Piattaforme Software per la Rete
Course Projects, part 2

Giovanni Agosta

Piattaforme Software per la Rete – Modulo 2

Outline

Rules of the Game

Short-term goal This is the goal that must be reached in order for the project to be evaluated.

Long-term goal While the long term goal needs not be reached within the project timeframe, the documentation and planning should reflect the goal, making it so that future development do not need to reengineer completely the application.

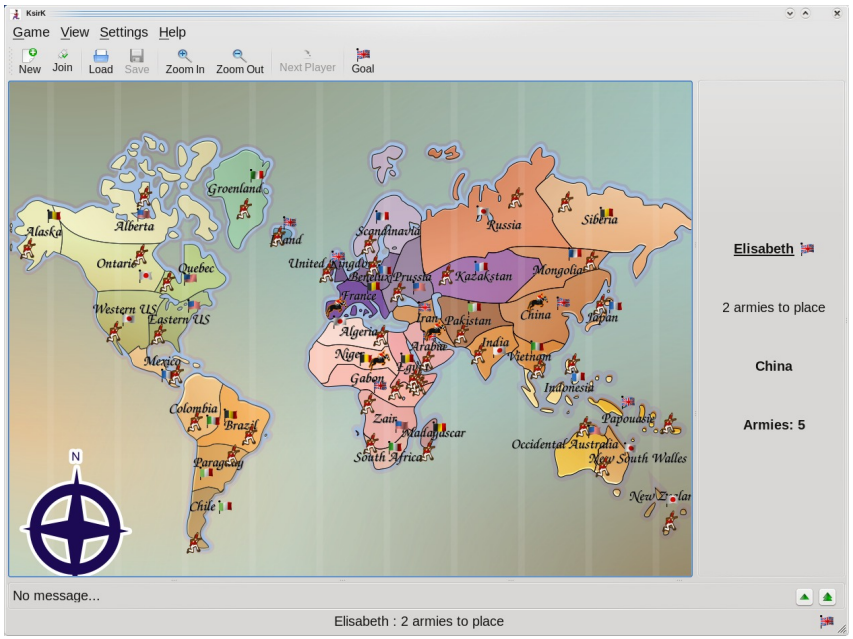
Technical details The required tools in terms of programming languages, libraries, documentation and version control tools. These are mandatory.

KsirK Android port

Native app with Qt

Description

Ksirk is a Qt Risk clone. The goal of this project is to port the game to the Android platform.



kbreakout Android port

Native app with Qt

Goals and Techniques

Short-term goal Functional port of the Krisk game.

Long-term goal Network multiplayer support.

Educational value Learn a major GUI framework (Qt), learn the Android native app development process.

Technical details C++ language, doxygen documentation, version control with google code or github; Android SDK.

OpenTyrian Re-engineering

Description

OpenTyrian is a scrolling space combat game. It has been released as Open Source and successfully reimplemented in C, but the implementation suffers from old “spaghetti” coding style. A clean, reader-friendly implementation is the goal of the project.

OpenTyrian Re-engineering

Goals and Techniques

Short-term goal remove global variables as much as possible employing meaningful names, use C exact types for structure definition.

Long-term goal reengineering of the codebase, integration of TCP/IP network game support.

Educational value put into practice the notions of practical programming learnt in the course.

Technical details C language, Google Code with mercurial/Github w/git.

P2012 Benchmark development

Image recognition algorithms

Description

STMicroelectronics Platform 2012 is a novel many-core accelerator for high end embedded systems. We want to compare its performance and programmability to those of GPUs and multi-core CPUs by developing an implementation of the Canny algorithm. The P2012 SDK, including simulator and emulator, will be used for this work.

P2012 Benchmark development

Image recognition algorithms

Goals and Techniques

Short-term goal design and implement the algorithm using the P2012 SDK.

Long-term goal allow for different optimizations for P2012, GPUs and CPUs.

Educational value develop an application for an increasingly common architectural model (programmable parallel accelerators for high end embedded systems).

Technical details OpenCL API and programming language; doxygen documentation, version control with mercurial on POLIMI servers.

Hardware-accelerated DTLS

SPEAr C3 PolarSSL integration

Description

The STMicroelectronics SPEAr platform includes a C3 crypto accelerator. The C3 crypto accelerator is an ASIC crypto accelerator embedded in modern ARM Cortex platforms by STM and already has a working interface in the form of linux device files. The goal of the project is to implement the DTLS protocol (TLS over UDP) within the PolarSSL library. A board endowed with the C3 Accelerator and the properly configured Linux distribution will be made available via SSH.

Hardware-accelerated DTLS

SPEAr C3 PolarSSL integration

Goals and Techniques

Short-term goal Support a DTLS session with one cipher suite.

Long-term goal Support the entire DTLS protocol.

Educational value work with standard technologies for embedded encryption (PolarSSL) and develop for a state of the art high-end embedded processor.

Technical details C language; doxygen documentation; version control with mercurial on POLIMI server; PolarSSL.

Amiga Emulator GUI

Adding a Qt GUI to the UNIX Amiga Emulator

Description

- The `fs-uae` is a command line emulator for the Amiga system able to emulate different Amiga machines when provided with firmware and disk images
- `fs-uae` is currently missing a well structured GUI

Amiga Emulator GUI

Adding a Qt GUI to the UNIX Amiga Emulator

Goals and Techniques

Short-term goal Obtain a working GUI that can control UAE with the same capabilities of the command line and a basic save-restore settings functionality.

Long-term goal Provide the gui with firmware and disk image management features, package for major distributions

Educational value Learn a major GUI framework (Qt) and C++ fundamentals.

Technical details C++ language, doxygen documentation, version control on Google Code/Github

Dynamic Code Re-Scheduling

Re-Scheduling ARM machine code

Description

To hide the actual operation of a function, it is useful to periodically (randomly) re-order the instructions, while preserving the semantics. A code scheduler should be implemented that is able to generate equivalent but re-scheduled functions. The scheduler must be integrated in an existing suite of side-channel countermeasures, and needs to be extremely fast.

Dynamic Code Re-Scheduling

Re-Scheduling ARM machine code

Goals and Techniques

Short-term goal Develop the code scheduler.

Long-term goal Support rescheduling of instructions that need recomputing of relative addresses.

Educational value Understanding of the ARM ISA; understand the nature of dependencies among machine instructions.

Technical details C and ARM assembly language; doxygen documentation; versioni control with Mercurial.

SPEAr Barebone mode

Running programs on SPEAr without OS

Description

Embedded systems often execute a single program, without an underlying operating system. The goal of the project is to build a framework to easily load and execute code on an STM SPEAr board (provided) without using an operating system.

SPEAr Barebone mode

Running programs on SPEAr without OS

Goals and Techniques

Short-term goal Be able to run C source code controlling General Purpose Input Output (GPIO) pins and handling the communication with a host PC via serial port.

Long-term goal Provide a flexible C library to run programs on the bare metal on the board

Educational value Learn to work in an embedded scenario, with a real world ARM processor implementation.

Technical details C language; doxygen documentation; any versioning system (git, mercurial).

Data recovery

Data carving software

Description

Common data carving for data retrieval purposes is performed by tools which recognize beginning and end sequences and save whatever there is in between as the "recovered file". `binwalk` is one of the most common tools to perform this task on a single file, but the matching is made in a trivial way rendering the program slow in analyzing files

Data recovery

`binwalk` extension

Goals and Techniques

Short-term goal Implement the matching strategy storing the rules in a tree, parallelize analysis with pthreads

Long-term goal obtain a tool able to match also common infixes of a file

Educational value learn efficient programming in C, learn to understand someone else's code

Technical details C language, doxygen documentation, GitHub/Google Code versioning (the project is hosted on Google Code)