GUI for upgrade checking engine integrated with Eclipse IDE.

Formal Verification and Security Group
Prof. Sharygina and Grigory Fedyakovich

Model Checking [1] is a promising scientific approach to check correctness of software. It is fully automatic approach to decide whether a program safe or not. For a software developer working on a project, it is highly convenient to use verification technologies to verify that his code satisfies certain properties. There are already existent verification systems to do it [2,3,4]. To be sure that the program is correct, it is needed to chose a verification system, set up environment for the system to be able to parse the code and analyze it, and just "push the button" for getting a result "program correct, or not".

Upgrade Checking [5] is a branch of model checking which is aimed at checking software based on its incremental changes. Once the program (later called "initial version") is confirmed correct after the check, it is possible to store some of the intermediate information and thus reuse the invested effort for verification of the next version of the program (later called "upgraded version"). Without this technique, it would be necessary to check the new version from scratch, and therefore to re-do significant part of the work.

In the Formal Verification and Security Lab of University of Lugano, we develop an Upgrade Checker FunFrog [6], which uses OpenSMT [7] solver. First, in order to verify an initial version of a given program, FunFrog encodes the program sources into propositional formula and passes it to OpenSMT. OpenSMT automatically decides, whether the formula is satisfiable or not. The case when the formula is unsatisfiable means the corresponding program is correct. The proof of unsatisfiability is enough to generate and store function summaries, which are reused to check an upgraded version later. Here, function summary is a propositional formula capturing the relevant facts from the function body.

The upgrade checking algorithm, employed by FunFrog, is simple and elegant. Unlike re-encoding the new sources to a new propositional formula, and passing it again to OpenSMT, FunFrog starts working only with certain parts of the sources which were syntactically modified. It finds all functions, whose bodies were modified and matches each function with its old summary. Then the upgrade checking is reduced to local check, whether the old summary still valid for the new function.

The next stage of implementation is essentially to integrate FunFrog with well-known and widely-used Integrated Development Environments, such as Eclipse. These IDEs are developed to simplify software development process. They have many opportunities for handling multiple files with sources of a project, debugging, testing, refactoring and software versioning. Using these IDEs, a developer does not need to use any external development tools.

But formal verification systems are poorly presented in IDEs. There is only a few expensive commercial systems to search for bugs, such as [8,9]. As for free and open source systems, only CBMC is known. The plugin for CBMC to Eclipse is available [10], but it allows only standalone verification. Getting a FunFrog plugin to Eclipse would be very useful for developers, it would allow them to check their programs just by "pushing button" in the same IDE, as they use for development and are already familiar with.

Software versioning systems, integrated to IDE are quite essential and useful for upgrade checking. Once a version of a project is committed, it can be considered as an etalon and a pattern for an upgraded version to compare with. When a developer is going to commit the version, he can run FunFrog to become sure, his code is safe and well-developed. Function summaries, generated at this run, can also be stored in the versioning system. Later, when an upgraded version is getting completed, developer can run FunFrog again. In this run, FunFrog will operate with the current source code and function
summaries, taken from the last commit. The answer, obtained from this check, will help the developer to proceed further. If the upgraded version is safe, he can commit it. Otherwise, the concrete bug location is returned to him, and he should fix it first, and then repeat the check.

We are sure, the proposed technology can be easily implemented, because the main ingredients already exist: the Eclipse IDE with available source code and APIs, and FunFrog, which so far can be run only from command line. We believe, such an integration will open the doors to formal verification for the majority of software developers, who are unfortunately still not familiar with this technology. And this will increase the quality of software, being developed.

There are two main benefits for a student who is going to implement the proposed technology. First, the student will become familiar with formal verification and state-of-the-art verification systems. It will give the young researcher a taste, what kind of research is being done in the modern Computer Science. No theoretical background in Formal Verification is required. Second, the student will earn an important experience in software development. Eclipse IDE is known in developers community, and experience in working with its source code will indicate the student is familiar with modern trends in software development. For this, initial experience in Java programming language is required.

References:

Project Description

The goal of this project is to create a set of interactive visualizations, called Bard, able to “tell the story” of a software system through the data extracted from its mailing lists.

In the first period, the student will familiarize both with the framework we devised to explore email data (i.e., Miler [1] and Remail [2]) and with HTML5 for implementing visualizations.

Bard must be able to gather the data from the document-oriented database (CouchDB [3]) at the backend of Remail, which provides access to its data through RESTful JSON APIs. Bard must work both as a plugin in Eclipse and as a stand-alone web application.

In the second period, the student will investigate the state of the art in visualizing email data and will use it as the basis to experiment and implement the novel interactive visualizations of Bard.


Cloud computing is a new paradigm that allows system components, like servers, to be decoupled from the underlying hardware via virtualization techniques. Clouds enable the dynamic and on-demand management of system resources in the form of virtual machines (VM). Notables feature of Clouds are: allocation, de-allocation, replication and live-migration of running VM.

USI participates to the development of the first European Cloud computing infrastructure in the context of the Reservoir EU Project.

The aim of this work is to re-design part of the Reservoir infrastructure and to develop new components in it. The infrastructure will be deployed in the University servers and it will used for carrying on important research activities.

We are looking for a motivated student that will have a chance to improve his/her knowledge on Cloud computing, Java and/or Ruby. The student will be coached while:

1. Formalizing the new requirements for the Cloud infrastructure
2. Identifying and substituting (or re-design) the “old” infrastructure components
3. Designing new components of the Cloud
4. Designing and running a set of tests using a model-driven tool for experiment automation

Sponsor and Mentors

This project is sponsored by Prof. Mauro Pezzé and will be mentored by Dr. Giovanni Toffetti Carughì and Alessio Gambi.
Automated test case generation is a challenging problem. The effort towards test automation has produced many tools that provide different degrees of automation of core testing activities, but there are many expensive testing activities that are still only partially automated. Current project focuses on the problem of automating one of these human-intensive activities: the generation of integration test cases.

In the current project we work on a novel idea to automatically generate new integration test cases using information available in existing unit test cases. Thus our approach enhances the existing sets of test cases automatically generating new integration test cases.

The development of framework for the automated code analysis and generation is thus necessary to experiment with large industrial applications to confirm latest results. We expect to generate additional test cases that may reveal hidden problems (if any) and we hope to investigate the possibility of generating system test cases from integration ones, thus further enhancing the technique.

The aim of this work is to extend an existing framework for source code analysis and test case generation. The framework is implemented as an Eclipse plug-in for Java source code analysis, including abstract-syntax-tree manipulation, and test code generation in JUnit format.

We are looking for a motivated student with a strong knowledge in Java programming, unit-testing methodologies (familiar with unit testing frameworks, such as JUnit), and good understanding of principles underlying the source code syntax and structure (abstract-syntax-tree manipulation).

Students familiar with or eager to have programming experience with the Eclipse Java Development Framework (JDT) and Eclipse Plug-in development are encouraged to apply.

The student will have a chance to improve his/her knowledge on the source code structure, analysis, code manipulation and code generation. As well as learn how to do experimental computer science.

The student will be coached while:

- Designing extensions for the framework to deal with various Java language constructs.
- Designing and running experiments to try new test generation techniques.
- Devising and conducting experiments for evaluating approach involving real open-source Java projects.
- Analyzing collected data and produce a brief report.

Mentor: Konstantin Rubinov
Sponsor: Mauro Pezzè
Mr.Edu 2.0

**Project Description.**
This research project is positioned in the field of educational technology design. The work will be focused on the design and development of a web-based authoring tool with educational purposes, which will allow groups of school children to create and share digital stories. The application is in part already developed and is called Mr.Edu. At the moment Mr.Edu is a tool to collect, categorize and retrieve multimedia contents such as images, text, audio and video files to help children in the creation of digital narratives. Mr. Edu is still a prototype and it is the result of a master thesis. The code is published and available online on Google code to be used for further development. The prototype is developed using these technologies: Python, Django, SQLite, RDF (through Python's rdflib and rdfstore), HTML5, CSS3, JavaScript.

**Duties.**
The aim of the project is to improve the prototype by adding some functionalities for supporting multimedia story creation by means of a timeline on which different media assets can be mapped (e.g. picture, text, audio, video, etc.). This project is part of a bigger research conducted for the last two years. The requirements for Mr. Edu have been elicited within a field study in an elementary school in Lugano. We will continue during the next weeks to refine these requirements by conducting additional studies with children and teachers at school. The methodologies used in the projects fall within Human-Computer Interaction approach and are inspired by cooperative design.

**Benefits for the student.**
The students can acquire knowledge regarding the application of a HCI approach in the design and development of an interactive application that includes wireframing, prototyping, and testing. The students will work on an active and engaging environment working in a multidisciplinary team made of designers, human factor experts, computer scientists and information scientists. This could enhance also her/his skills to collaborate with a work group and to work in a real context of use.

**People to contact**
Project Responsible: Monica Landoni  monica.landoni@usi.ch
Project referee: Elisa Rubegni elisa.rubegni@usi.ch
Co-referee: Eynard Davide  davide.eynard@usi.ch
VoICN: Voice-over Information Centric Networking

*project supervisors*

Antonio Carzaniga and Michele Papalini

February 2012

Background

Information-Centric Networking is based on the idea that communication is centered on the content rather than on the location of that content. In other words, an information-centric network allows the user to address *what* content they want to access, as opposed to *where* that content might be in the network. The information-centric network architectures proposed so far, and specifically the Content-Centric Networking (CCN) system developed at the Palo Alto Research Center (PARC), allow users to retrieve data on-demand, using requests based on the name of the data as is typically done with Web content. However, a lot of traffic on the Internet is now based on a different form of communication called “publish/subscribe”. In this case, senders simply publish messages that the network then delivers to all the subscribers that are interested in those messages. Examples of application that use this kind of communication are Twitter, the Facebook notification system, RSS feeds and many others. Here at USI, we have been working on an extension of the CCN system that implements publish/subscribe communication natively.

Project Description

The goal of this project is to implement a voice and video conferencing application, such as Skype or Viber, that runs over our ICN architecture. This application will use all the communication facilities provided by the new network layer, including subscriptions and publications. We will start from VoCCN (Voice-over CCN), a similar application developed by PARC on CCN, as a starting point for the project. In fact, we would then like to compare the new conferencing application that also exploits publish/subscribe communication with VoCCN, which only uses on-demand communication.

Organization and Requirements

The project will be carried out in cooperation with and under the supervision of Antonio Carzaniga and Michele Papalini, and will last the full 8 week period of the UROP program. Candidate students must have an interest in networking. At a minimum, they must have successfully completed the Computer Networking class. Further, applicants should be comfortable with programming in Java but possibly also C. Most importantly, applicants must enjoy learning about and using new technologies, in particular new network architectures and protocols.
The Voronoi diagram is one of the fundamental geometric data-structure that encodes proximity information of a given set of geometric objects called sites (such as points, segments, curves, etc). Voronoi diagrams find application in many real life application areas, such as computer graphics, computer vision, geographic information systems, computer-aided design (CAD), computational science, etc. There are many variants of Voronoi diagram such as the nearest neighbor, the higher order, and the farthest neighbor.

In this project, we focus on the farthest line-segment hull, a recently defined structure that is related to the farthest line-segment Voronoi diagram, similarly to the way a convex-hull is related to the farthest point Voronoi diagram. We aim to adapt and implement different algorithmic paradigms for the construction of the convex-hull such as Graham scan, divide and conquer, gift wrapping and quick hull, to construct the farthest line segment hull. We would like to compare the performance of these algorithms both theoretically and experimentally.

Project Flow

Start

Study existing algorithms for construction of Convex-hull. Focus on Graham’s scan approach.

Understand definition of Farthest line-segment hull and how it is related to Farthest line-segment Voronoi diagram

Research on adapting Graham’s scan approach for the farthest line-segment hull

Implement the adapted Graham’s scan algorithm in your preferred programming language (C/C++/Java/Python/...)

Time permitting, perform comparisons with different algorithmic paradigms

Stop

Benefit

You will be able to acquire profound knowledge about fundamental geometric data-structures, convex-hull and Voronoi diagrams, which are useful in many real life applications.

Contact

Prof. Evanthia Papadopoulou, SI-209, evanthia.papadopoulou@usi.ch

Sandeep Kumar Dey, Open Space (2nd Floor), deys@usi.ch

References


1 Motivations and Goals

PreDiVine (Predicting Diseases of Vine) is a service developed in the Smart-Vineyard (SV) project\(^1\). The aim of SV is to study and develop a Wireless Sensor Network (WSN) for predicting vineyard diseases. PreDiVine is an innovative integrated solution to monitor, prevent and face vineyard pests and diseases, providing a new prevention and Decision Support System (DSS). It consists of: 1) a dedicated WSN, distributed in the vineyard for measuring the micro-climate conditions (temperature, humidity, leaf wetness, etc.); 2) a set of software implementing disease biological models for infection prediction.

Aim of the present proposal is to develop a Web Application (WA) for monitoring and management of PreDiVine services. In particular, the WA has to collect the sensors’ data and to visualize such data by means of interactive graphs. Alarms has to be generated based on pre-defined conditions.

Different sources for data acquisition have to be considered in the design of the WA. These sources correspond to different WSN installations geographically distributed as well as to other websites (e.g., weather forecast sites).

Access control mechanisms have to be put in place to manage the roles of Administrators and Customers (vineyard’s owners and phytosanitary offices). The former use the WA for checking the status of the WSN and for changing PreDiVine service parameters; the latter use the WA to visualize the PreDiVine data, receive the PreDiVine alarms in case of high disease probability, and report the presence of a disease to the Administrator.

Historical data and alarms have to be managed by the WA. Pesticide treatments has to be registered as well, for treatment validation.

Finally, the WA has to be developed by using open source technologies and it has to be designed with a Service Oriented Architecture (SOA).

2 Expected Results

In particular, a possible roadmap could be:

- Design of the database back-end to model the overall system;
- Evaluation of possible open source technology for Web Application design;
- Analysis of the technology for data acquisition (e.g., RESTful webservices);
- Design of Access Control features and site-views;

\(^1\)SMART VINEYARD is an ongoing Swiss Funded CTI/KTI project (CTI- 11307.1 PFES-ES).
• Implementation of the Web Application;
• Integration of data sources (e.g., sensors);
• Testing and refinement of the WA.

3 Student Details

For this project the following background is required:

• HTML/CSS;
• PHP and JAVA;
• Database (MYSQL);
• Linux Operating System.
LoSha2: Location Sharing Made Simple

UROP Project Proposal

As part of the Swiss National Science Foundation project "PALS ", we are designing a novel location sharing system that simplifies the act of sharing one’s location with others. Work so far as identified the underlying architecture, created an initial Android prototype app (“LoSha1”), and designed a set of interfaces for efficient control and inspection of location sharing. In a final step, we are now planning a large-scale rollout of an updated and streamlined application (“LoSha2”) with dozens of participants.

As part of the UROP project, a student will join the research team in order to implement and deploy a rock-solid Android application to dozens of study participants, and oversee the operation and overall data collection. The focus of the project will lie in advanced Android programming, mobile interaction design, and in supporting actual deployments.

In detail, the various tasks are

• Implement and possibly customize/extend a design for a location sharing application on Android, based on UI sketches and two prototypical implementations.
• Design and implement a robust data collection mechanism, consisting of both Android client code and Java server code.
• Design and conduct an actual study deployment, involving dozens of participants and ensuring the continuous data collection.

Applicants should have strong skills in Java programming; Android programming experience and HCI/UX design skills are a plus.

Professor: Marc Langheinrich
Supervisor: Marcello Paolo Scipioni
USI-Display: An application framework for networked public displays

UROP Project Proposal

As part of the international research project "PD-Net", we are deploying a number of large public screens on the USI campus. Contrary to existing public display systems, those screens will form part of an open network of displays that can be customized by both their owners and their viewers.

As part of the UROP project, a student will join an international research team from four European countries – Germany, Great Britain, Portugal, and Switzerland – in order to design, develop, and deploy a number of interactive applications both in Lugano and possibly at partner sites. The focus of the project will lie both in application and system design, as well as in participating in actual deployments.

In detail, the various tasks are

- Analyze requirements and design a reactive, multi-application framework around state-of-the-art Web technologies (HTML5)
- Interface with an existing scheduling component that will control the injection and termination of applications within the framework.
- Assemble, test, and deploy the system on a public display infrastructure in Lugano, and possibly at partner sites across Europe (extra travel funds available)
- Assist project members during public trials (data collection, system maintenance)

Applicants should have strong skills in Web design and HTML5/Javascript coding, and should be comfortable working in an international research team.

Professor: Marc Langheinrich
Supervisor: Ivan Elhart, Nemanja Memarovic