills such as “active listening,” “respectful questioning,” that represent the types of “soft-skills” or “non-cognitive” skills so favored among computer scientists in the work environment will also be an explicit part of the course curriculum. These social skills to be developed are based on (1) an awareness of what all learners (especially girls) need in order to work productively in a classroom, (2) research on what it takes for productive implementation of social and cooperative learning models, (3) what the CS industry name as desirable skills among employees.

The course will be designed as an 18-week experience that consists of separate curriculum strands. A strand is a set of lessons that does not use Objektgraph and may be woven into the course in parallel. Each strand might have one or two class periods. This will probably be no more than 20% of the course. Lessons will be divided over 12 sequential modeling modules and 7 strands. There will be optional projects to select from with reference materials for the student and sample implementations for the teacher. This design will increase the number of schools able to substitute the ICSEM course for existing CS offerings as it is compatible with 9-week quarter terms. For example, an Advanced Placement CS class interested in the modeling portions would be able to use just the object-oriented explicit modeling modules without having to offer the entire course. It should be noted however that during this project, it is expected that participating schools will be expected to implement the 18-week course in its entirety. The curriculum stands are illustrated in Table 1.

To make the curriculum unit available to the widest audience, the materials will be designed so that they do not require students to use Objektgraph in the classroom. Each student develops his or her own version of each exercise or project, even though the teams develop many of the components working together with paper and pencil. This will ensure that each team member is participating, provide deeper comprehension of the solution, and also a sense of “doing it on their own.” Exercises and projects will provide instructions for what to print and bring to the next class meeting for use by the team. We anticipate that the student will email their Objektgraph project file to the instructor to submit their work-product. This will save paper and allow the instructor to monitor the student’s overall progress and time spent on each task.

### Table 1: Content Strands in the Introductory Computer Science Curriculum

<table>
<thead>
<tr>
<th>Curriculum Strand</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The Field of Computer Science</td>
<td>- Overview of the computer science field. Addresses roles and team work in the computer science field. Considers overlaps between computer science and other fields (i.e., psychology, human factors, software engineering).</td>
</tr>
<tr>
<td>2. Computer Science Careers</td>
<td>- Characteristics of successful computer scientists. Introduces possible careers open to computer scientists. Compatibility with personal values, characteristics, and career/life goals.</td>
</tr>
</tbody>
</table>


3. The Evolving Field of Computer Science
- Traces relevant history to understand the present and future of the computer science field. Focus on future directions as dictated by the present innovations.

4. Software Development Practices
- Philosophies behind software development and software development tools; the differences between various programming paradigms.
- Collaboration, Teamwork, and Roles
- Discussion of approaches to software development such as open-source technologies versus closed-source and proprietary technologies.

5. Human/Computer Interface (HCI) Design
- Explores the theory behind interface design, the practical effects of quality interface design, methods for the production of quality interface design, evaluation of interface design.

6. Gender and Computer Science
- Discussion of what is known about gender and the sciences with a focus on computer science. Comparison of personal experience with those documented in the field.

7. Abstractions in Computer Science
- Hardware gates, memory, Boolean algebra, numeric representations, symbolic representations, digitized analog and lossy representations, finite state automata, and network protocols.

8. Current events related to Computer Science
- Connection of course content to daily news stories and emerging issues and events in the field and larger society (Wikipedia, YouTube, social media, blogs, Twitter, public data)

The Introduction to Computer Science Course Pedagogical Approach

The ICSEM course pedagogy leverages constructivist learning principles to create a meaningful and effective experience for all students. It capitalizes on social learning models to enhance learning, to promote students’ development of “soft skills” required for successful computer science, and to create a positive social environment supportive of girls. In the sections that follow, we describe these.

Constructivist Learning

Constructivist learning experiences promote students’ construction of meaning and “sense-making” through active engagement in meaningful learning experiences. Through intensive work “doing” computer science (i.e., object-oriented explicit modeling) we expect to provide students an effective way to develop CS competencies. Experiences exploring the general curriculum strands and with Objetktgraph will give students opportunities for sensing, feeling, thinking, sharing, and processing. They will engage students in both the cognitive and affective domains of learning. The pedagogical features of Objetktgraph are well-suited to support constructivist learning. Detailed information about selected Objetktgraph modules—which demonstrate constructivist learning—are provided in the in the section below.

Selected Objetktgraph Modules Expressed in POGIL terminology
1. Discovery and synthesis based on objects of predefined types. The model for discovery is an extant object diagram and its meaning.
2. Synthesis of application domain object models representing concrete situations. The models analyzed are natural language descriptions of the situations.
3. Discovery and synthesis of class diagrams based on generalizing from concrete object models.
4. Discovery and synthesis of storyboards/concrete use cases based on concrete event descriptions expressed in natural language.
5. Discovery and synthesis of story patterns and story diagrams based on a class diagram model, generalizing event outcomes from storyboards, and the student’s own knowledge of the application domain, and testing the story diagrams with the storyboards.
6. Team project implementing an application. The development process and roles within it are an important learning outcome. It involves developing object diagrams, generalizing object diagrams...
into class diagrams, developing storyboards for typical events, developing story diagrams
generalizing on the storyboards, and then using the storyboards as tests of the story diagrams.

7. Discovery of the need to interact with objects in various ways though an interface, leading to the
Model View Controller (MVC) pattern.

8. Synthesis of a user interface and its connection to the application domain model using the MVC
linkages supported by Objektgraph.

9. The discovery of the need for an application starting point, including the creation of an initial set
of objects. The “main” story (constructor) of an application is introduced, followed by synthesis
of a main story.

Working with Objektgraph will enable constructivist learning and help students develop computer science
competence, enhance their sense of computer science efficacy, and enable them to more fully understand
what “doing computer science” is all about. The teacher will act as a facilitator for the students, guiding
them, provoking their processing of experiences and sense-making, providing “real-time support,”
offering resources, and suggesting supports as needed.

The ICSEM Course Use of Social and Cooperative Learning Models
There are many instructional models which might be used to facilitate students’ learning in the ICSEM
course. But in this project, we leverage social learning models to meet the course and project goals. Use
of these models will promote students’ effective learning of CS content while also enabling them to
develop some of the “soft skills” required for success on teams and in the real world career environments
of CS. Though these features would be enough to warrant their use, these models will also offer support
for girls taking part in the course as they enable the creation of the positive learning environment and
collaborative learning environment that girls favor and in which girls thrive. Cooperative learning models
are characterized by five critical attributes which include: positive interdependence, individual
accountability, promote interactions, social skills teaching, and group processing. These attributes will
ensure students have motivation and purpose for their teamwork, the social skills to do so, and have
accountability individually and as a group for productive collaboration. Social learning models offer
opportunities for less structured interactions among students and provide opportunities for less
structured verbal exchanges and collaboration. Each of these models approximates different types of collaborative
environments found in the field of computer science and helps students prepare
for productive
participation in them. One cooperative learning model, Process Oriented Guided Inquiry Learning, and
one social learning model, Team Based Learning, will be used in the ICSEM course. We describe these
briefly in the sections that follow.

Process Oriented Guided Inquiry Learning (POGIL). This pedagogical method [22] teaches process
skills (such as collaboration and written expression) as well as content using an inquiry approach. This
model is attractive due to its support for discovery learning which is necessary for learning object-
oriented, explicit modeling and using Objektgraph. The theory behind guided inquiry is consistent with
constructivism—that when key ideas are discovered by the student, they are understood deeply. The
structured group learning used in POGIL reflects that in some CS workplace collaborations. Once an idea
or concept is discovered, standard terminology is immediately introduced so that everyone will be able to
understand each other.

Team-Based Learning (TBL). This model will be utilized to support student learning with object-
oriented, explicit modeling [21]. The success in TBL is thought to be derived from the healthy team
dynamics fostered by its guidelines, and the common sense intuition of the practitioner developing the
materials within those guidelines. Advocates of TBL claim significant improvements in student learning
outcomes utilizing teams. These unstructured teams approximate some of the less formal collaborative
environments in CS workplaces.

Synergy of Combining Two Models. Using these two powerful models together will be practical and
effective. It will enable students’ exposure with and opportunities to develop the skills for productive
participation in structured and unstructured collaborative groups. It will enable an effective match
between the most specialized pedagogical tool available and certain student learning outcomes. Both will make learning more interesting, rich, social, and affirming for the students involved and in turn improve their CS efficacy, CS competence, and interest in CS careers.

Project Activities
To attain the projects’ anticipated outcomes the project team will work in a coordinated way to accomplish the project activities. Figure 1.1 provides a graphic depiction of how these activities relate to the project goals and how they relate to each other. We describe these briefly here.

Develop and refine the Objektgraph software. Duane Buck (PI), working with a research assistant, will refine the Objektgraph software to ready it for use in the pilot (year 3). This work will involve “live testing” with users, modifications based on formative evaluation, and pedagogical refinement that promotes its usability and stability with students in dynamic settings. Additionally, a smartphone app deployment capability will be developed and integrated within the Objektgraph tool. Ira Diethelm will coordinate on these efforts and assist to localize the Objektgraph software for use by school participants in Germany.

Develop the Introduction to Computer Science with Explicit Modeling Course (ICSEM), course manual, materials, and multimedia. The responsibilities for developing the learning goals, curriculum modules, multimedia and other resources used in the ICSEM course will be shared by Duane Buck (who will function as the subject matter expert), Clare Kilbane (who will perform the roles of instructional designer and digital transformation expert), and Niki Fayne (who will act as project manager and instructional coordinator). An in-depth breakdown of the various individual tasks associated with this work is provided in the timeline.

Design the training program for teachers. A great deal of this project’s success rests in the hands of the secondary school teachers who will facilitate the use of Objektgraph and implement the ICSEM course. A high-quality training program (and support materials) are required to ensure their ability to deploy the outputs of the project with fidelity to the project goals. The training program for the teachers will be developed and implemented with input from the entire project team. Each member will play an essential role in expanding their knowledge of the teachers. For example, Sylvia Rimm will be responsible for creating training modules that address the best methods for recruiting and retaining girls in CS. In addition to learning about Objektgraph, the ICSEM course, and other details related to this project, we will expose them to the International Society for Technology in Education Computer Science Education Standards. [18] We believe it is important for our teacher associates to make connections between this project and the nationally recognized standards for their competences. We believe it will strengthen their purpose in our joint collaboration and their recognition of the project as valuable in a broader context.

Create recruitment strategies for high school girls: Sylvia Rimm, assisted by the project team, will be in charge of this activity. It will involve several stages for completion. First, Sylvia will conduct focus-group interviews with girls who represent the target audience for participation in the ICSEM course. In this work, Sylvia will gain greater understanding into the socio-emotional needs of girls as they relate to studying CS. With this knowledge, she will coordinate with Clare Kilbane and other team members to develop a social marketing strategy and recruitment materials for girls. It is expected that text and multimedia materials will be created and that these resources will be designed for deployment in the project and beyond. Their primary purpose will be for use in recruiting female students to participate in ICSEM at the pilot schools but care will be taken so they can be made available outside the project to aid recruitment in other NSF funded projects (such as “Teachers Attracting Girls to Computer Science: NSF grant 1042452)

Developing strategies and materials to support the retention of girls in CS education: Sylvia Rimm will also spearhead efforts to create materials to support the success of girls in the ICSEM course. As with the recruitment materials, we anticipate that materials Sylvia creates will be useful both within the context of this grant and beyond it. Sylvia will conduct focus group interviews with women enrolled in post-secondary CS programs. During these interviews, she will gain knowledge that will enable her to produce
text and multimedia materials that promote girls’ success in CS environments. We expect that these materials will aid girls with developing an awareness of some of the barriers they will face as minorities in CS education and present them with practical strategies that can help them succeed.

Recruitment of Participants: In addition to these major activities, the project team will also actively recruit 12 secondary schools for involvement in the project. It will also work to develop partnerships with the schools’ teachers and administrators that will promote successful implementation. Schools in Ohio, New York, and Germany (where members of the project team are located) will be sought for participation during the pilot phase (year 3). Several of these schools will also be invited to assist in trials of the “beta version” of Objektgraph and ICSEM course which will aim to provide formative evaluation feedback to promote instructional quality. Recruitment will target schools with the following characteristics: (a) the potential to include students from underrepresented groups, (b) willingness to actively recruit these students, (c) eagerness to participate, (d) an invested and competent teacher, (e) adequate facilities, and (f) administrative support. The schools are expected to have community and student demographics that are different from one another. They may be public or private and they may or may not have a STEM focus.

### Project Timeline

<table>
<thead>
<tr>
<th>Period</th>
<th>Activity</th>
<th>Personnel</th>
</tr>
</thead>
</table>
| **Autumn 2014**<br>Proposed project begins | • Develop initial curriculum module including object diagrams and class diagrams.  
• Release a beta version of Objektgraph with pedagogical features needed for the initial piloted curriculum module.  
• Develop assessments to measure the identified learning outcomes of the module. | DB, CK    |
|                          |                                                                          | SS, DB    |
|                          |                                                                          | CK, NF    |
| **Spring 2015 – Summer 2015** | • Pilot the initial curriculum module at local high school.  
• Revise curriculum module based on assessment of pilot.  
• Objektgraph: Develop of usage tracking for assessment, storyboard test suites, live story diagrams.  
• Develop story diagram curriculum module.  
• Develop Software Development Practices thread.  
• Develop assessments to measure the identified learning outcomes of the new module and thread. | DB, CK    |
|                          |                                                                          | CK, DB    |
|                          |                                                                          | SS, DB    |
|                          |                                                                          | CK, DB, ID|
|                          |                                                                          | ID, NF    |
| **Autumn 2015 – Spring 2016** | • Pilot three module sequence at local high schools.  
• Develop curriculum module for the GUI/MVC diagram.  
• Pilot four module sequence that enable the student to develop a complete interactive application locally and in NYC.  
• Adapt curriculum for German students.  
• Objektgraph: Respond to issues raised during pilots.  
• Objektgraph: Internationalize, initially to support German keywords and messages, and also to facilitate support for additional languages.  
• Develop the traditional method coding curriculum module.  
• Develop additional curriculum threads  
• Objektgraph: Begin development of smartphone deployment capability for student applications. | DB, NF    |
|                          |                                                                          | CK, DB    |
|                          |                                                                          | DB, NF    |
|                          |                                                                          | ID        |
|                          |                                                                          | SS        |
|                          |                                                                          | SS, ID    |
|                          |                                                                          | CK, DB    |
|                          |                                                                          | CK        |
|                          |                                                                          | SS, DB    |
Evaluation Plan

The evaluation plan for the proposal will employ a combination of methodologies and will focus on three areas: (1) context, or how the project functions within the environment/community; (2) implementation, or documentation of the evolution of the project from planning, setting up, to the actual carrying out of the project’s objectives, and (3) outcomes, or the assessment of short- and medium-term results of the project. By focusing on these three areas, data will be collected to give indication of the project’s effectiveness as well as the future sustainability and growth of the project. To ensure an effective evaluation, the Standards (1994) of Utility, Feasibility, Propriety, and Accuracy will be adhered to.

Sites. Twelve secondary schools in Ohio, New York, and Germany will be potential participants. An attempt will be made to recruit the participation of schools with diverse socio-economic populations as well as those with minority students. These schools will have varying community and student demographics. They will include both public and private high schools and may or may not have a STEM focus. There will be no minimum level of student academic performance required for the inclusion of a school in the project.

Context/Planning Component

<table>
<thead>
<tr>
<th>Evaluation Question</th>
<th>Information Source/Collection Strategy</th>
<th>Analysis/Product</th>
<th>Timeline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Who on the faculty are potential collaborators (e.g., content experts, curriculum developers, and high school (HS) faculty, and prof. dev. providers)?</td>
<td>Curriculum vitae; Networking with colleagues</td>
<td>Content analysis to determine faculty most aligned with project goals</td>
<td>Year 1 (formative)</td>
</tr>
<tr>
<td>What are participant perceptions of the training for Objektgraph-based curriculum?</td>
<td>Development of training workshop materials and evaluation forms. Based on ITTE Standards – CSE.</td>
<td>Evaluation Forms for Workshops</td>
<td>Years 1, 2, 3 (formative &amp; summative)</td>
</tr>
<tr>
<td>What factors (e.g., teacher preparedness, facility equipment, space, and resources) facilitate/inhibit sites from full implementation of the</td>
<td>Interviews with HS faculty</td>
<td>Content analysis to identify factors that need to be in place for successful implementation of the Objektgraph-based curriculum</td>
<td>Years 2, 3 (formative &amp; summative)</td>
</tr>
</tbody>
</table>
To what degree does the developed software adhere to accepted criteria for high-quality educational software?

Develop evaluation rubric based on widely-accepted criteria for quality: platform requirements, goals and objectives, the content, the pedagogy, ease of use, and costs.

Experts (software developers, CS teachers, will evaluate software using the developed rubric.

To what degree does the developed Objektgraph-based curriculum adhere to high-quality C & I standards?

Develop evaluation rubric based on widely-accepted criteria for quality: alignment, quality, design, rigor, relevance.

C & I experts will evaluate curriculum using the developed rubric.

### Implementation Component

<table>
<thead>
<tr>
<th>Evaluation Question</th>
<th>Information Source/Collection Strategy</th>
<th>Analysis</th>
<th>Timeline</th>
</tr>
</thead>
<tbody>
<tr>
<td>To what extent does the Objektgraph-based course effect all students understanding of key CS competencies?</td>
<td>Pre-Post achievement data collected (intervention and comparison groups)</td>
<td>Descriptive analysis; 2-Level HLM analyses</td>
<td>Years 2, 3</td>
</tr>
<tr>
<td>To what extent does the Objektgraph-based course effect efficacy in the area of STEM?</td>
<td>Survey data (STEM Sematic Survey) collected pre/post intervention (intervention</td>
<td>Descriptive analysis; 2-Level HLM</td>
<td>Years 2, 3</td>
</tr>
</tbody>
</table>

### Outcome Component:

This component will specifically target the following questions:

<table>
<thead>
<tr>
<th>Evaluation Question</th>
<th>Information Source/Collection Strategy</th>
<th>Analysis</th>
<th>Timeline</th>
</tr>
</thead>
<tbody>
<tr>
<td>To what extent does the Objektgraph-based course effect all students understanding of key CS competencies?</td>
<td>Pre-Post achievement data collected (intervention and comparison groups)</td>
<td>Descriptive analysis; 2-Level HLM analyses</td>
<td>Years 2, 3</td>
</tr>
<tr>
<td>To what extent does the Objektgraph-based course effect efficacy in the area of STEM?</td>
<td>Survey data (STEM Sematic Survey) collected pre/post intervention (intervention</td>
<td>Descriptive analysis; 2-Level HLM</td>
<td>Years 2, 3</td>
</tr>
</tbody>
</table>
To what extent does the Objektgraph-based course engage students in non-cognitive competencies? Analyses

Focus group interviews conducted three times during implementation *Qualitative analysis of focus group interviews *C.R. Observations Years 2, 3

Year 3 Pilot: The pilot design will be a concurrent triangulation mixed method design where classrooms will be randomly assigned to treatment (Objektgraph pedagogy) and comparison (traditional pedagogy) conditions using a cluster-randomized design, with students nested within classrooms.

**Instrumentation: Quantitative – Student (Level 1)**

*Achievement.* The achievement test will assess the extent that students can perform key competencies related to object-oriented explicit modeling and computer science practice. Items will measure specific cognitive modeling competencies addressed in the ICSEM curriculum.

*STEM Dispositions and Career Interests.* STEM Semantic Survey is a semantic differential instrument designed to assess students’ perceptions of STEM disciplines plus interest in STEM as a career. The STEM Semantic Survey is comprised of five subscales with five items each: (1) Perception of Science, (2) Perception of Technology, (3) Perception of Engineering, and (4) Perception of Mathematics. Reliabilities for each subscale have been shown to be greater than 0.80 with the full scale reliability being >0.90 (Tyler-Wood, Knezek, & Christensen, 2010).

**Quantitative - Teacher and Classroom (Level 2).** Teachers will complete a brief demographic questionnaire that will include questions regarding their gender, education, and number of years teaching.

**Qualitative.** Semi-structured student focus groups interviews in intervention classrooms will be conducted. Interviews will center on the students’ experiences with the Objektgraph tool, their experiences working in a team setting, and their perceptions of and beliefs about careers in computer science fields.

**Data Collection**

**Quantitative – Student (Level 1).** All student outcome data will be collected twice: once prior to the beginning of the intervention and once at the conclusion of the intervention. Data from the comparison group will be collected on the same schedule.

**Teacher/Classroom data (Level 2).** Prior to the intervention, teachers will complete the demographic survey.

**Qualitative.** Focus group interviews will be conducted three times over the course of the intervention: prior to the intervention, during the intervention, and at the conclusion of the intervention.

**Analytic Model**

**Quantitative.** Data are considered to have a hierarchical structure when persons are nested within organizational units or multiple observations are nested within persons, resulting in an interdependency of scores for units within clusters [23]. A series of 2-level multilevel models using hierarchical linear modeling version 6.06 will be employed. The dependent variables of interest will be students’ achievement performance, self-efficacy in STEM, and interest in STEM. For each dependent variable, a two-level model (Level 1 being student and Level 2 being teacher/classroom) with restricted maximum-likelihood estimation will be used given the small Level 2 sample size.

**Qualitative.** Two separate researchers will analyze the interview data employing a contrasting strategy as a way to increase credibility of the findings. The approach employed by the first data analyst will be deductive analysis [14]. The purpose of this data analyst’s work will be to begin with *a priori* hypotheses and confirm or disconfirm their existence. The second data analyst will employ a constant comparative method as described by Corbin and Strauss [6]. This more naturalistic approach will allow for the development of hypotheses about the relationships among the treatment, students, and classroom elements.
References:


