A graphical reporting tool for Moodle

Project description
The goal of this project is to implement a graphical and textual reporting tool for the Moodle Course Management System. The main purpose of this tool is to mine and discover the usage of specific Moodle courses. The tool has to provide a graphical interface that, at a glance, shows the usage of a specific module (for instance: discussions, quiz, content module, ... ) in every course managed by the Course Management System. The tool will allow comparing the usage in different courses, and seeing the distribution of the usage over time. Textual reports, in form of tabular data, that summarize statistics on courses’ usage has to be provided as well. The target users of this tool are the administrators of the Course Management System (who need to know how much the modules have been used in courses), the administration of the university, and the faculties (for organizational and educational purposes).

Organization
The project will be supervised by Riccardo Mazza and Fabio Crestani. The student will work in contact with people from the "Laboratorio per applicazioni di eLearning (eLab)”, http://www.cleavinglab.org. Since the Moodle platform is implemented in the PHP programming language, it is required the the student will implement the tool in such language and learns the details of the Moodle tool.

Contact
For more information, please contact Riccardo Mazza.
Covering, Independent Sets and Other NP-Complete Problems

Project supervisors: Laura Pozzi (laura.pozzi@unisi.ch)  
                          Paolo Bonzini (paolo.bonzini@lu.unisi.ch)  
Project duration: Two–three months

Research Background

The architecture of embedded systems (such as cell phones and set-top boxes) is different from that of general-purpose computers. Traditionally, embedded processors used application-specific hardware together with very simple and low-power processors. However, because of the increased power of these embedded systems (think of how your mobile phone is evolving!), it is becoming less and less cost-effective to develop custom hardware.

For the areas where performance is still not sufficient, microprocessor producers are exploring the area of customizable processors. These processors can include custom instructions that are arbitrarily complex and that have the purpose of speeding up specific applications, such as cryptography or audio/video decoding. Our goal is to develop effective ways of automatically extracting these custom instructions from C code.

Project Description

Many algorithms that deal with program analysis for the sake of processor customization work on graphs, and are in the class of NP-complete problems. This means basically two things: first, that they are hard to solve exactly; second, that many of these problems naturally transform into each other. The latter is often a breath of fresh air, because it is possible to find good solutions to a problem (though not necessarily the optimal one) by transforming the task to a different one.

The project’s goal is to concentrate on covering, one of the steps to automatically derive custom instructions, and analyze how to solve it. To this end, we’ll analyze both approximations to the covering problem (like branch-and-bound), and transformations of the task into different NP-complete problems (such as the maximum independent set problem).

Required Skills

The student must have interest in algorithms, and have completed both the Discrete Structures and Algorithms And Data Structures courses.

The student should have a good knowledge of C, and a willingness to become a serious, disciplined C hacker.

How to Apply

Please contact Paolo Bonzini.
Context
SpyWare is a research prototype we have implemented. SpyWare monitors programmer activity to allow a fine-grained analysis of their programs and their behaviors. SpyWare records all the actions the programmer performs, and converts it to change operations. Examples of change operations are:

- On 03/04/06, at 14:42, the programmer (let’s call him Bob) added the class Bar to the system.
- On 03/04/06, at 14:45, Bob added the method ‘baz’ to class Bar.
- On 03/04/06, at 14:50, Bob changed the method ‘baz’ in class Bar.
- On 03/04/06, at 15:02, Bob extracted two line of code from ‘baz’, to the new method ‘foo’, in class Bar.

SpyWare provide tools to exploit, export and visualize this information, such as graphs, time-aware code browsers, and a change matrix, as shown in this screen capture:

![SpyWare Visualization](image)

Description
SpyWare provides graphical visualizations of its data, but they are static. To visualize the change information in an efficient manner, animations are needed. Support for animations and incremental layout of graphical entities would allow SpyWare to display the history of a system as a “movie”, rather than a “photo album”.

Here is a tentative roadmap of the tasks you will have to do during the project:

- Investigate the Mondrian graphical framework, which can be used to implement static visualizations of data.
- Read articles about algorithms laying out graphs incrementally. Such algorithms are necessary to consider objects appearing after a visualization is displayed.
- Extend Mondrian to support incremental addition of figures to a visualization.
- Extend Mondrian to support animations.
- Implement a lot of cool visualizations!
Figure 1: ArgoUML in CodeCity[1, 2].

α : OpenCroquet
β : X3D
γ : DynaMoose
δ : Software Tycoon

References


Context

When developers are faced with a new system that they have to understand, a visual representation of the system is an important aid in comprehension. Many software visualization tools exist that support program comprehension (e.g., SHriMP, Rigi, CodeCrawler, PBS, Softwarenaut, etc.) but most of them are stand-alone tools. Recently, X-Ray, a simplified version of Softwarenaut has been integrated in the Eclipse development framework. However, there are many features which still have to be added to the tool.

Description

Some of the tasks that have to be done in relation to X-Ray are:

- A user study which would show which are the preferred tool usage patterns. This is important to get the pulse of what is really important for the user. Probably new implementation requirements will follow from this.

- Provide a mechanism for view customization and persistence. The user should be able to define multiple perspectives on the system.

- Implement other features that the user study suggests, or the student considers important in the limit of the available time.

Milestones

The length of the project will be two months. There will be weekly meetings between the advisors and the student.
Clustered In-Memory Databases

Project Description

In-memory databases (IMDBs) provide high throughput and low response time by avoiding disk I/O. IMDBs have traditionally been used by specific classes of applications (e.g., telecommunication). Current performance requirements and cheaper semiconductor memories, however, have pushed them to more general contexts (e.g., web servers, trading systems, content caches). In most cases, applications are limited by the memory capacity of the server running the IMDB. In a clustered environment, the database could be partitioned and replicated into segments and stored in the main memory of several nodes. Applications would be limited by the aggregated memory capacity of the cluster, and not by the capacity of a single node.

The goal of this ambitious project is to extend FastDB, and IMDB, to a cluster of servers.

Distributed Orchestration of Composite Web Services

Advisor: Walter Binder

Service-oriented computing, a new approach to software development, enables the construction of distributed applications by integrating services that are available over the web. The building blocks of such applications are web services that are accessed using standard protocols. The composition of individual web services into an added-value, composite web service is usually represented as a workflow.

Even though a workflow may invoke services distributed over multiple servers, the orchestration of the workflow is typically centralized. E.g., BPWS4J acts as centralized coordinator for all interactions among the individual services within a workflow. While this approach gives complete control over the workflow orchestration to a single entity (which may monitor the progress), it often leads to inefficient communication, as all intermediary results are transmitted to the central workflow orchestration site, which may become a bottleneck. This is particularly problematic, if a workflow is executed on a mobile device with limited or expensive network connection.

This UROP project will investigates a new model of fully decentralized workflow orchestration. We will introduce proxies for individual service invocations that collect the required input data before they invoke the service. Moreover, they forward service outputs to exactly those sites where they are actually needed, supporting multicast. Therefore, inefficient routing of intermediary results through a centralized coordinator can be avoided.

In this UROP project, the student will be involved in the following tasks:

1. Specification of the proxy functionality and of the necessary communication protocols.
2. Development of an algorithm to translate workflows so as to leverage the proxy infrastructure.
3. Evaluation of savings in network bandwidth compared with centralized orchestration in simulation.
4. Prototype implementation in Java.

In tasks 1 and 2, we will consider typical workflow patterns and find an optimal tradeoff between completeness regarding the supported workflow patterns and simplicity of the proxy infrastructure. Furthermore, we consider optimizations for the frequent case where the same workflow is executed multiple times.
Distributed Simulation Environment

**Project supervisors:** Cyrus Hall, Jeff Rose, and Antonio Carzaniga  
**Project duration:** Two months

**Project Background**

Many types of research require massive amounts of simulation. Often simulations grow too large to run in a reasonable amount of time on a single machine. While running multiple simulations simultaneously across many machines can offer large speedups, the parameter space exploration process would be made much easier if individual simulations were faster. Further, some simulation are just too large to fit into main memory on a single machine, and therefore need to be distributed.

**Project Description**

The student will help with design and implementation of a distributed simulation environment. In particular, we will be trying to identify good strategies for simulation object placement across many machines, in order that communication between machines is minimized. Other integrated efforts will include organizing data collection and workload distribution. There is lots of room for experimentation in this project.

**Project Organization**

The project will be under the supervision of Antonio Carzaniga, Cyrus Hall, and Jeff Rose. The project should last about 8 weeks, with set two week goals to measure progress. The student must have an avid interest in networking and modeling, have completed the Computer Networking class and have a working knowledge of Ruby (with a willingness to become a serious Ruby hacker).
Dynamic On-the-fly Code Generation for Efficient Trace Analysis & Visualization  
Mentor: Matthias Hauswirth

Today’s computing platforms (hardware, operating systems, virtual machines, middleware) consist of highly dynamic components. Engineers and researchers study the dynamic behavior of these platforms to find opportunities for new performance improvements. For this purpose they often instrument the platforms to generate a trace (a log) of relevant events. When capturing the behavior of commonly used benchmark applications, the size of such traces can vary from a few dozen events (e.g. one event for each garbage collection in a virtual machine), to thousands of events (e.g. one event per time slice of a multithreaded application), to millions of events (e.g. one event for each virtual method call in an object-oriented application), to several billion events (e.g. one event for each completed instruction on a modern CPU, traced for one second).

Due to the large number of events, trace data often is aggregated before it is written to disk. For example, instead of a sequence of individual instruction completion events, the trace file may contain a sequence of values (a time series), where each value corresponds to the number of events observed over a fixed time interval (e.g. 10 ms). Traces often contain a mix of different types of individual events in addition to aggregate time series.

Due to this heterogeneity and size of the traces, classical tools for data analysis and visualization often break down. Researchers thus often produce special purpose tools for each new type of trace they need to analyze. At USI and at the IBM T. J. Watson Research Center we are developing a flexible trace analysis and visualization infrastructure that can efficiently handle large traces. Moreover, the infrastructure can even process traces online, as they are received from the instrumented application.

In this UROP project you will develop and evaluate various design alternatives of a crucial component for this infrastructure. Your component will compile trace format specifications (definitions of all event types and time series) into Java classes for the efficient processing and in-memory storage of corresponding traces. The component will generate the bytecode for these classes and load them on-the-fly, so a running trace processing application can read the trace specification and immediately start processing that trace. You can use a parser generator (such as ANTLR or JavaCC) for producing a parser of the event specifications, and a library like the BCEL (Apache Byte Code Engineering Library) to generate the class files.

**Required Skills**
UROP internship project proposal

**Sponsors**: Paolo Giordano, Alberto Vancheri, Denise Andrei (MACS-Lab, USI)
**Mentor**: Giovanni Ciampaglia

**Georeferencing with heterogeneous Web APIs**

**Project description**

The intern student may select one or more tracks for his project among the following; note that some tracks are dependent on some other tracks:
- Extraction of a graph representing road connections out of Google Maps or Swissgeo imagery. The graph must be detailed with street names and information about one way streets. Each street is described as a polyline. The graph will be stored as a shapefile. The shapefile (shp) format is an industry standard for GIS applications, developed by ESRI.
- Construction of an approximate graph out from the previous one. In the detailed graph a link is a part of a street that connects two consecutive junctions between different streets. The union of consecutive links that follow a straight path can be aggregated in a single component. The goal is to provide a faithful description of a road map, yet as lightweight as possible.
- Extraction and georeferencing of commercial activities stored in Google Maps and other mapping services on the Web.
- Design and implementation of an algorithm for the selection of routes for agents with an agenda-based planning process.

**Internship description**

The intern student will work in close connection to the researchers of MACS-Lab to design and implement solutions for the above tracks. Part of the student work will be to select and integrate existing sets of API to access and manipulate data from several heterogeneous sources (e.g. Javascript API to access Google Maps imagery, MATLAB API etc.).

This is a research oriented project; internships characterized by a high degree of autonomy and that provide brilliant results are open to the possibility of publication on a scientific journal in the field of geographical computing.
Informa Clicker
Design, Implementation, and Evaluation of Solution Similarity Measures
Mentor: Matthias Hauswirth

Informa is a classroom response system developed at USI. In the classroom, students run the Informa Clicker Java application on their notebooks, while the instructor runs the Informa Instructor application. The instructor publishes a question or problem to be solved by students in their clicker application. Students solve the problem, and anonymously submit the solution to the instructor. Besides presenting each individual submitted solution, the instructor application also provides aggregate information that helps the instructor and the students identify problems of understanding.

The research aspect in the Informa project is the automatic summarization of the submitted solutions. For multiple choice problems (the simplest type of problem implemented in Informa) this summarization is trivial: a histogram presents the number of students who picked each available choice. However, for less constrained problems, for example where a solution consists of an arbitrary list of words, a snippet of Java code, a UML diagram, or a digital logic circuit, summarization becomes a challenge. Are two digital logic circuits (two graphs) the same? How similar are they? Did the students who created the two circuits have the same issue in understanding the problem? A system that can automatically answer these questions can eliminate similar solutions and present a summary that consists of one solution for each issue the students had.

Visualization of the hierarchical agglomerative clustering of solutions of the regular expression problem “select the first substring matching the regular expression ab+”

In this research project you will focus on one type of problem (e.g. the digital logic circuit, or the list of words, ...), and provide the means to effectively summarize a large number of solutions in a visualization that fits on a single screen. One approach to such summarization is the clustering of solutions based on their similarity. The summary consists of the presentation of representative solutions from a small number of clusters of similar solutions. Clustering algorithms require a means to compute the similarity of two solutions (a similarity measure).

You will design and implement a set of such similarity measures for the chosen type of problem. You will then use these similarity measures to cluster a set of solutions, and you will evaluate the usefulness of the resulting clusters. The result of your project should be a plug-in for Informa that provides the most useful automatic summarization of the chosen problem type.

Required Skills
Monitoring conversational web services

Project Mentor  Domenico Bianculli (domenico.bianculli@lu.unisi.ch)
Project Sponsor  prof. Carlo Ghezzi (carlo.ghezzi@polimi.it)

Project Background
Web services support a dynamic architectural style where the binding among components may change at run-time. New services may be developed and published in registries, and then discovered by possible clients. Previously available services may disappear or become unavailable. This situation has been characterized by the term open-world software, to describe a situation where applications are composed out of parts that may change unpredicatably and dynamically. It has been observed that open-world software introduces the requirement of continuous validation. Since a software architecture evolves dynamically, validation must extend from development time to run-time.

Monitoring is a necessary component of a run-time validation facility. Monitoring consists of collecting data about a monitored service and checking that the data conform to the expected behavior of that service.

Project Description
In this project, we will focus on conversational web services, i.e. stateful services whose expected behavior depends on the history of previous interactions between a client and a given service instance. We assume the expected behavior to be described as a formal, algebraic specification of the monitored service.

The goal of this project is to extend an open source BPEL engine, ActiveBPEL, to support monitoring of conversational web services, by integrating a symbolic evaluator of algebraic specifications. The monitoring architecture will be developed using aspect-oriented programming techniques.

Required skills
Good programming skills in Java are required. The student should also be interested in learning web services technologies, as well as AspectJ (aspect-oriented programming language) and CafeOBJ (algebraic specification language).

Contact
For more information, contact Domenico Bianculli.
Model Annotation, Constraint and Code Generation
for Self-Checking Programs

Project mentor/sponsor: Jochen Wuttke, Mauro Pezzè

Models are commonly used to describe the structure and behavior of software on a higher, more abstract level than the actual source code. Unfortunately, current modeling techniques for system behavior have only limited possibilities to express in detail what is the correct behavior. To make models more complete with respect to the requirements and the actual implementation, a mechanism to specify constraints on the allowed behavior is required. In addition, if these high-level constraints can then be automatically translated into code, which checks the program while it is running, we also have a means to check if our program actually behaves like we specified.

This project has the goal to use annotations in software models (e.g. UML or EMF class models) to automatically derive constraints in the form of pre- and post-conditions, translate them into source-level checks, and integrate them into existing applications to control their runtime behavior.

Details

The final outcome of this project should be an Eclipse plug-in that transforms EMF and/or UML2 models and annotations into Java aspects, which check the constraints on the program. To do this you need to familiarize yourself with the EMF and UML2 plug-in frameworks to assess which one is better suited for the task. With the chosen framework, you then have to create a model of the Java Pet Store web application as the running example throughout this project. The code generator based on the model should generate Java aspects and possibly a comprehensive OCL specification of the constraints. To know which aspect oriented framework to use, you would also have to assess the available frameworks, and chose one that is suitable for the project. Once you are able to generate code, you will then have to assess the effectiveness of the generated checks, and possibly propose improvements on the constraints.

The key research questions you will need to tackle are how to find the right places to add checking code just by looking at the model. And once you know how to do that, what kind of checks are actually useful, how much information must the model contain, so that you are able to create the checks, etc.

Key Outcomes

- You learn how to generate code from software models.
- You learn how to use aspect oriented programming to deal with cross-cutting concerns in programs.
- You learn how to use (and how not to use) specification techniques like pre/post-conditions to improve software quality and reliability.
- You learn how to write an Eclipse plug-in.
- You get to write a nice project report ;-)
Service Composition Algorithms Operating on a Compact Directory Digest

Advisor: Walter Binder

Service-oriented computing enables the construction of distributed applications by integrating services that are available over the web. The building blocks of such applications are web services. Service discovery is the process of locating providers advertising services that can satisfy a given service request. Automated service composition addresses the problem of assembling individual services based on their functional specifications in order to create a value-added, composite service that fulfills a service request. Most approaches to automated service composition are based on AI planning techniques and assume that all relevant service advertisements are initially loaded into a reasoning engine.

However, due to the large number of service advertisements and to the loose coupling between service providers and consumers, services are indexed in service directories. As loading a large directory of service advertisements into a reasoning engine is not practical, we investigate a compact encoding of service directories as Zero-Suppressed Reduced Ordered Binary Decision Diagrams (ZDDs) in this UROP project. In such an encoding, only those aspects of service specifications are preserved that are relevant for automated service composition. The ZDD representing a large service directory shall be small enough to make frequent distribution to service composition engines practical.

In this UROP project, the student will be involved in the following tasks:

1. Design of service composition algorithms operating on ZDDs.

2. Implementation of these algorithms in Java.

3. Creation of a testbed for large-scale, automated service composition.

4. Evaluation of the implemented service composition algorithms.

5. Improvement of a ZDD library implemented in Java (improvement of the programming model as well as performance optimizations).
Simulating MAC-Level Protocols for Sensor Networks

Project Description
The goal of this project is to implement a small suite of established, well-known MAC-Level protocols for wireless sensor networks. The protocols should be implemented in C++ for the network simulator Omnet++. The project will give you the opportunity to learn quite in depth how sensor networks work and what are their goals and challenges. The ready suite will be published as an official part of the Omnet++ simulator and probably be used widely from researchers in that area.

Project Organization
The project will be under the supervision of Anna Egorova-Förster and Amy Murphy. Together with them you will learn how to use the Omnet++ simulator and how to program for it, and discuss the protocols to be implemented. Some basic networking knowledge (as from the Computer Networking course) and very good programming skills in Java or C++ are required.

Contact
For more information, please contact Anna Egorova-Förster.
Simulating Routing Protocols for Sensor Networks

Project Description
The goal of this project is to implement a small suite of established, well-known routing protocols for wireless sensor networks. The protocols should be implemented in C++ for the network simulator Omnet++. The project will give you the opportunity to learn quite in depth how sensor networks work and what are their goals and challenges. The ready suite will be published as an official part of the Omnet++ simulator and probably be used widely from researchers in that area.

Project Organization
The project will be under the supervision of Anna Egorova-Förster and Amy Murphy. Together with them you will learn how to use the Omnet++ simulator and how to program for it, and discuss the protocols to be implemented. Some basic networking knowledge (as from the Computer Networking course) and very good programming skills in Java or C++ are required.

Contact
For more information, please contact Anna Egorova-Förster.
UNDERGRADUATE RESEARCH OPPORTUNITIES PROGRAM

Project: Verification Conditions with Strings
Mentor: Stefano Tonetta
Sponsor: Natasha Sharygina

University of Lugano
Faculty of Informatics
Formal verification provides tools either to give a mathematical proof that some property holds in a given program or to detect its bugs and vulnerabilities. Typical security vulnerabilities such as buffer overflow and format string are caused by user-provided strings. These strings are usually stored in dynamic buffers whose sizes are unknown a priori.

In this project, we address the problem of expressing and verifying program conditions that involve strings. We investigate the use of a formalism recently developed for the analysis of programs with strings. We want to demonstrate the applicability and advantages of our approach on a real-life case study taken from the Apache HTTP server.
Project Goals

The final goal of the project is to implement a tool that takes in input a program written in (a ultra-simplified version of) C with some verification conditions, and checks if the program satisfies the conditions.

In order to simplify the task of the implementation, we will consider a core subset of C that involves just constants, strings and string manipulations.

A case study taken from the Apache HTTP server will be used to provide a test-bench to validate the tool.

The tool will parse the program, encode its execution into a formula, and check if the verification conditions are satisfied. The last step will be accomplished by interfacing with a particular decision procedure.
Project Organization

The project activity will be organized according to the following schedule:

- definition of case study: 1 week
  - identify interesting functions
  - insert verification conditions
  - simplify the code
- tool design: 1 week
- tool implementation: 4 weeks
  - lexical analyzer
  - parser
  - program simulation
  - interface with decision procedure
- tool validation: 1 week
- report: 1 week
Proposal for a Summer Internship: Web Service for a Metadata Catalog

Several Grid projects that CSCS is part of are in need of a metadata service for small and large datasets. The community has already invested a lot into providing such services, however the problem always was that these services were too specific, had proprietary interfaces or were abandoned after the project has stopped.

The internship would be about reviving and modernizing an existing service, filling out the gaps and adding features. The existing metadata catalog comes from the EGEE project (Enabling Grids for e-Science in Europe II), which CSCS is member of.

The internship project would need to
- Start from the existing middleware
- Brush up the Web Service Definition Language
- Apply newer service technologies to it (later versions of Axis, Tomcat, etc)
- Provide a catalog instance for one of CSCS’s projects (with specialized client-side APIs).
- If time allows: provide a web interface through AJAX.

The programming environment can be freely chosen by the student, we would prefer to use IntelliJ Idea, for which we have a project-license; a commercial development environment, which is similar to (but in many ways better than) Eclipse.

The contact person at CSCS is Peter Kunszt, details see below.
Hugo Boss Shoe Design

**Proof of concept of a two-handed 3D manipulation environment**
We are looking for motivated students for an internship at the Innovation Department working on a project in the area of virtual shoe design.

Shoe design is a task still performed with mock-ups, pen and paper, at best with graphical applications like for example Illustrator. This makes the design process inflexible and relatively time consuming.

One of the main obstacles to designing in a 3D virtual environment seems to be the hardware interface; single-handed interfaces based on conventional devices such as a mouse and/or track ball are poor substitutes for the freedom experienced when working with pencil and paper.

The object of this project is to create a ‘demonstrator’; showing the potential of having an environment that supports two handed input. You will develop a simple 3D design environment supporting the simplest of use cases (for example drawing a line on a 3D object) for two-handed interaction using two out of the box input devices.

More information
Fred Voorhorst, tel: 091 696 6642
DESCRIZIONE PROGETTO
Il progetto prevede inizialmente la verifica dei processi interni. Tale analisi permetterà, da un lato, di verificare se siano possibili delle ottimizzazioni che possano garantire un incremento di efficienza e, dall’altro, garantirà che il futuro sistema informativo sia coerente con l’operatività aziendale. Dopo tale fase di analisi, inizierà quella di implementazione articolata in diverse attività, sia tecniche che applicative. L’implementazione si protrarrà per diversi mesi, permettendo gradualmente all’impresa di familiarizzare con il nuovo sistema e verificarne la rispondente alle necessità dell’azienda stessa.

SOLUZIONE ADOTTATA
Microsoft Dynamics NAV

REQUISITI
- Buona capacità di analisi delle procedure aziendali
- Buona visione dell’intera operatività aziendale
- Capacità di relazione e di formalizzazione della documentazione di progetto

COMPETENZE ACQUISTABILI
A fine progetto lo studente avrà acquisito maggiori competenze analitiche nelle diverse aree aziendali, che gli permetteranno di poter affrontare in maniera confidente le sfide proposte dal mondo professionale.

In un progetto di tale portata inoltre si apprende, da un lato, a rispettare tempi e costi di progetto e, dall’altro, a lavorare sia in autonomia che in team.

Infine, grazie a questo progetto, lo studente apprenderà a rapportarsi con i propri clienti (in questo caso i futuri utenti del sistema) in maniera professionale.

PERSONA DI RIFERIMENTO
alessandro.parravicini@tc-systems.ch

Massagno, maggio 2007