Getting Undergraduates Involved in your Research by Designing Well-scoped Undergraduate Research Projects

Lori Pollock
University of Delaware

SIESTA 2025

Attribution

This presentation and associated resources were adapted from materials for the Computing Research Association's UR2PhD program by Kelly Shaw (Williams College) which were modified by Christine Alvarado (UC San Diego) and Lori Pollock (Univ. of Delaware).

Goals of this Session

Learn how to scope research projects for undergraduates whom you are mentoring or plan to mentor as part of your graduate research training

- Questions to guide project planning
- Steps for creating actionable deliverables
- Strategies for mapping skills and needs to a project

Session Materials

https://tinyurl.com/SIESTA-Pollock-folder

Why Get Undergraduate Researchers Involved?

Benefits

- Additional contributors can help advance your research project
- Inspire and prepare the next generation of researchers
- Instill skills and experience in students for their careers
- Improve your mentoring skills
- Recruit future graduate students

Concerns

- UGrads work more slowly than you
- UGrads require frequent guidance and oversight
- UGrads may disappear quickly, abandoning projects before completion
- Students might have mismatched background/skills for a project
- Bad research experiences turn UGrads away from research

Why Get Undergraduate Researchers Involved?

Planning Unlocks Benefits

Design projects to advance your research

Consider student skills in project design

Create well-scoped projects for given timeframe with frequent, concrete deliverables

Stitch multiple student research experiences together

Step 1: Determine a Project Area

Approach 1

Start a new exploratory project related to other research group efforts

Approach 2

Join an existing group project and work on an independent part

Suggestions for New Exploratory Projects

Projects with an initial hypothesis that can be **evaluated empirically** and whose results can lead to new questions to explore

Examples:

- Replicate an existing study and extend it using new data with different characteristics
- Study state of the art software tools on a new data set with different characteristics than the software was originally designed for
- Conduct a human subject study on a specific user interface or tool
- Modify and evaluate existing software to implement a new idea based on a hypothesis
- Analyze a new data set for specific characteristics
- Evaluate a system created for automating a task

Suggestions for Joining an Existing Project

Projects may involve **starting in the middle** of an existing project

Examples:

- Data collection or generation for existing system or study
  - Perform interviews or administer survey
  - Collect data using existing software on new data inputs
  - Create scripts to generate data
- Analysis of already collected data
  - Write scripts to retrieve and analyze existing data
- Visualization of already collected data
  - Write scripts to retrieve and visualize data
- Implementation of a new instrument
  - Add new functionality to an existing system
  - Create a software implementation of a theoretical algorithm
  - Create a new human subject survey
  - Install new data sets or applications for use with existing system

Both Approaches Involve Subsets of Research Activities

- Search literature for papers
- Read technical papers
- Create instrument
  - Modify existing and/or create new
- Collect or generate data
- Analyze data
- Visualize data
- Share the research

Tips for Research Project Design


Design authentic research projects that matter to your research group


Design projects not on the critical path


Start at a theoretical level students can understand


Draw on skills students already possess or can learn quickly


Design projects with clearly defined, frequent milestones


Keep scope modest and flexible enough to be simplified or extended


Create low-risk projects with a good chance of producing results within given time frame

Let’s Take a Look at the Worksheet and an Example

https://tinyurl.com/SIESTA-Pollock-folder

Step 1: Brainstorm a meaningful, non critical research question/project

- Do you have some **existing research results that are not fully explained** and you would like to better understand by collecting and analyzing more data?
- Do you have some e**xisting research results that have made you curious** about how your tool or approach would **apply in a different setting or for different inputs?**
- Do you have **existing collected data that you have not found time to analyze** or visualize for specific characteristics, where the results could lead to deeper understanding of your approach or point to new problems to explore?
- Is there a small, relatively **straightforward artifact** (e.g, survey, software feature) **that you need implemented and evaluated t**hat builds on an already existing system or process?

Software Engineering Example

Step 1:

- Do you have some existing research results that have made you curious about how your tool or approach would apply or work in a different setting or for different inputs?

Our research group invented techniques to automatically generate descriptive comments for Java code, both at the method level as a summary of the method, as well as internal comments for describing smaller segments of the code. **We are curious to know** whether similar techniques **could be applied effectively with some possible adaptation for** automatically generating commit messages to summarize the changes in code during development.

Step 1: Brainstorm a meaningful, non critical research question/project

Step 2: Design the Research Project's Scope

Turn the research problem into explicit, actionable steps

Step 2.1: Make the research problem precise and concrete:

What is the precise research question you want this research project to answer?

Software Engineering Example

Step 1: Our research group invented techniques to automatically generate descriptive comments for Java code, both at the method level as a summary of the method, as well as internal comments for describing smaller segments of the code. We are c**urious to know** whether similar techniques **could be applied effectively with some possible adaptation for** automatically generating commit messages to summarize the changes in code during development.

Step 2: We want to answer the precise research question:

Step 2.1. What are the characteristics of commit messages that an automatic generator should strive to achieve, based on human readers and writers of commit messages, and how likely is it that an extension of an existing tool could achieve those characteristics?

Software Engineering Example, continued

Step 1: *Can our Java code comment generation be applied effectively with some possible adaptation for automatically generating commit messages to summarize the changes in code during development?*

Step 2.1: *What are the characteristics of commit messages that an automatic generator should strive to achieve, based on human readers and writers of commit messages, and how likely is it that an extension of an existing tool could achieve those characteristics?*

Step 2.1 Subquestions/subgoals:

- What do humans write in commit messages so we can understand what human writers of commit messages believe should be documented?
- What do human readers find most useful about commit messages in helping them understand the code changes and their intent?
- How promising is the existing state of the art tool, DeltaDoc, which analyzes diff files between two different program versions and generates a pseudocode summary of the changes, as a basis for an approach to generating natural language English phrase summaries as commit messages?

Your turn! (10 minutes + 5 minutes for share out)

Think of a new project or an extension of an existing project that is:

- Related to your own work or that would move your work forward
- Suitable for undergraduates

See slides 8 and 9 for guidelines and examples.

⇒ Complete step 1 and 2.1 on the worksheet based on your project idea.

Step 2: Designing the Research Project's Scope

Turn the research problem into explicit, actionable steps

Step 2.2: Identify measurable deliverables or subgoals that are contributions that move the project forward

Software Engineering Example, continued

Step 2.2: Identify Measurable deliverables.

- **Create a data set** of human-written commit messages from a variety of open-source software repositories to represent a large range of application domains.
- **Create scripts to** measure the non-linguistic metadata related to the commit messages such as their length in lines, how many authors were in each project, and how many commits each author made. Then, develop infrastructure to **analyze the linguistic properties** of the human-written commit messages. Run the scripts, gather data on the characteristics of human-written commit messages, and analyze the data on human-written commit messages.
- **Create an online survey for users of software repositories** to learn about how they use commit messages, what kinds of commit messages they find useful, and to present an initial model of output for natural-language commit messages. Distribute the survey to a group of users of software repositories. Analyze the results of the survey.
- **Download, install, and be able to successfully run the tool on the data set.** Using the characteristics of useful commit messages discovered in the initial studies as a guide, **use the tool to generate commit messages and analyze the characteristics of the generated pseudo code** as a basis for English phrases.

Your turn! (10 minutes)

Define the measurable deliverables of your project.

⇒ Complete step 2.2 on the worksheet.

Neighbor Share Out

Share your progress with one other person.

What are you struggling with?

What do you think will work well?

Step 2: Designing the Research Project's Scope

Step 2.3: Identify knowledge and skills required to complete the project.

Software Engineering Example, continued

Step 2.2:

- Create a data set of human-written commit messages …

Step 2.3: Identify the knowledge and skills required to complete the project

- **Use of software repositories.**
- **Ability to write small scripts** in python **to extract characteristics** from a file of commit messages. Ability to identify verbs and direct objects in commit messages manually.
- **Use of NLP library to automatically extract parts of speech from commit messages.**
- **Ability to write unbiased questions for an online survey** to gain human opinions of commit messages using Likert scale answers. Ability to create an online survey and gather data from it. Straightforward analysis of results from an online Likert-scale survey using spreadsheets.
- **Ability to download, install and run a tool successfully on a set of data.** Manually conduct analysis of generated pseudo code as a basis for English phrases in verb-direct object format.

Step 3: Pre-research Mentor Preparation

Step 3.1: Identify complicated tasks for **you** to complete in advance for the undergraduate mentee

Step 3.2-3.3: Identify resources to help students develop needed knowledge and skills

Software Engineering Example, continued

Step 1: *Can our Java code comment generation be applied effectively with some possible adaptation for automatically generating commit messages to summarize the changes in code during development?*

Step 2.1: *What are the characteristics of commit messages that an automatic generator should strive to achieve, based on human readers and writers of commit messages, and how likely is it that an extension of an existing tool could achieve those characteristics?*

Step 2.2: Identify measurable deliverables or subgoals

Step 2.3: Identify knowledge and resources required to complete the project

Step 3.1: Identify complicated tasks to complete in advance

- Set up a document that describes the locations of any example scripts, surveys, and data sets relevant to this project, and an example research diary, description of the project based on this worksheet, and tentative calendar timeline for the students.

Step 3.2-3.3: Identify resources to help student develop needed knowledge and skills

- Identification of existing similar scripts, papers, talks, documentation

Your turn! (10 minutes)

- Identify the knowledge and skills students need to complete the project.
- Identify the complicated tasks you will need to complete in advance and the specific resources a student will need to complete their tasks.

⇒ Complete steps 2.3 and 3 on the worksheet in the time you have.

Whole Group Share Out and Questions So Far

What issues or themes have come up so far?

What questions do you have so far about this process?

What questions do you have about other undergraduate research mentoring topics?

Homework: Revise and get feedback!

Based on your discussions with each other and with the whole group,

revise your responses to the worksheet so far (steps 1, 2.1 and 2.2).

Step 4: Creating a Project Timeline

- Create a dependency graph of tasks
  - Including concrete deliverables and skill learning
- Realistically estimate time required for each task

- Map tasks to a timeline using a Gantt chart

- Adjust deliverables to fit work into time frame
  - Identify repetitive tasks
  - Employ team of student researchers to parallelize repetitive tasks
  - Turn subgoal into a final goal

Initial Subproject Schedule for SE Example

Discuss how to read technical papers critically

1 wk — Learn to critically read related paper(s)

Read 1-2 related papers on their own using a guide sheet

Discuss critical analysis of paper

2 wks — Learn to write scripts to analyze commit messages

Read, adapt & test existing script slightly to count length

Write & test small scripts for each characteristic of interest from human-written messages

1 wk — Create initial data set of commit messages for analysis

Decide on source for commit messages

Create/Identify script to extract commit messages

3-4 wks — Run scripts and analyze results on real data set of human-written commit messages

On Your Own - Create a Tentative Project Timeline

Quick Tips for Mentoring Undergraduates

- Give students frequent and early wins
- Help students learn when to ask for help
- Help students develop resilience to failure and understand that research is open-ended
- Work with students to adapt timeline and goals as research unfolds
- Consider working with groups of students
- Consider who else in your lab can provide support to students
- Set expectations, including for meetings and communication
- Get to know students as multifaceted people

References

- Shanahan, Jenny Olin, et al. "Ten salient practices of undergraduate research mentors: A review of the literature." Mentoring & Tutoring: Partnership in Learning 23.5 (2015), 359-376.
- Craig, Norman C. "The joys and trials of doing research with undergraduates." Journal of Chemical Education 76.5 (1999): 595-598.
- Shellito, Cindy, et al. "Successful mentoring of undergraduate researchers." Journal of College Science Teaching 30.7 (2001): 460-464
- THE LEADERSHIP ALLIANCE Mentoring undergraduates in summer research programs. http://www.theleadershipalliance.org/matriarch/documents/mentor guide complete.pdf
- Barrow, Michael, Shelby Thomas, and Christine Alvarado. "Ersp: A structured cs research program for early-college students." Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education. 2016.
- Izhikevich, Katherine, Kyeling Ong, and Christine Alvarado. "Exploring Group Dynamics in a Group-Structured Computing Undergraduate Research Experience." Proceedings of the 2022 ACM Conference on International Computing Education Research-Volume 1. 2022.

Congratulations and Welcome!

to the community that is inspiring and developing the next generation of computing researchers!

While

- ★ Deepening your own understanding of how to mentor
- ★ Learning an essential skill for long-term career success
- ★ Strengthening your own communication and research skills