Security Assurance Case Design Tool

Project Plan

Team Dec1718

Client &/Advisor:
Lotfi ben Othmane

Team Members &/Role:
John Koehn: Concept holder
Brandon Huegli: Team Lead
Chiakang Hung: communication
Jordan Lawrence: Webmaster

Contact:
1718dec@iastate.edu

http://dec1718.sd.ece.iastate.edu

Revision: 21 April 2017

Table of Contents

1 Introduction

1.1 Project statement

We are developing an eclipse plugin that will be used to design and manage security assurance cases. Users will be able to create, edit, and save/load assurance case diagrams, which look similar to a UML diagram in structure. Nodes of the diagrams represent specific claims or evidence, and will be tied to portions of the code and unit tests for the project the plugin is loaded into. Claims or evidence that are changed or are no longer valid will be marked visually for the user.

1.2 Purpose

Companies that need to meet security standards spend a large amounts of energy after each iteration of the product making sure they still meet them. A security assurance tool will allow the developers to know when they update the code that they still meet the security standards. Therefor, they no longer will have to go through the code changes in a large group to verify they still meet them. This would allow for faster updates of their products and save developers time. It will also give them confidence in their updates and that the application is secure.

1.3 Goals

1. Create an easily accessible eclipse plugin that any developer could download
2. The gui will be easy and intuitive to use for the developer. They should be easily able to link code in different projects that meet different security cases and be able to develop test to verify they meet them.
3. The plugin will give visual feedback to the developer if they are no longer meeting a security case
4. The plugin will actually be helpful to developers and speed up the development process
5. The plugin has documentation that a developer could understand to learn how it works and be able to use successful
6. Help Dr. Othmane further his research and be able to use the plugin in the future

2 Deliverables

- A plug in importable into any existing eclipse project.
- A GUI for creating and modifying assurance case diagrams.
- A system where diagram nodes are tied to code snippets within the project, as well as unit tests.

3 Design

The primary user interface of the plugin will be developed using Eclipse's Graphiti framework. It is primarily designed for user defined diagram editors as part of an eclipse plug in, and so should be well suited to our project. Code will be written in Java to support the background features of the diagrams.

3.1 Previous work/literature

Assurance Case General Information:

These documents were read over by the team to gain some level of understanding on what a security assurance case looks like, and what the project is intended to do for a user.

Research paper that discusses how a security assurance tool should be implemented and the benefits of doing so. The paper also gives examples of how security assurance diagrams should look and relates them to real world examples. ("Goal-Oriented Co-Engineering of Security and Safety Requirements in Cyber-Physical Systems")

An article from Carnegie Mellon University discusses the design of assurance cases for software tools. It covers the basics of the different requirements of assurance cases being represented as a graphically flowing argument. This gave us a lot of good insight as an overview to help us understand the goals of the project.. ("Assurance Cases").

A government article on security assurance cases in software from US-Cert examines much more in-depth how to create these diagrams, allowing us to better understand exactly what the diagrams we'll be able to create will look like. There were also some good example cases we can use as a basis for our own testing purposes and goals. (Goodenough)

Assurance Case Software:

This software was looked over by the team, and was supposed to be existing software that did something similar to our project. However, the team was unable to actually use it. (CertWare)

GUI Libraries:

These are the GUI libraries the team evaluated for use in our project.

Name	**Pros**	**Cons**	**Evaluation**
JGraphx	Would be able to meet the requirements of the project. Would be easy to begin development with.	Built on Java Swing. Not eclipse specific. Dated aesthetic.	We decided that while this framework was a candidate as it met our requirements, it was not the best option available.
Eclipse Graphiti	Built for making eclipse plug ins. Aesthetically the most pleasing. Acceptable documentation and tutorials.	Team lacks experience with plug in development.	The team has chosen Graphiti for development of the project.
JGrapht	Would've been likely able to meet requirements in most ways.	Built on Java Swing. Not intended for the exact style of diagram necessary.	This framework was not considered a candidate compared to the previous options.
Graphviz	Able to draw diagrams needed.	Poor available information. Doesn't appear to be built for Java or Eclipse.	As with JGrapht, this library was not considered a candidate.

3.2 Proposed System Block Diagram

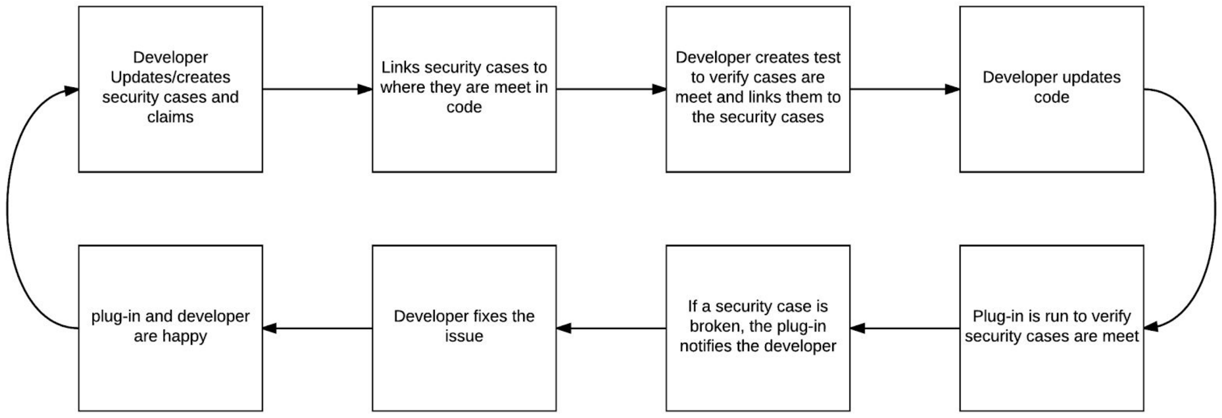

Developer Updates/creates security cases and claims
Links security cases to where they are meet in code
Developer creates test to verify cases are meet and links them to the security cases
Developer updates code
Plug-in is run to verify security cases are meet
If a security case is broken, the plug-in notifies the developer
Developer fixes the issue
plug-in and developer are happy

Fig. 1 System Block Diagram

3.3 Assessment of Proposed Methods

We have several different tools we could use to build the eclipse plugin. JGraphx would allow us to use java swing to create the eclipse plugin and gui. However, JGraphx is old and outdated. This makes it undesirable to use for the project. We could also use eclipse EMF which would allow us to build the GUI from the ground up using different low-level graphic libraries. However, doing so would cost an immense amount of time.

The approach we will take is to use eclipse graphiti to create the interactive gui that would allow developers to build complex diagrams. As shown in fig. 1, the developer can create an assurance case, and use our tool to create the diagram, and verify their code all in one application. Graphiti is meant for plugin development and allows us to create a modern looking ui. The library allows has plenty of capability and support to make the interactive gui that the project requires.

3.4 Validation

We will validate our plugin by ensuring it is able to create existing assurance cases properly, and support them with the specified feature requirements. We will also use it in conjunction with other assurance software (I.E., ZenCart) to make sure it meets reasonable requirements for existing conventions. Further, we will attempt to get user feedback from our actual potential user base.

4 Project Requirements/Specifications

4.1 Functional

1. Ability to add/remove graphical elements to assurance case diagram

2. Multiple graphical elements to choose from
    a. Case claim
    b. Evidence
    c. Edges

3. User interaction with elements
    a. Move elements around the chart
    b. Edit text in the element
    c. Draw connections between elements

4. Open submenu for each node to connect underlying code or tests to elements
    a. Open currently linked code sections
    b. Select new code sections to link
    c. Update diagram node in response to code changes appropriately

5. Visually alert user when claims/evidence nodes are no longer valid.
    a. Propagate invalid status to parents.
    b. Failed tests change element color.
    c. Changed, potentially vulnerable, code changes element color.

6. Save/Load diagram as part of the Eclipse project

7. Import and export diagram files into the project
    a. Use human readable formats, so diagram files can be edited directly.

4.2 Non-functional

1. Visually pleasing and professional interface

2. Intuitive and easy to use.

4.3 Standards

We will be following the IEEE code of ethics for this project. Our tool is meant to help developers make more secure software. If our tool has a bug that results in it not notifying a developer that a security case is no longer being meet then we would be compromising a software’s security which could cause damage.

The plugin will also meet established conventions on the design of safety assurance cases as specified by Carnegie Mellon University, and governmental agencies such as NASA and the FDA.

5 Challenges

- The team is not knowledgeable about eclipse plug in development, assurance cases, or Java based GUI frameworks outside of Swing, and will need to spend additional time researching these topics while supporting ongoing development.

- While we anticipate the selected GUI library will meet our requirements needs, it may turn out to not work as well as anticipated, or otherwise cause unforeseen issues, creating a risk after development has started. Specifically, while developing the GUI, we are unsure if Graphiti will later allow the interactions we need to add for the other features of the plugin.

- We’ll need to be able to interface our GUI tool with another security tool, OWASP, to achieve functionality goals. Depending on the constraints of the two tools this may or may not cause issues. Research will need to be done into OWASP.

- Eventually we have to map security claims to actual code. The claims and code need to communicate in order to determine validity and achieve a secure program. The team has no experience in this type of interaction.

6 Timeline

We have divided the timeline to every two weeks.

6.1 First Semester

#1, 1/29~2/11

- Research technology for safety assurance
- Evaluate pros and cons of each technology

#2, 2/12~2/25

- Finish Project Plan V1
- Write requirements for the app

#3, 2/26~3/11

- Get familiarized with Graphiti
- Work on Design Doc V1
- Add functions to the app: adding/removing graphical elements to assurance case diagram

#4, 3/12~3/25

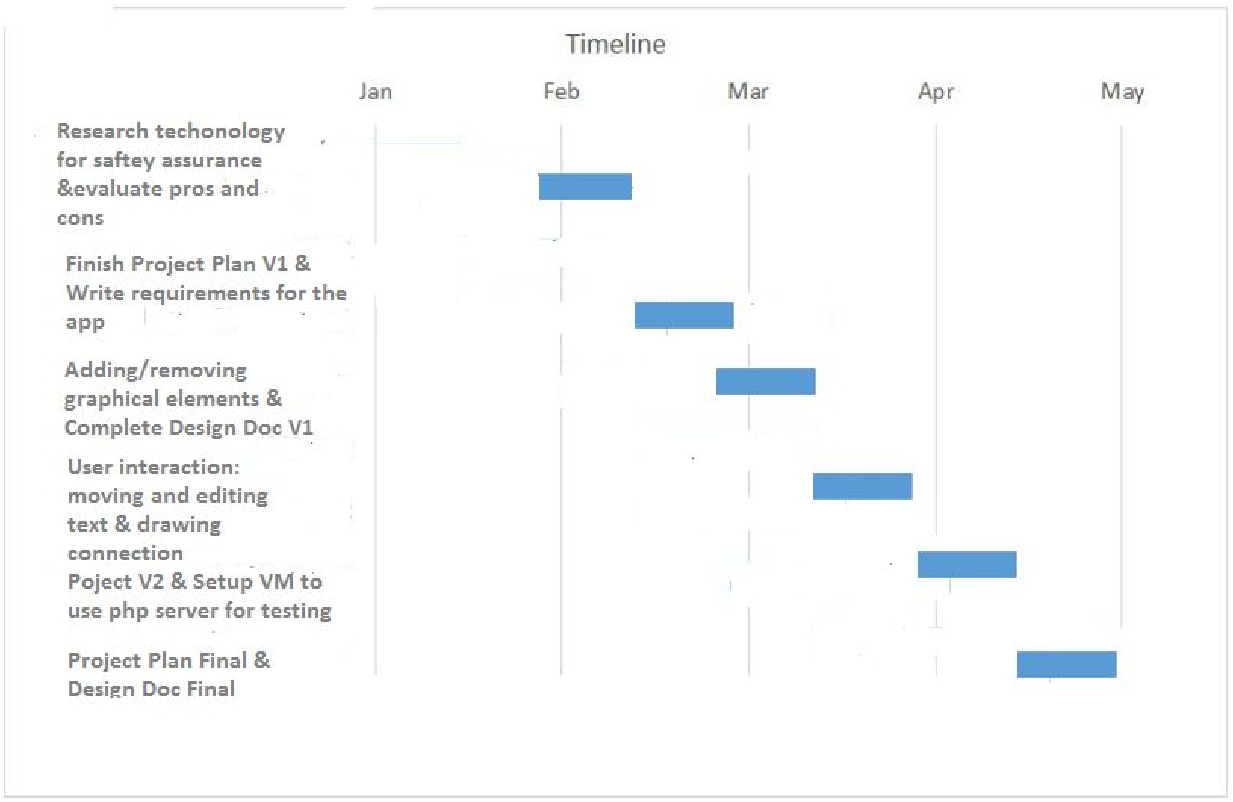
- Work on User interaction with elements: moving and editing text, and drawing connections between elements
- Continue working on adding functions to the app.

#5, 3/26~4/11

- Work on Project Plan V2.
- Continue Working on User Interface Interaction
- Set up VM to use a php server that we can do local testing on

#6, 4/12~4/25

- Work on Project Plan Final
- Work on Design Doc Final
- Finish work on User Interface Interaction



Spring Semester Timeline

6.2 Second Semester

Next semester, we will be focused on adding additional features onto the GUI we have designed this semester. These features include the ability to link code within an eclipse project to nodes of the diagram, along with test cases, and visual cues. This will allow the diagram to serve as a complete reference for the users' assurance case. The goal is to make our plugin able to replicate results and replace existing assurance case software in a more intuitive, easy to use way.

From there, depending on how time allows, we will expand and improve with Dr. Othmane's continued guidance, though we are unable to give a specific timeline or any additional details at this time.

7 Conclusions

We will develop an eclipse plugin for creating and managing security assurance cases, associating diagram elements with code and unit tests. The main portion of the application will be developed using the Eclipse Graphiti framework, with Java code implementing supporting features. It should be easy to use, and importable into any existing Java project.

8 References

Assurance Case General Information:

Goodenough, John, et al. "Arguing Security - Creating Security Assurance Cases." Carnegie Mellon University, 4 November 2017. https://www.us-cert.gov/bsi/articles/knowledge/assurance-cases/arguing-security-creating-security-assurance-cases. Accessed Feb. 2017

Ponsard C., Dallons G., Massonet P. (2016) Goal-Oriented Co-Engineering of Security and Safety Requirements in Cyber-Physical Systems. In: Skavhaug A., Guiochet J., Schoitsch E., Bitsch F. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2016. Lecture Notes in Computer Science, vol 9923. Springer, Cham

"Assurance Cases." *Software Engineering Institute*. Carnegie Mellon University, 2017, http://www.sei.cmu.edu/dependability/tools/assurancecase/. Accessed Feb. 2017.

Assurance Case Software:

Certware. Nasa, 2012. https://nasa.github.io/CertWare/. Accessed Feb. 2017

GUI Libraries:

Naveh, Barak. Jgrapht. http://jgrapht.org/. Accessed Feb. 2017

Graphiti. The Eclipse Foundation. https://eclipse.org/graphiti/. Accessed Feb. 2017

Jgraphx. https://github.com/jgraph/jgraphx. Accessed Feb. 2017

Ellson, John, et al. *Graphviz*. http://www.graphviz.org/. Accessed Feb. 2017