iGame

Game Authoring Tool for iPhone / iPod touch

Navid Ahmadi

Problem: Handheld devices, e.g., iPhone, iPod Touch, and Nokia N800 series are the next collaboration platform for end-users to run games and scientific simulations. Such end-user developed applications need to run efficiently on handheld devices. However, interaction and resources limitations of such platforms require specific support for running end-user developed applications.

Objective: Ristretto[Mobile] is a Web-compliant framework for end-user development on handheld devices. Ristretto Mobile has a Web-based architecture, which not only enables seamless integration of it into other Web 2.0 applications, e.g., Social Networks, but also addresses the lack of unified programming platform for handheld devices. Figure 1 depicts an iPhone running a game called Sokoban in the iPhone's Safari using Ristretto[Mobile]. Sokoban was developed using AgentSheets, an end-user development environment, which enables end-users to develop their own games as well as simulations based on a graphical rule-based programming language. The ultimate vision is to build an infrastructure that enables end-users to collaborate through complex applications such as games and simulations. For more information, please take a look at Ristretto[Mobile] Web page[1].

Approach: We have started implementing Ristretto[Mobile] as an engine to run end-user created interactive content such as simulations and games in a Web browser. It is especially customized for running in the Safari browser on iPhone and iPod Touch, and also in Firefox on Nokia N800 Internet tablets. Till now, the fundamental components and a subset of instructions have been implemented and we got some preliminary results.
There is still more work to complete the Ristretto[Mobile], suitable as a bachelor project. Initially, you will find out how a 2D game engine and a compiler, both written in JavaScript, work. Then you will extend both the game engine and compiler to support enough subset of instructions. The game engine also needs some speed and graphical optimizations to run efficiently on handhelds.

Figure 1. Ristretto[Mobile]: An iPhone running an end-user developed game

At the second step, a small but reasonable amount of the authoring environment has to be implemented, which can be a combination of both server side and client side programming. The goal is to enable end-users not to run their games on the Web, but also create their games on the Web.
Finally, you will need to do some back-end programming to create a collaborative, Web 2.0 type of Web site where end-users share their developed games with others.

Learning opportunities: Besides becoming familiar with research areas such as *Web 2.0*, *end-user development*, *game design*, *human-computer interface*, and *technology-enhanced learning*, you will be trained in both client side and server side of writing Web-based applications:

- **Client side Web programming:** To extend the game engine and compiler, you will learn JavaScript to the ultimate extent, including object-oriented programming in JavaScript. To optimize the game engine, you will learn how to debug and profile JavaScript programs. To customize it for iPhone and Nokia N800, you will learn how to develop specialized applications for apple's handheld devices. To communicate over the Internet, you will learn Ajax.
- **Server side Web programming:** To create the authoring tool, again you need to learn JavaScript very well, together with a server side Web development language such as Ruby on Rails or Google Web Toolkit, or any of your choice. Server side Web development will also be used to create the collaborative Web site for end-users to share their games.

Bio: **Navid Ahmadi** received his B.Sc. in Computer Science from Mathematics and Computer Science faculty, University of Tehran, and his M.Sc. in Computer Engineering from Tarbiat Modarres University, Tehran, Iran, in 2004 and 2006 respectively. He is an Informatics Ph.D. at University of Lugano since November 2006, under supervision of professor Mehdi Jazayeri. His main research interests include World Wide Web and related topics such as the Semantic Web, Web 2.0, Service-Oriented Computing, Computer Supported Cooperative Work, End-User Development, and their application in social and pedagogical activities.

[1] **http://www.inf.unisi.ch/phd/ahmadi/RistrettoMobile/index.html**

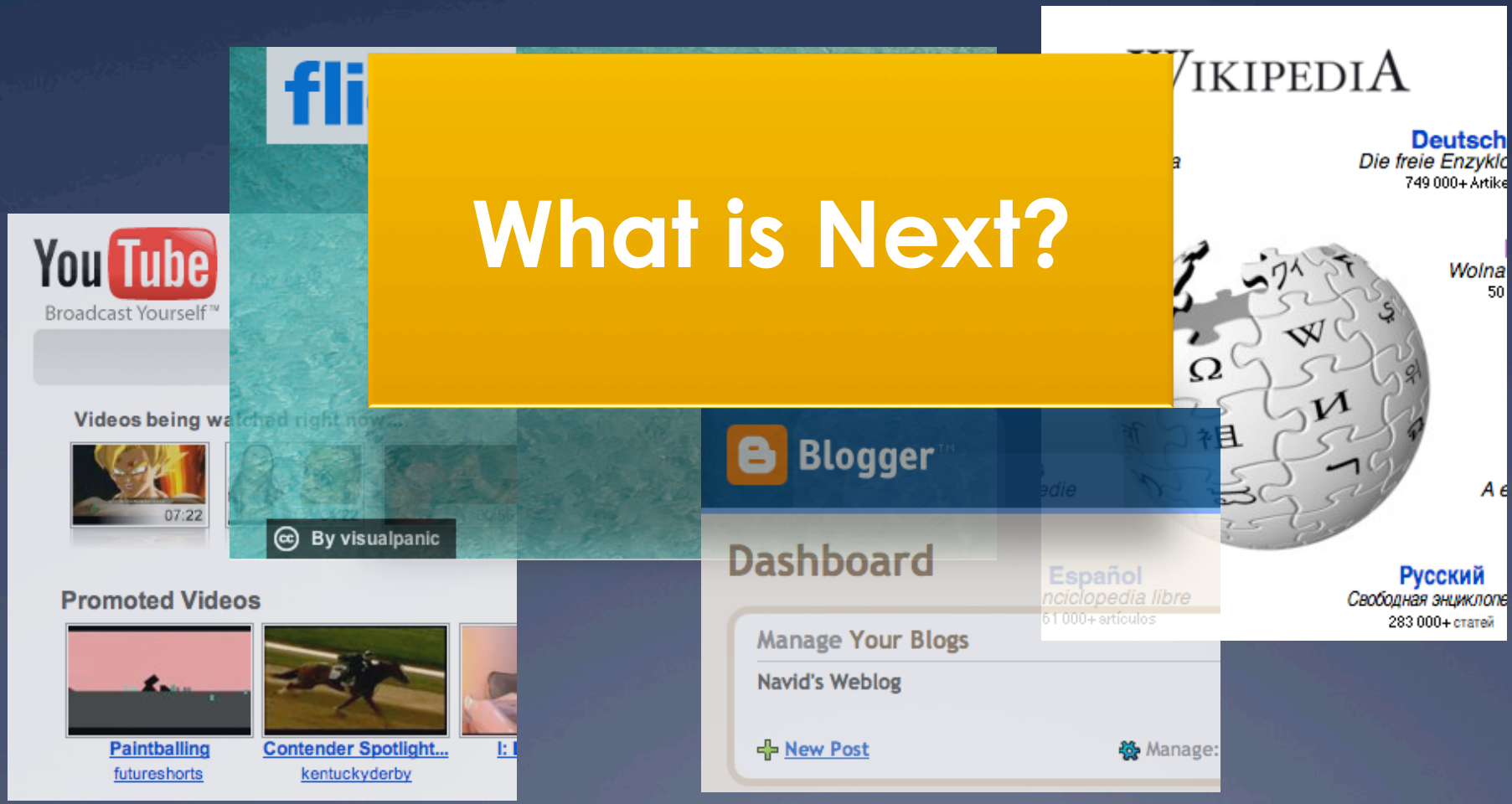
iGame

Game Authoring Tool for Handheld Devices

Proposal for UROP'08
Navid Ahmadi

USI-Informatics, May 20, 2008

Web 2.0
User Generated Content

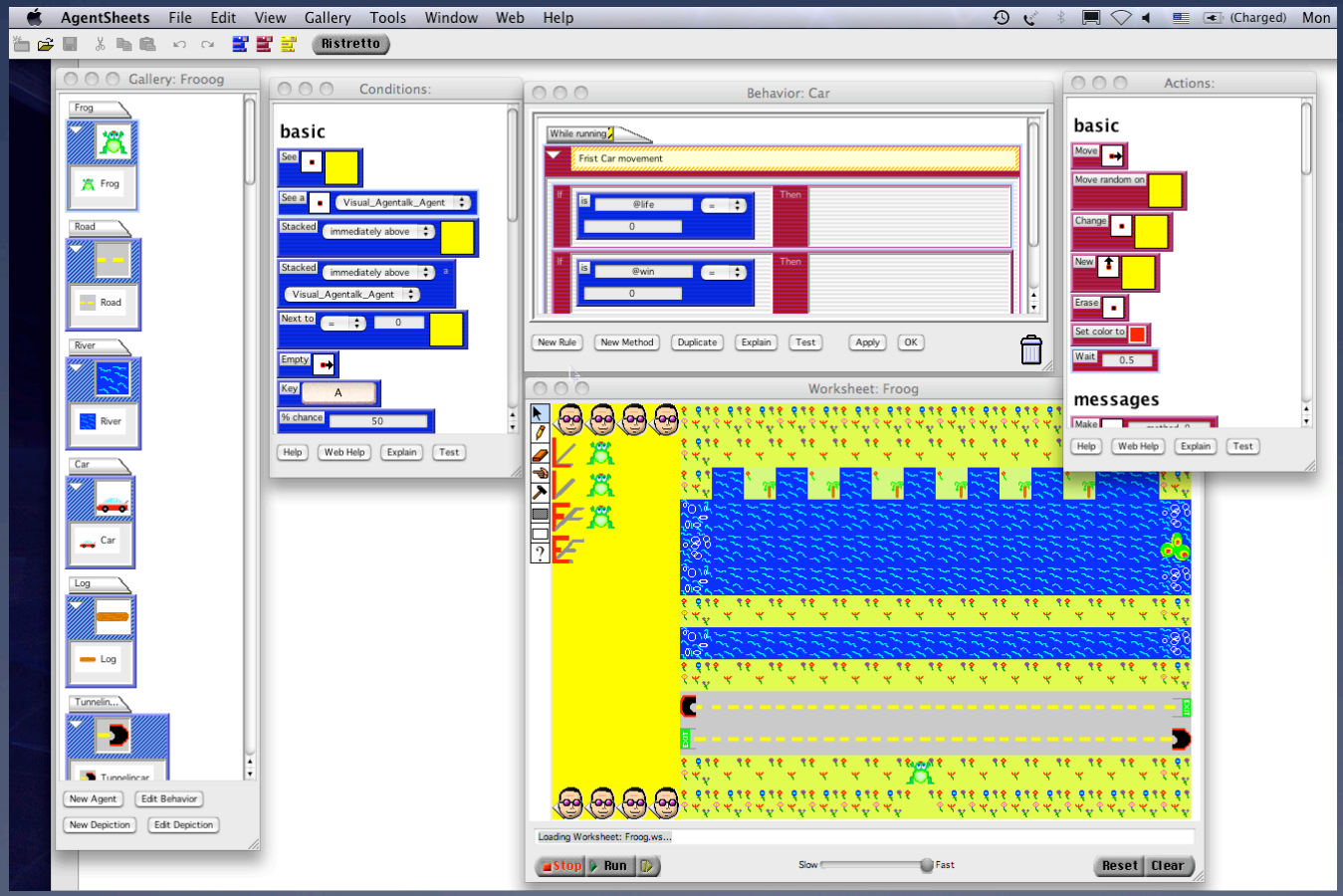


Make Your Own Casual Game on Handheld Devices

By 2010: **80 Million** Casual Gamers, **$2.1 Billion** Market

* RistrettoMobile: Game Development for Handheld Devices
  * iPhone, iPod Touch, Nokia N800 Series
* http://www.inf.unisi.ch/phd/ahmadi/RistrettoMobile/index.html

AgentSheets Authoring Tool



Developing the iGame

- Client-side development
  - AJAX-based Web application that simulates AgentSheets development environment
  - Customized for running on handheld devices
- Server-side development
  - Provides collaborative functionalities such as sharing and editing of games

Interested in?

- Stop by my office and ask your questions
  - First Floor, room SI-104
- Or, send an e-mail: ahmadin@lu.unisi.ch

Università
della
Svizzera
italiana

**Facoltà
di scienze
informatiche**

A coarse grained reconfigurable architecture test bed

Project supervisor: Laura Pozzi (laura.pozzi@unisi.ch)
Giovanni Ansaloni (giovanni.ansaloni@lu.unisi.ch)
Project duration: Two-three months

Research background

The reconfigurable computing field has gained increasing interest in recent years, due to its promise to achieve high performance, low area and low power consumption. Unfortunately, the vast majority of commercially available reconfigurable devices (FPGAs and CPLDs) focus on fine-grained flexibility, thus sacrificing performance. Coarse-grained reconfigurable architectures instead strike a balance between the degree of programmability and the desired performance; this approach has shown promising results in research, especially in the embedded application field, were computation is mostly centered on the execution of data-intensive loops (kernels).

Project description

Recent work by Laura Pozzi, Paolo Bonzini and Giovanni Ansaloni has introduced a coarse-grained architectural template, aimed at accelerating embedded applications. The project goal is to integrate this template as a custom instruction on a System On a Programmable Chip (SOPC) to be downloaded on a FPGA, for testing purposes.
The student will design the required accelerator-microprocessor hardware interface and code a software test application that takes advantage of the accelerator.
During the project, the student will have a deep insight on reconfigurable architectures and commercially available tools aiding designers in SOPC realization.
The work will build on previous work on custom instructions definition.

Required skills

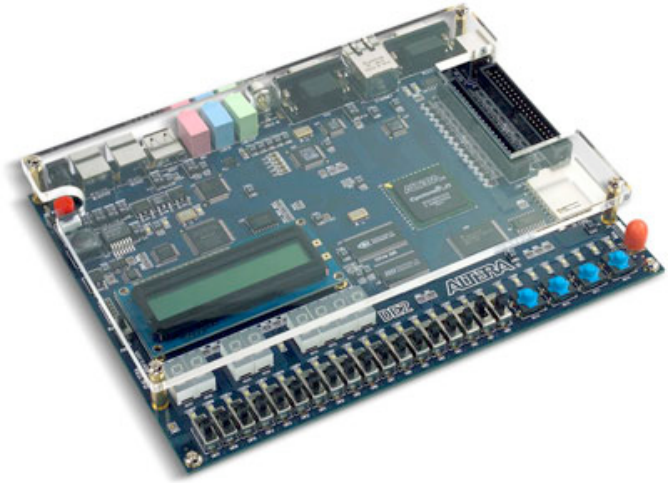
The student must have interest in computer architectures and low-level software.
Knowledge of a hardware description language (VHDL, Verilog) is recommended.
Patience, commitment and curiosity about HW-SW interactions are also a plus.

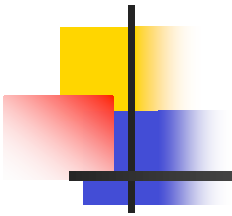
How to apply

Please contact Giovanni Ansaloni

A coarse grained reconfigurable architecture test bed

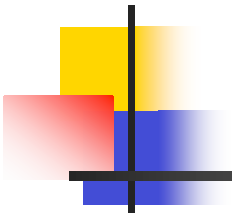
- Advisors:
  - Laura Pozzi
  - Giovanni Ansaloni
- Duration:
  - two months





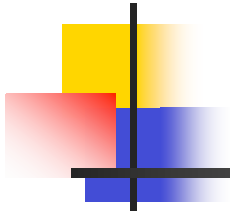
Resarch background

- FPGAs are hardware devices that can flexibly morph their behaviour to serve multiple applications
- We are researching a novel kind of FPGA, called Expression-Grained Reconfigurable Array (EGRA) that outperforms FPGAs in some cases
- EGRA present itself as an accelerator for embedded system applications, and was designed in previous months
- A demonstrator of EGRA capabilities is needed, in order to:
  - test the design
  - quickly evaluate additional features



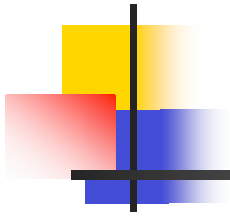
Project goals

- The demonstrator will be realized on a “classic” FPGA, comprising a microprocessor and the EGRA accelerator
- The project deals both with HW and SW
  - HW: integration of an EGRA into a System-on-a-programmable-chip (SOPC)
    - EGRA already designed - interface to be designed
    - SOPC to be designed (high- level, graphical interface)
  - SW: coding an example application that takes advantage of the accelerator
    - application written in C
    - to be specialized for the accelerator



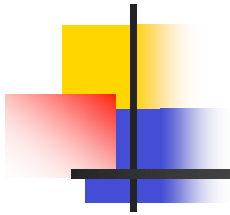
What you will learn

- Hw/Sw interactions
- How to use commercially available tools to realize complex HW systems on reprogrammable devices



Required skills

- Interest in computer architectures
- Knowledge of VHDL or Verilog is a plus
- Commitment, curiosity
  ... and patiance (a lot!!)



To apply contact: Giovanni Ansaloni

ansalong@lu.unisi.ch

UROP Project – Summer 2008

Monica Frisoni, Prof. Cesare Pautasso

Composable Web 2.0 Widgets for Real-time Business Process Monitoring

Background

JOpera (www.jopera.org) is a rapid composition tool to build distributed applications out of reusable services of any kind (including Web services).The main abstraction of *JOpera* is the Process: modeling the flow of control and data between the various services.

The *Monitoring Widget* is a simple, standalone Web application that allows the monitoring of the processes running on a remote server from a common Web browser. It may be easily integrated and executed within any HTML Web page by embedding a snipped of code within the user page.

This project is offered in collaboration with the Swiss National Supercomputing Center (http://www.cscs.ch) where the results of the project will be deployed.

Objective

The student will contribute to the optimization and extension of the existing Monitoring Widget by accomplishing some or all of the following tasks:

1. Collect multiple requests coming from more widgets before sending them to the Web server.
2. Display more accurate information about the monitored process using advanced visualization.
3. Optimize the interaction and the customization/configuration of the end user with the widget.

Technologies

You will learn a lot about Java, JavaScript, HTML, CSS, and AJAX.

Contact

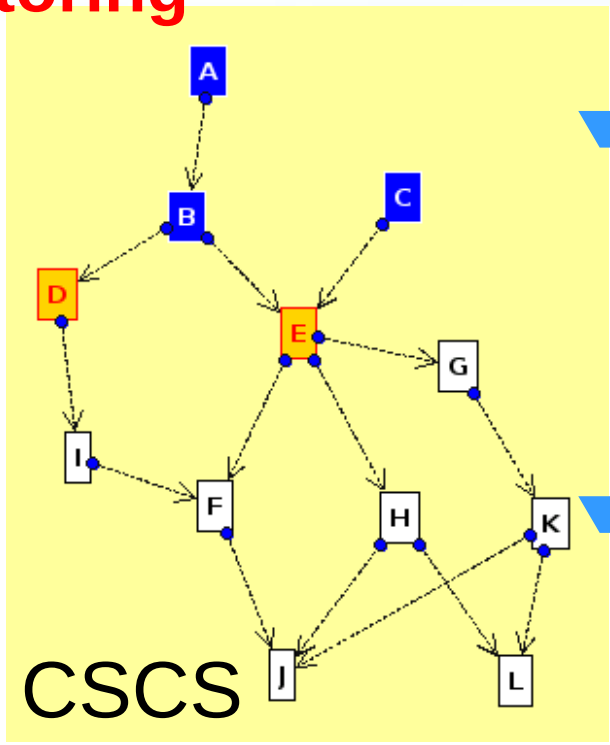
For more information, please contact Monica Frisoni <monica.frisoni@lu.unisi.ch>

Composable Web 2.0 Widgets for Real-time Business Process Monitoring

Monica Frisoni - Prof. Cesare Pautasso
monica.frisoni@lu.unisi.ch
http://www.jopera.org

UROP Undergraduate Research Opportunities Program

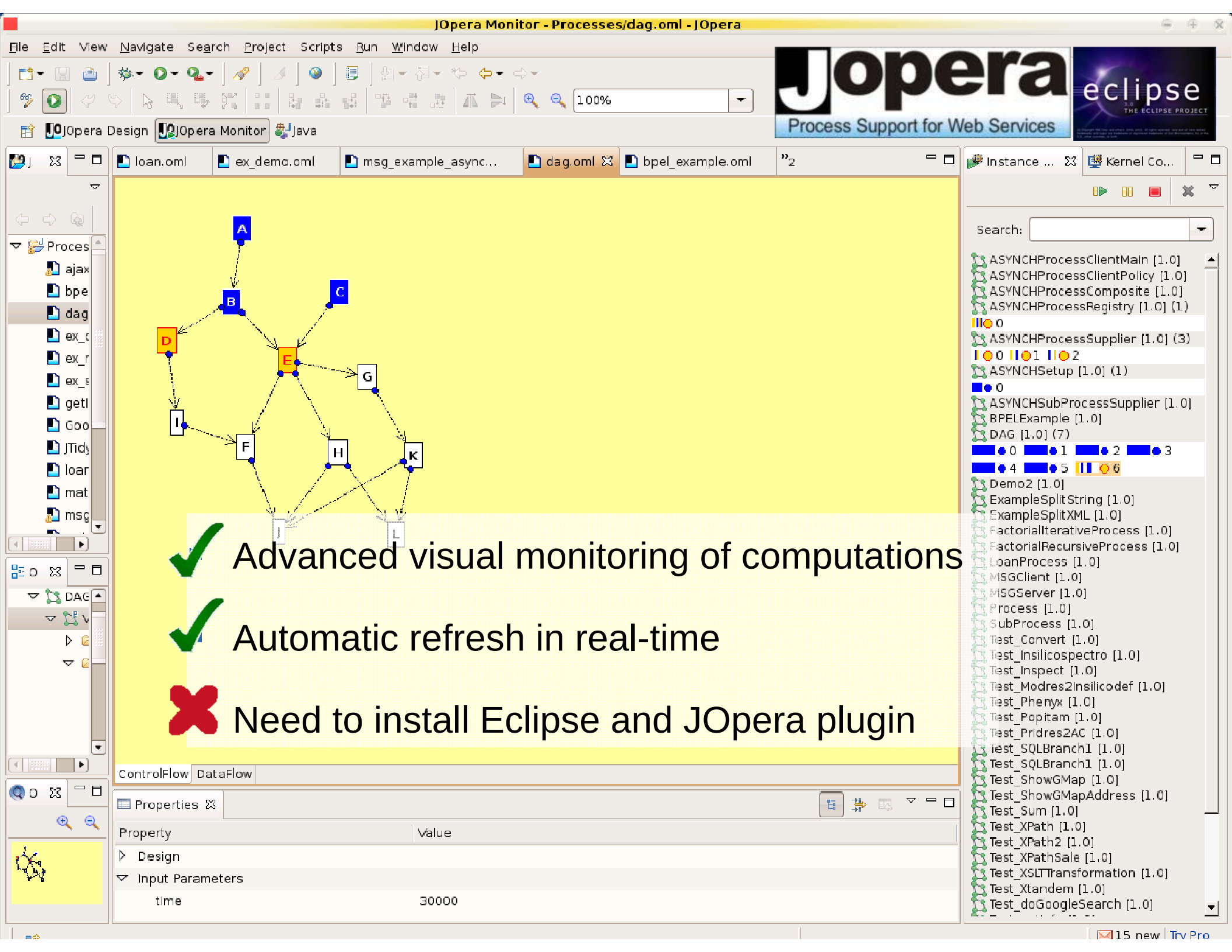
Composable Web 2.0 Widgets for Real-time Business Process Monitoring



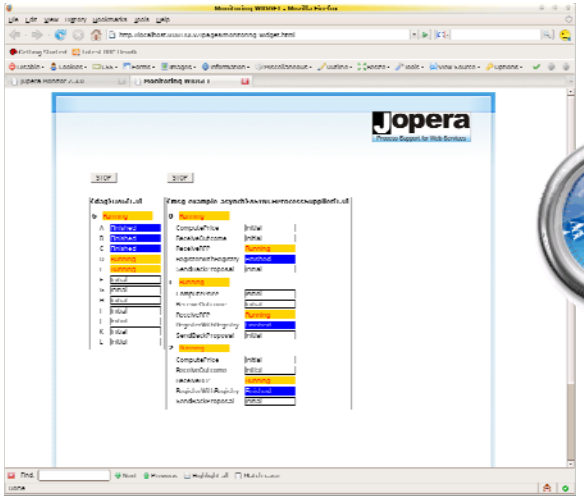
Context

Scientists run computations at CSCS and need to monitor their progress, check for failures, download results

UROP Undergraduate Research Opportunities Program


JOpera Monitor - Processes/dag.oml - JOpera
File Edit View Navigate Search Project Scripts Run Window Help
JOpera Design
JOpera Monitor
Java
JOpera
Process Support for Web Services
eclipse
THE ECLIPSE PROJECT
loan.oml
ex_demo.oml
msg_example_async...
dag.oml
bpel_example.oml
Instance ...
Kernel Co...
Search:
A
B
C
D
E
F
G
H
I
J
K
L
Advanced visual monitoring of computations
Automatic refresh in real-time
Need to install Eclipse and JOpera plugin
ControlFlow
DataFlow
Properties
Property
Value
Design
Input Parameters
time
30000
ASYNCHProcessClientMain [1.0]
ASYNCHProcessClientPolicy [1.0]
ASYNCHProcessComposite [1.0]
ASYNCHProcessRegistry [1.0] (1)
ASYNCHProcessSupplier [1.0] (3)
ASYNCHSetup [1.0] (1)
ASYNCHSubProcessSupplier [1.0]
BPELExample [1.0]
DAG [1.0] (7)
Demo2 [1.0]
ExampleSplitString [1.0]
ExampleSplitXML [1.0]
FactorialIterativeProcess [1.0]
FactorialRecursiveProcess [1.0]
LoanProcess [1.0]
MSGClient [1.0]
MSGServer [1.0]
Process [1.0]
SubProcess [1.0]
Test_Convert [1.0]
Test_Insilicospectro [1.0]
Test_Inspect [1.0]
Test_Modres2Insilicodef [1.0]
Test_Phenyx [1.0]
Test_Popitam [1.0]
Test_Pridres2AC [1.0]
Test_SQLBranch1 [1.0]
Test_SQLBranch1 [1.0]
Test_ShowGMap [1.0]
Test_ShowGMapAddress [1.0]
Test_Sum [1.0]
Test_XPath [1.0]
Test_XPath2 [1.0]
Test_XPathSale [1.0]
Test_XSLTTransformation [1.0]
Test_Xtandem [1.0]
Test_doGoogleSearch [1.0]
15 new
Try Pro

Composable Web 2.0 Widgets for Real-time Business Process Monitoring



1. Monitor the computation from a Browser
2. Use a Widget to let scientists embed monitoring in their own Web pages

Idea

UROP Undergraduate Research Opportunities Program

iGoogle - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address http://www.google.ch/ig?hl=de Go Links

Web Bilder News Groups Bücher Google Mail Mehr ▾ drpautasso@gmail.com | Klassische Startseite | Mein Konto | Abmelden

iGoogle

Erweiterte Suche
Sucheinstellungen
Sprachtools

Google-Suche Auf gut Glück!

Suche: Das Web Seiten auf Deutsch Seiten aus der Schweiz

Google.ch angeboten in: English Français Italiano

Home Seite hinzufügen Neu! Hintergrund ändern | Gadgets hinzufügen »

Datum & Uhrzeit

Di MAI 20

M D M D F S S
1 2 3 4
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31

Wetter

Zurich

14°C
Meistens bewölkt
Wind: NO mit Windgeschwindigke von 19 km/h
Feuchtigkeit: 66%

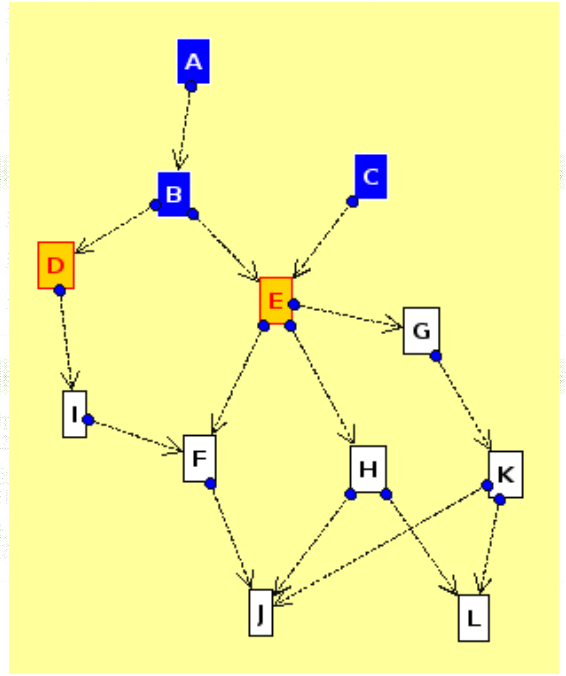
Heute	Mi	Do	Fr
15° | 5°	11° | 9°	18° | 8°	18° | 10°

Turin

16°C
Meistens bewölkt
Wind: S mit Windgeschwindigke von 23 km/h
Feuchtigkeit: 82%

Heute	Mi	Do	Fr
17° | 10°	18° | 10°	20° | 10°	20° | 13°

swissinfo - News Frontseite



Google Mail

Posteingang (1) Vorschau ausblenden Neue Nachricht

Wibke Sudholt - [seed-wg] Draft slides for talk at CCGrid 2008	14:42
bh - invitation to ICPCA08 programme committee - This letter	13:18
newsletter - Newsletter Opernhaus Zürich - Highlights im Jun	12:56
ecows08 - ECOWS 08 - deadline extension - Dear Cesare, F	12:20
Birgitta - European Conf. on Web Services ECOWS 2008 - Ap	12:16

JAJAH

jajah FREE.YOUR.VOICE

My phone numbers:
Home (+393355633361)
Office (+390119351562)
Mobile (+393391368745)

My Friend's Phone Number:
Switzerland
+41
Country Area code and phone number

My account
Log out

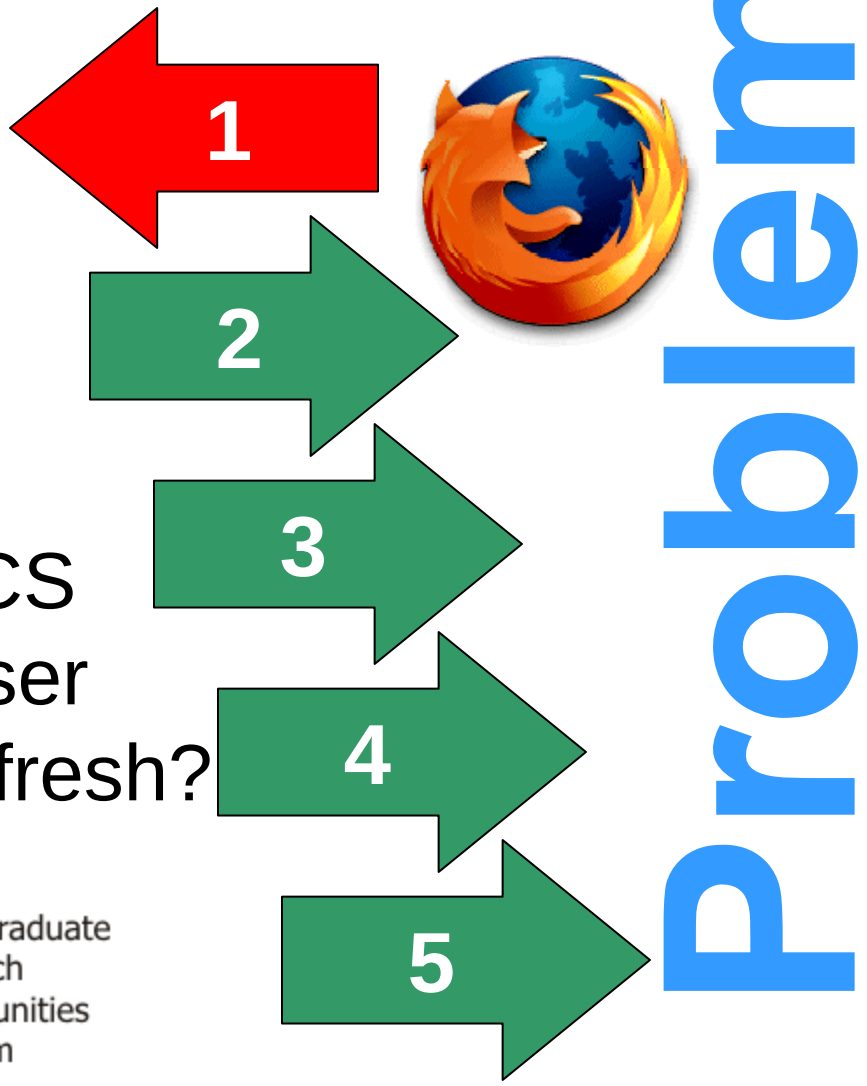
CALL

Erstellen Sie Ihr eigenes Gadget und geben Sie es für Freunde frei.

Internet

start Sent for ce... UropLogo-... Team Sync... Desktop Desktop Talks 5 Micros... 2 Adobe... iGoogle - Mi... EN 69% 4:56 PM Tuesday 5/20/2008

Composable Web 2.0 Widgets for Real-time Business Process Monitoring



How to make events from CSCS immediately appear in a Browser without users doing manual refresh?

Undergraduate Research Opportunities Program

Develop the JOpera Monitoring Widgets

1. **Optimize** the communication of Event Notifications
2. Develop advanced **Visualizations**
3. Help users **Configure** the widget

Your code will be deployed at CSCS

Learn **Java, JavaScript, HTML, CSS, AJAX**

Project

Uniform Sampling in Peer-to-Peer Networks

project supervisors
Cyrus Hall and Antonio Carzaniga

May 2008

Background

The process of *sampling* a peer-to-peer (P2P) network returns a random node from the set of all participating nodes. *Uniform sampling* is a particularly useful form of sampling, as it assures the probability that a node is returned is equal across all nodes. Many services, such as search and replication in unstructured networks like Gnutella, work much better if a P2P network supports uniform sampling. Our research focuses on a generic algorithm called *Doubly Stochastic Converge* (DSC), that we have designed to enable uniform sample of any directed P2P network.

Project Description

Our goal this summer is to deploy DSC with several mature P2P protocols. The precise targets depend partially on the student and initial findings as the work begins, but possible P2P networks include one of several Kademlia implementations, Vrije University's social and file-sharing application Tribler, or the anonymous file store Freenet. If it becomes clear that integration with existing projects is not possible within the given time frame, we will use simulated versions of the various protocols instead.

Organization and Requirements

The project will be under the supervision of Antonio Carzaniga and Cyrus Hall, and will last the full 8 week period of the UROP program. An applying student must have an avid interest in networking and have completed the Computer Networking class. Further, the student should be comfortable with Java (anonymous inner classes, serialization, generics), and have a desire to learn other programming languages. Most importantly, applicants must enjoy learning about and using new techniques and technologies, including peer-to-peer systems and possibly network simulation.

Undergraduate Research Opportunities Program

Adagio – Why is it so slow?
Mentor: Matthias Hauswirth

In the Adagio project you will design and develop a tool that automatically identifies performance regressions in Java programs.

Imagine a Java programmer, let's call him Felix, who is working on a big open-source project together with several very productive developers. They use subversion as a common repository. Before committing a change to their software, they always run a big suite of regression tests. This way their changes rarely break the system. However, they found that sometimes changes and new features significantly degrade the performance of their application. They always had a hard time to figure out how. Until they started using Adagio.

The goal of the Adagio project is to automatically compare two versions of a Java program, measure the performance of both, determine whether the performance significanly differs between versions, and identify the causes (specific changes in the source code) responsible for that performance regression.

If you are a very strong Java programmer *and* are interested in (learning more about) virtual machines, compilers, how to automatically analyze and change the source code or the byte code of Java programs, or how to measure the performance of a Java application, this may be a project for you. This is a big project, so there are many different aspects you could work on. Come see me to discuss which aspects would match your skills and interests.

Example Project Outline

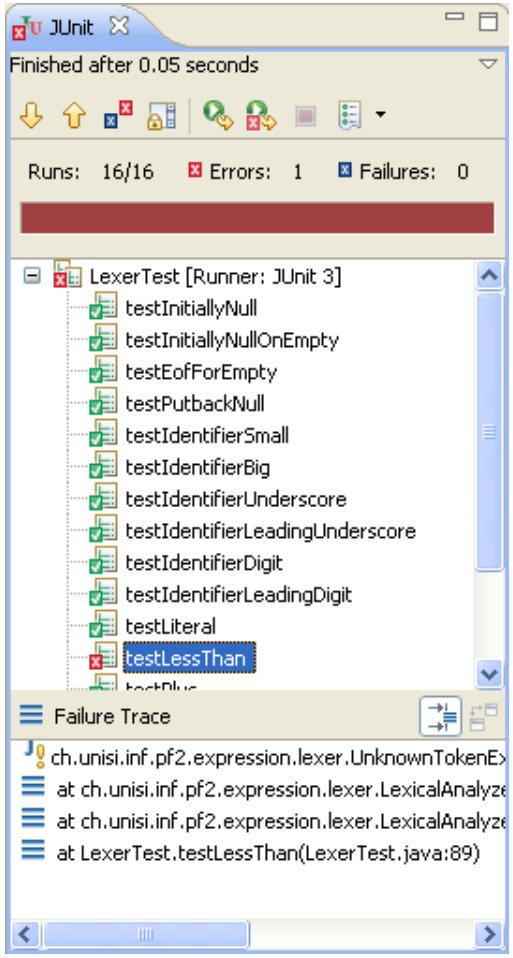
One possible way in which to progress in this project would be for you to create an Eclipse plugin that extends JUnit to allow regression *benchmarking*. Instead of checking whether a test case passes or fails, this plugin would measure the performance of that test case and report a problem (a performance regression) if the test case ran significantly slower than expected.

In a first phase you would measure performance by simply measuring the wall-clock time between the beginning and end of the test case execution. You would extend the JUnit Eclipse GUI to present the benchmarking results in a way similar to test case results. In later phases, depending on your skills and interests, you could:

- Use a richer set of performance metrics (e.g. memory consumption)
- Measure resource consumption separately for each executing Java thread
- Run regression benchmarks against older versions of the program (retreived from version control)
- Use statistical techniques to determine when a performance difference caused by a change is significant
- Use techniques that try to identify the specific cause (e.g. which new lines of code) of a performance regression
- Execute regression benchmarks remotely (to prevent slowing down your desktop)
- Automatically run regression benchmarks each time the user does an edit in Eclipse (to better detect the causes of performance regressions)

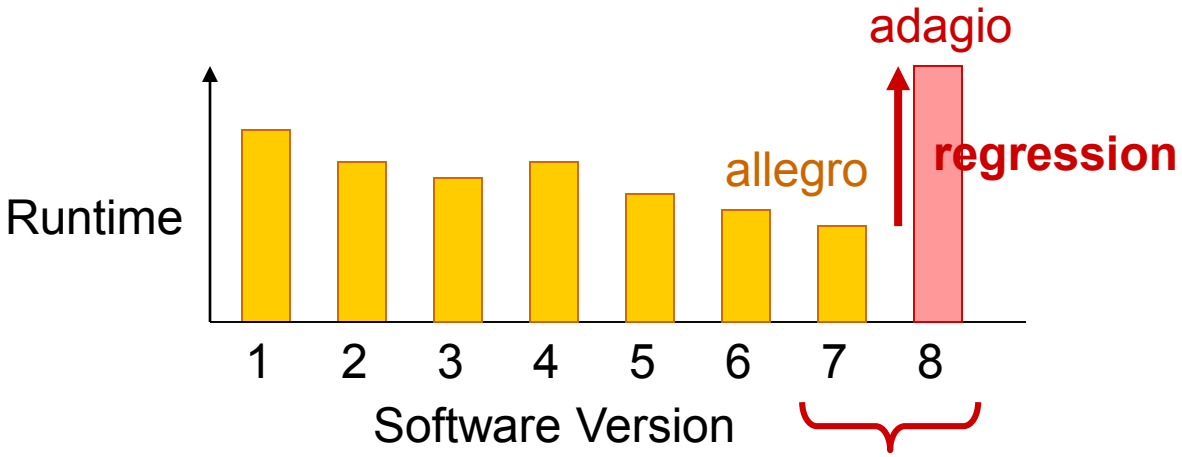
Note that this is just one example of what you could do within the Adagio project. There are many alternative opportunities to focus on specific areas you are excitied about. If you are very strong in Java development and would like to know more about this project, please contact Matthias Hauswirth.

Adagio – Why is it so slow?



Regression Testing → Correctness

Regression Benchmarking → Performance



Undergraduate Research Opportunities Program

Mentor:
Matthias Hauswirth

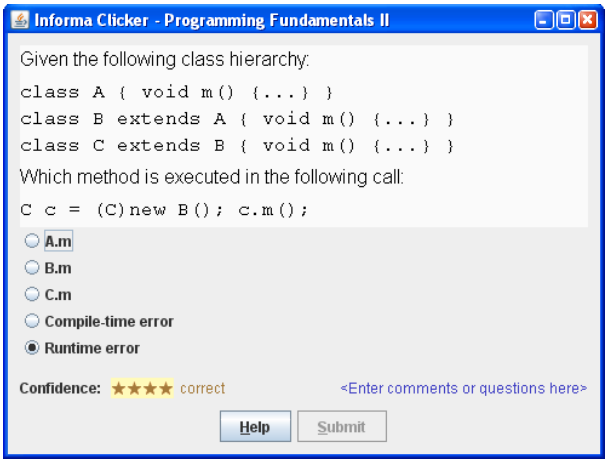
Undergraduate Research Opportunities Program

Informa: Classroom Response System for Teaching Programming

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 during the lecture, 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.

It is not the goal of Informa to automatically *grade* student assignments. Usually students' solutions are anonymous and Informa only provides aggregate information to the instructor and the class. The goal of the automatic aggregation of student solutions is to find overall misunderstandings and point them out to the instructor, so the instructor can immediately provide clarifications in class.



Currently Informa supports a few simple types of questions. For example, an instructor can post a multiple-choice question. The instructor application collects the answers and displays them in a histogram. Multiple-choice questions, however, only allow a very narrow view of a student's understanding. For this reason, Informa is a generic system, where new question types can be added as plugins.

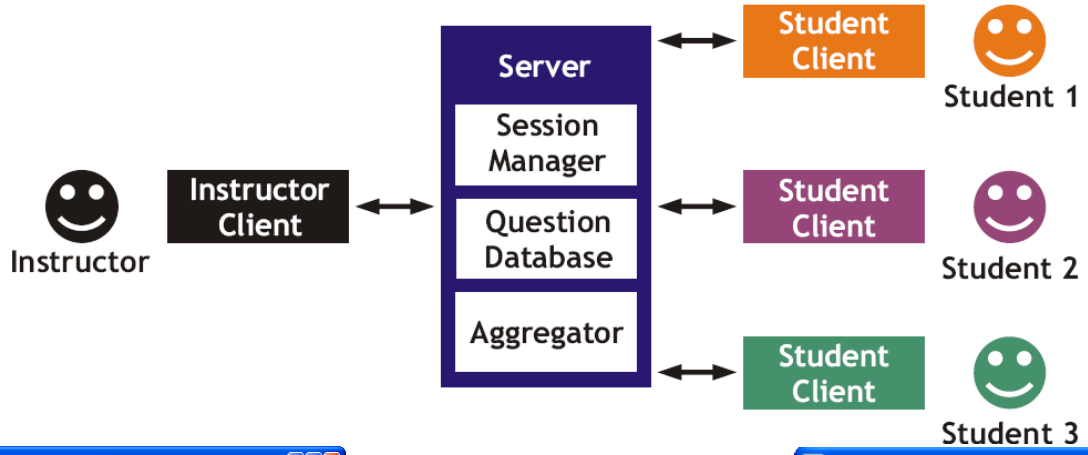
Your Contribution

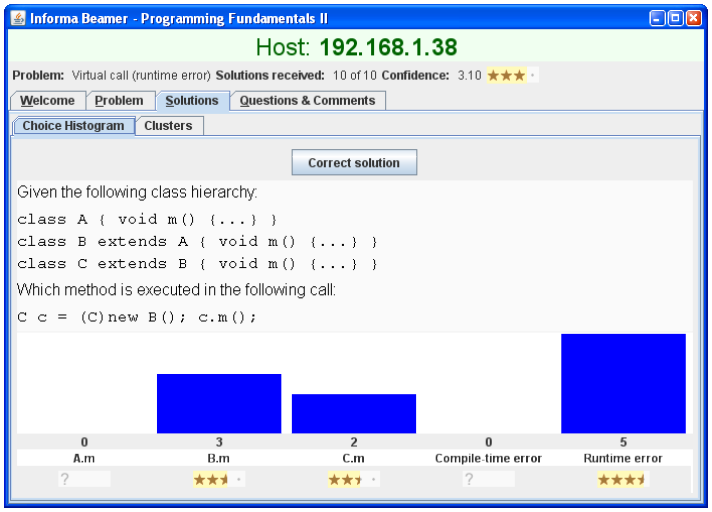
In this project you will develop a plugin for Informa to support a new type of question: Java programming assignments. On their clicker application, students will be able to use a built-in editor to solve small Java programming problems. That editor will support syntax highlightning and possibly other relevant features. When they submit their code, the instructor application will compile the code, and if it does compile, run it against a suite of unit test cases provided by the instructor. It may also measure the performance of the application, run certain static analysis tools to check for errors or code style, or compare the submitted solution against an instructor-provided reference solution. The instructor application will then aggregate the information from all submitted solutions to provide the instructor (and, via the classroom beamer, the students) with an overview of the problems the students encountered.

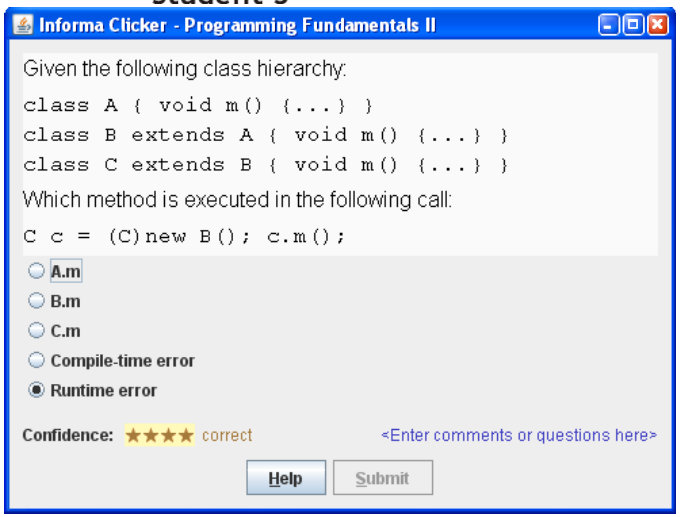
Resources

Informa is implemented in Java and uses RMI for communicating between the instructor application and the students' clicker applications. To implement your plugin, you can reuse existing open-source components, such as JEdit's text editor, JUnit's unit testing framework, or static analysis tools like Checkstyle.

Informa: Classroom Response System for Teaching Programming







Undergraduate Research Opportunities Program

Mentor:
Matthias Hauswirth

Undergraduate Research Opportunities Program

Speculative GUI Components

Mentor: Matthias Hauswirth

How many times have you clicked on a little "plus" sign next to a directory tree node in Windows Explorer (or an equivalent application) and then had to wait for a few seconds until the node expanded to expose its subdirectories? Does behavior like this annoy you? The goal of this project is to eliminate these kinds of annoying pauses in interactive applications.

Background

The feature-rich GUI toolkits for Java enable the relatively easy development of rich GUI applications. The two major toolkits for Java, AWT/Swing and SWT/JFace, provide dozens of powerful components. The more complex components, such as lists, tables, and trees, are based on a derivation of the model-GUI-controller design pattern. These components (e.g. Swing's JTree) present data which is maintained in a model. They access the model through well-defined interfaces (e.g. Swing's TreeModel). If used properly, the components only fetch data (list rows, table cells, or tree nodes) from the model when that data really needs to be displayed. This lazy approach allows to quickly display very large lists, tables, and trees, since only the small visible subset of the model needs to be instantiated. However, when a user navigates the data in these components (e.g. by scrolling a table or expanding a tree node), the components will have to fetch more data from the model, and the model may have to first produce that data (e.g. fetch it from a database, from disk, or compute it). This can lead to significant delays, and thus to a decrease of the perceived performance of the application.

Objective

In this project you will improve the perceived performance of such GUI components by using speculative prefetching. This technique has already been used in the context of web browsers: a prefetcher guesses which links the user will follow next, and starts downloading the corresponding pages ahead of time. This project applies this idea to general-purpose GUI components such as lists, tables, and trees. You will write a prefetcher that hooks into the GUI component and predicts when which data item will need to be displayed. Your prefetcher will then request that item from the model speculatively. At the time the user scrolls or expands the component, your prefetcher will already have fetched that data from the model, and the component will be able to render the new data without any perceptible delay.

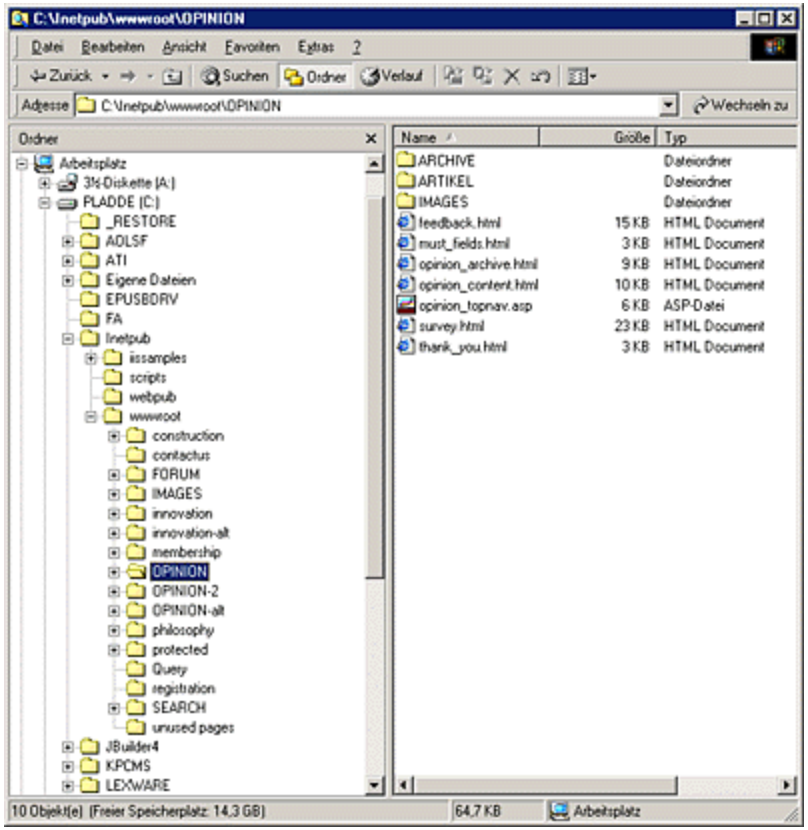
Depending on your interest and experience, you can use AWT/Swing and/or SWT/JFace for this project. In a first phase you will focus on one of the three components (e.g. the tree), and implement and evaluate a prefetcher. In a second phase, you can expand your work to other components (e.g. to a table), or you can develop more advanced prefetching heuristics.

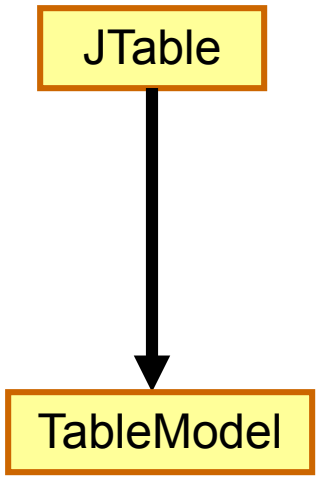
Technologies

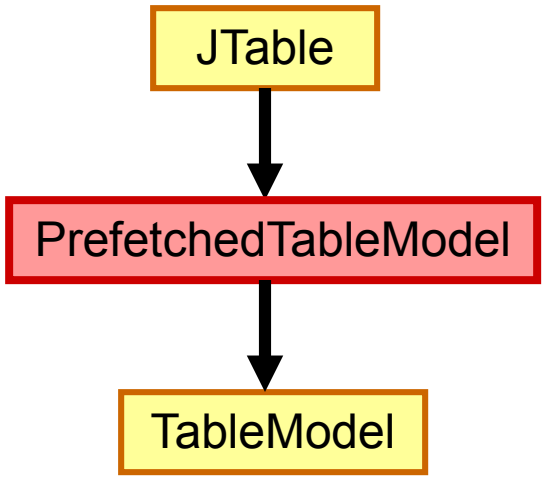
This project is heavily focused on Java. You will have to understand and exploit the object-oriented design of Java GUI toolkits, and you will have to be creative in inventing clever ideas for predicting the user's next actions (e.g. when and in which direction they will scroll, or which tree nodes they might expand).

If you are strong in Java development and would like to know more about this project, please contact Matthias Hauswirth.

Speculative GUI Components







Undergraduate Research Opportunities Program

Mentor:
Matthias Hauswirth

Undergraduate
Research
Opportunities
Program

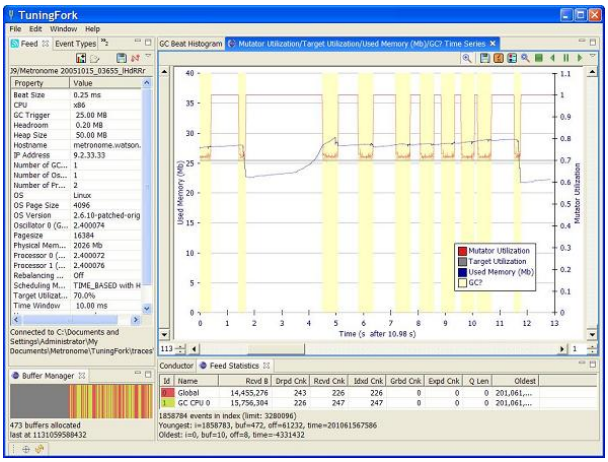
Dynamic On-the-fly Code Generation for Efficient Trace Analysis & Visualization

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.

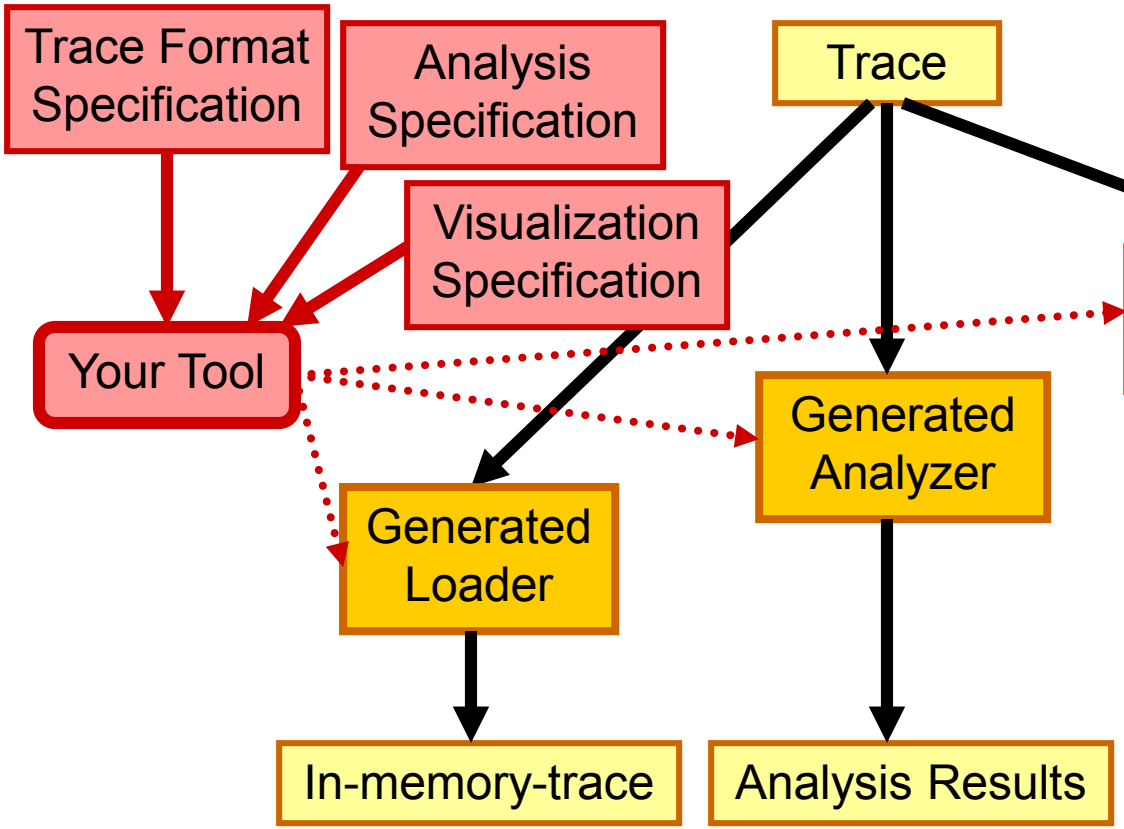
Due to this heterogeneity and size of the traces, classical tools for data analysis and visualization often break down. Researchers thus often produce special purpose tools for each new type of trace they need to analyze. At USI and at the IBM T. J. Watson Research Center we are developing a flexible 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.

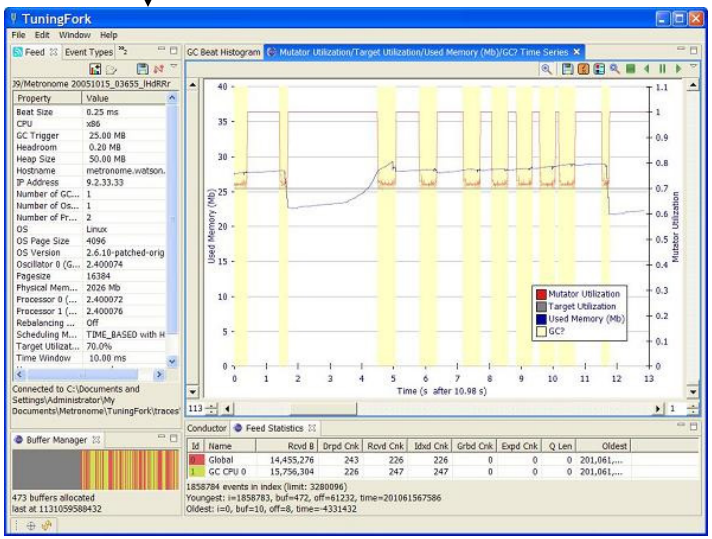


In this UROP project you will develop a crucial component for this infrastructure. Your component will compile trace format specifications (definitions of all event types and time series) into Java classes for the efficient processing and in-memory storage of corresponding traces. The component will 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 could use a parser generator (such as ANTLR) for producing a parser of the trace format specification, and a byte code manipulation toolkit (such as ASM) to generate the Java class files.

Depending on your interest, you can focus on generating efficient code for (1) loading and representing traces in memory, (2) online (streaming) processing of traces, or (3) visualizing of traces. In any of these three cases, the user should be able to provide a high-level specification of the trace format and of what to do with the trace, and your tool should generate the Java code that performs the corresponsing work.

Dynamic On-the-fly Code Generation for Efficient Trace Analysis & Visualization





Undergraduate Research Opportunities Program

Mentor:
Matthias Hauswirth

Title: **Building a Web 2.0 Application using a social and semantic approach**

The Semantic Web is an extension of the current Web that will allow people to find, share, and combine information more easily. It relies on machine-readable information and metadata expressed in RDF format (Resource Description Framework). Social applications refer to software that allows users to interact and share data with other users: consider MySpace, Facebook, flickr or YouTube as successful examples.

Nepomuk is a modern platform that simplifies the creation of Collaborative and Semantic based applications by providing a set of common Web Services and uses simple and flexible Java API.

The student will implement a collaborative web application using this platform and he/she will contribute to the understanding of the usability of the proposed platform. Once the adoption of semantic technologies is included, there are no constraints on the application that could be built. Examples include: photo sharing or tagging, document annotating and so on.

bio: **Francesco Lelli** is taking post-doctorate studies at the University of Lugano under the supervision of the Dean Mehdi Jazayeri. His research interests include High Performance Computing, Grid, Web Services, Web 2.0, Peer to Peer, and Artificial Intelligence. He followed master studies in Computer Engineering at the University of Pisa and, after a short experience in the industry he followed PhD studies at the University of Venice. Within that time he also worked as full time Research Associate at the Italian National Institute of Nuclear Physics (INFN) in Legnaro (Padova) and as post-doc after.

bio: **Sasa Nesic** received the B.S., and M.S degrees in informatics and computer engineering from the Department of Computer Science, University of Nis, Serbia, in 2002 and 2004 respectively. Currently he is a PhD student at the Faculty of Informatics, University of Lugano, under the supervision of the prof. Mehdi Jazayeri. He is associated with NEPOMUK project - The Social Semantic Desktop (Networked Environment for Personalized, Ontology-based Management of Unified Knowledge). His research interests include Semantic Web, Document models and representation languages, Knowledge representation, and Semantic annotation.

Building a Web 2.0 Application using a social and semantic approach

Sasa Nesic, Francesco Lelli

Recent trends in the WWW

- Growth of Web 2.0 systems which are characterized by:
  - Easy-to-use
  - Interactive
  - Participatory
- Active role of users in Web 2.0 applications
  - Add and annotate content with standardized metadata (DC, LOM, etc.)
  - Create and manage tags (social tagging)
  - Rate content
  - Create content with sorts of online diaries (blogs), etc.
- The Web 2.0 focuses on the usage-dimension

Combining Web 2.0 ideas with semantic technologies

- The enhancement of the semantic-dimension improve:
  - Accessibility
  - Provide means to reason about Web content
- Resources are embedded in a machine readable context;
- Knowledge bases (ontologies) provide pointers to both the content and the context of Web resources;
- Combining Web 2.0 with semantic technologies gives benefits to both approaches;

NEPOMUK – the Social Semantic Desktop

- Two core groups of services:
  - Purpose specific: Data Wrapping, Context Elicitation, Mapping and Alignment, and Text Analytics.
  - Data Services: Data storage and Data search
- Building Web 2.0 systems using NEPOMUK platform
- Case study of the usability of the NEPOMUK platform

Domain Specific Languages for Model Analysis and Translation

Project mentor/sponsor: Jochen Wuttke, Mauro Pezzè

Model driven techniques can improve todays software development methodologies in various ways. Two important factors are that (1) model driven techniques can speed up the development cycle, because many tedious manual tasks can be automated. (2) The automation of frequent, but tedious tasks can improve the quality of the produced software by reducing errors introduced by developers.

One challenge in the development of model driven tools and techniques is that the required model analysis and transformation is complex and often domain or even application specific. They thus require flexible techniques that can be customized easily to fit individual project needs.

Domain specific languages tailored to the task of model analysis and transformation are on step towards the realization of such flexible techniques. Instead of hard-coding analysis and transformation rules in MDA tools, DSLs provide facilities to define and execute these rules. Together with a library of standard functions, this approach can significantly ease the definition and customization of project specific rules. It even invites developers to experiment with various possible rules, because change and evaluation are quick and easy.

Project Details

The focus of this project defining a domain specific language framework for model analysis and transformation. The implementation language for the framework should be a dynamic language, such as Ruby or Groovy, that integrates well with the Java platform.

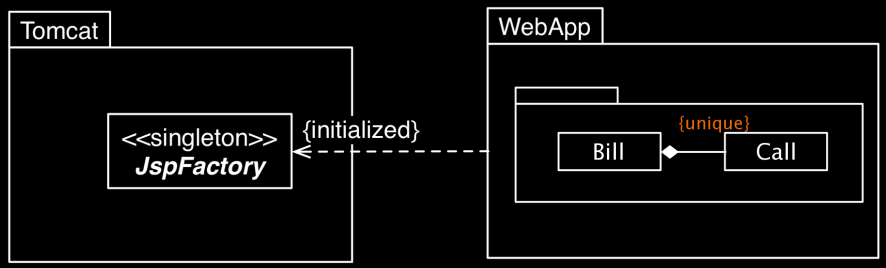
Open research questions in this project include:

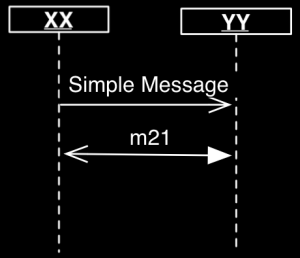
- What are the right abstractions that the framework has to provide?
- Which aspects of model transformation are inherently application specific and cannot be incorporated in the framework?

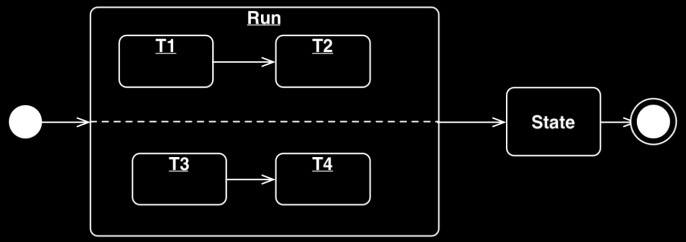
Project Goals

The expected outcome of the project is a prototype implementation of a DSL and a set of test cases to demonstrate the capabilities of the language. The minimum capabilities of the DSL include *context determination*, *pattern extraction/matching* and *translation rule templates*. At least XMI must be supported as input language for models to be processed. Ideally the prototype will be integrated as an Eclipse plugin, but this is strictly optional.

DSLs for Model Transformation



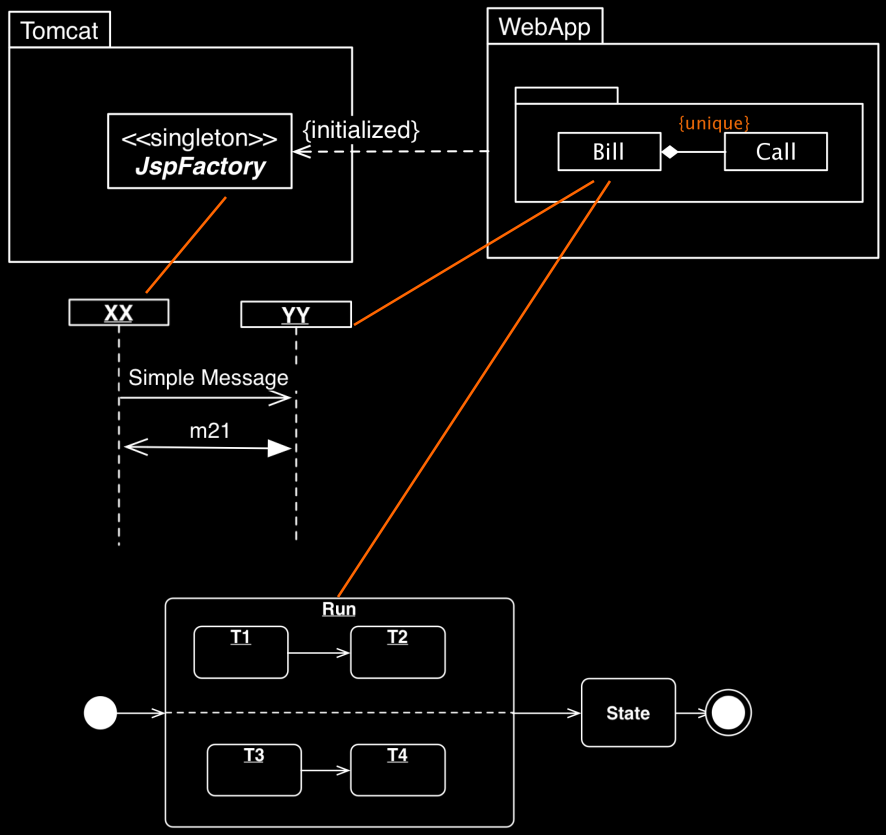


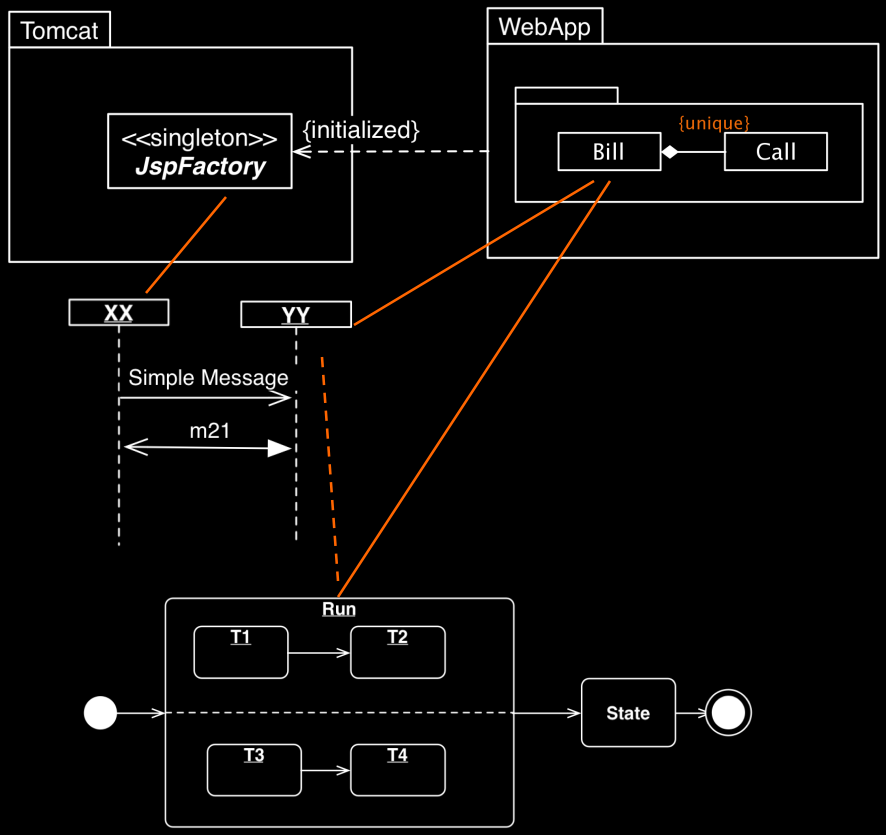


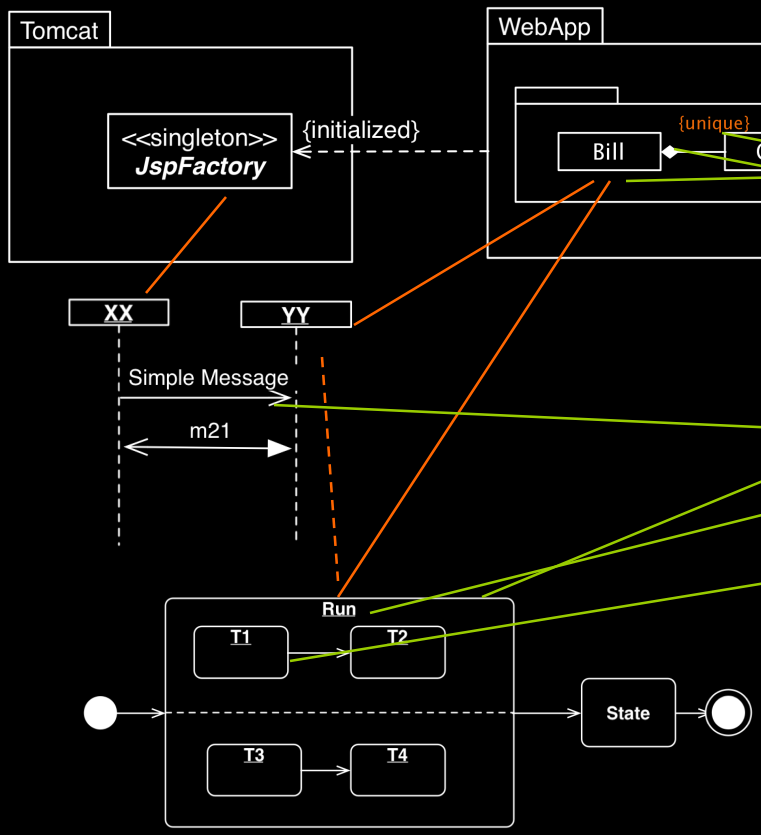

```
public class Bill {
private Set<Call> calls;
private State state;

public void run() {
   yyObject.simpleMessage();
   state = T1.getState();
// ...
}

//...
}
```


Tomcat
<<singleton>>
JspFactory
{initialized}
WebApp
{unique}
Bill
Call
XX
YY
Simple Message
m21
Run
T1
T2
T3
T4
State


Tomcat
WebApp
<<singleton>>
JspFactory
{initialized}
{unique}
Bill
Call
XX
YY
Simple Message
m21
Run
T1
T2
T3
T4
State




```
public class Bill {
private Set<Call> calls;
private State state;

public void run() {
   yyObject.simpleMessage();
   state = T1.getState();
// ...
}

//...
}
```

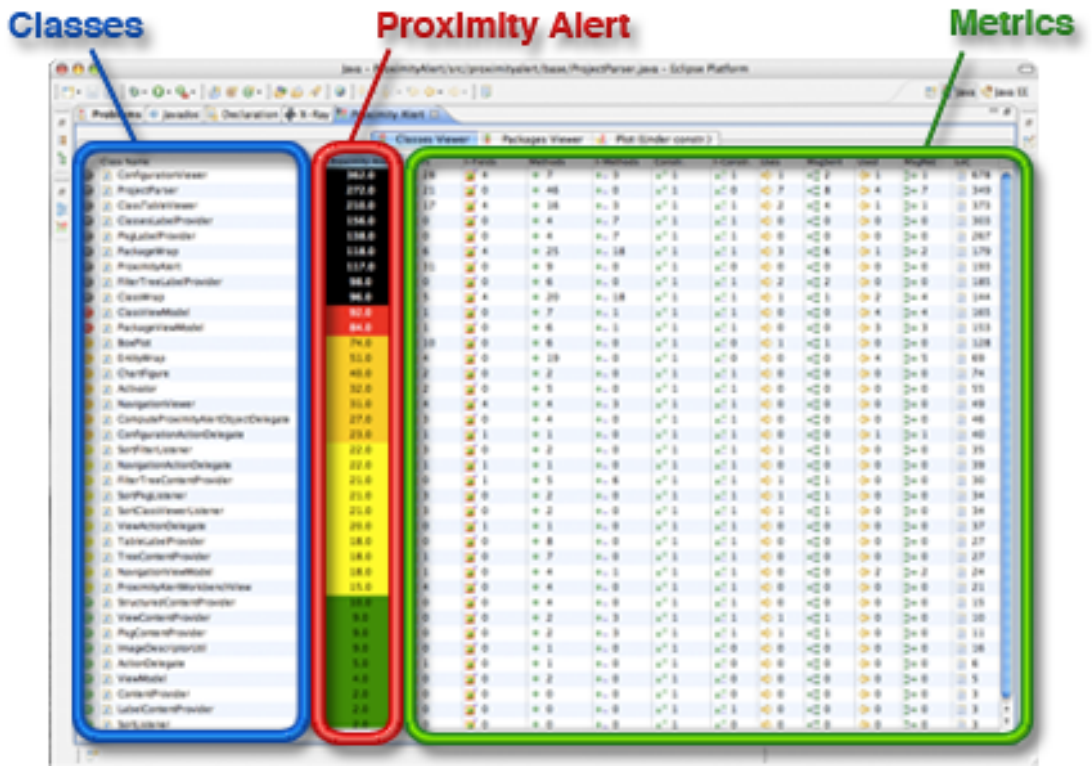
Project Challenge

- ✓ Define a DSL for model analysis and transformation
  - ✓ Sort our the mess of different model views
  - ✓ Find the right level of abstraction in DSL
  - ✓ Implement DSL in Groovy
  - ✓ Define good code generation rules
  - ✓ Build working examples

ProximityScan

UROP Project 2008

Advisor: Michele Lanza
Student: Andrea Casarella



Project Description

The goal of this project is to extend ProximityAlert, an Eclipse plugin for computing coupling metrics with a series of enhancements, such as:

1. Visualization support

2. Customizability of the Proximity Alert metric

3. Integration with the X-Ray Eclipse plugin

4. Export of metrics to CSV files

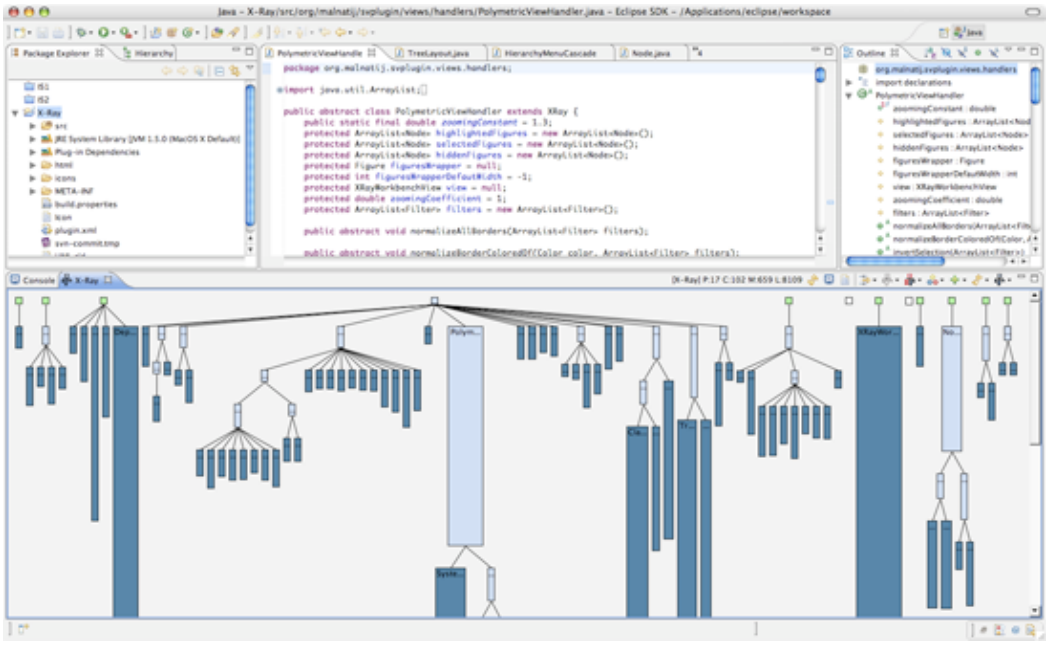
5. Automated report generation

6. etc.

X-Ray3

UROP Project 2008

Advisor: Michele Lanza
Student: Jacopo Malnati



Project Description

The goal of this project is to extend X-Ray, an Eclipse plugin for visualizing Java source code with a series of enhancements, such as:

1. Explicit declaration of a meta-model for Java source code

2. Export of models towards the mse format used by the Moose reengineering environment

3. Integration with the ProximityAlert plugin to enrich the set of available metrics

4. Clear separation of the currently monolithic plugin into a set of distinct plugins

UROP Project – Summer 2008

Monica Frisoni, Prof. Cesare Pautasso

Browser-based Visualization of Architectural Decision Spaces

Background

Architectural decision modeling is used to gather recurrent design decisions and help software architects share their knowledge and experiences. Since for an average-size project there are hundreds of decisions to be made, it is important to help architects navigate in such large decision space to keep its complexity under control. Also, architectural decisions (and their corresponding architecture alternatives) influence one another and it is important to understand the impact of a decision on the rest to be able to identify upfront the critical decision path for a given project and technology context. The project will leverage the current IBM alphaworks decision model database recently released by Olaf Zimmerman's team at the IBM Zurich Research Lab.

Objectives

The main goal of the project is to develop a graphical tool to display and interact with a potentially large set of architectural decisions. The tool will enable software architects to navigate through a graph of related decisions laid out according to different criteria (e.g., grouped by abstraction level, scope, importance), explore and visualize different kinds of relationships between decisions/alternatives, and perform what-if analysis (suppose an alternative is taken, the impact on the design space should be visualized).

Technologies

The tool will run in the Firefox browser and be built mostly using JavaScript, XML/SVG in the browser and some Java/PHP on the Web server.

Contact

For more information, please contact Monica Frisoni <monica.frisoni@lu.unisi.ch>

Flash-Based Database Management Systems

project supervisor
Fernando Pedone

May 2008

Background

Most of today's Database Management Systems use magnetic disks for persistent storage. Magnetic disks are non-volatile, which means that when power is turned off, data persists, and is not lost. Due to the mechanical parts of the disk, access latency is quite high. A recent storage medium, flash memory, has been widely adopted as data storage for mobile computers, such as PDA's, MP3 Players, mobile phones, digital cameras, etc. Flash memory has some advantages over magnetic disks: smaller size, lighter weight, better shock resistance, lower power consumption, less noise and faster read/write performance. This type of memory, as disks, is non-volatile.

Project Description

A current approach that has been proposed for running DBMS with flash-memory is called In-Page Logging (IPL). The idea is to log the changes done to data pages, and then write back to flash memory only those logs. Without writing entire pages the write performance can be increased. However, the In-Page Logging approach presents some problems: (a) The read operation has to get all the logs belonging to a page, and then apply the interesting logs to this page, before returning it. (b) The merge operation might copy around pages that have not been modified, and that actually could be left where they are.

The goal of this project is to identify the details of the IPL problems, and provide solutions for optimizing flash-based database management systems further.

Organization and Requirements

The project will be under the supervision of Fernando Pedone, and will last the full 8-week period of the UROP program. An applying student must have an avid interest in databases and have completed the Computer Systems class. Further, the student should be comfortable with systems programming, a programming language (e.g., Java, C++, C) and notions of simulation tools (e.g., discrete event simulation).

Robtic Lab Demonstrators and Educational Framework

Mentor Alexander Förster

UROP 2008

Background

The IDSIA/SUPSI/USI robotic lab was founded 2008 with the goal of having a centralized laboratory for robotic related research and education. Different robotic platforms and equipment were purchased to provide a wide spectrum of robotic types: small palm size robots, autonomous out door robots, robot arms, humanoid robots. All these platforms have different application programming interfaces, control software, application frameworks, and algorithmic libraries, based on the robot's capabilities, application scenarios, but also dedicated customers and manufacturers philosophies.

Most of the software was developed for Microsoft Windows, especially applications with a graphical user interface. Some low level programming interfaces are also available for Linux and can be compiled with modifications under Mac OS X with some restrictions. Most of these low level libraries are open source and published by the robot manufactures under the GNU public license. Some non free manufacturer developed software can be substituted by free software from the robot's user community.

Project Description

Mac OS X is widely used both in the faculty of informatics at USI and IDSIA, being a user friendly platform for education and research. The goal this summer is to develop an easy to understand robot programming interface under Mac OS X, primarily for educating basic principles in robotics at USI and SUPSI. The interface has to fulfill two goals: to provide a simple unified programming interface of all available robot platforms and at the same time to give the user the full control over each individual hardware component. Important features of the project beside the programming interface development are also

- a well written documentation of the programming interface,
- a detailed compilation and installation description for dependent software, libraries, and necessary embedded software on the robots,
- demo applications to demonstrate the characteristics of the programming interface and the properties of the specific robots.

The demo applications for the robots should be explained by posters which help visitors of the lab to understand the key concepts of the demonstration. Demo applications should be implemented for one or two different robot systems. The design of the programming interface should respect potential follow-up projects, like the integration with a simulator, e.g. the Webots simulator, extensions for other hardware and cross-compilation potentialities for the embedded micro-controller systems on the robots.

Organization and Requirements

The project will be under the supervision of Alexander Förster and will be located at the robotic lab in Manno. The project will last 8 weeks. An applying student must have an avid interest in robotics and have completed the Robotics class or traceable comparable skills. Further, the applicants should be comfortable with C/C++ and must enjoy learning and using new techniques, including low level device programming as well as high level behavior control.