Austrian Cubes
Team Description Paper for RoboCup 2009

Patrick Sturm, Emanuel Plochberger

University of Applied Sciences Technikum Wien
Hoechstaedtplatz 5, 1200 Vienna, Austria

email: patrick.sturm@technikum-wien.at
web: http://npsolvers.wootstube.de/

Abstract. This document features a short introduction of the NP.Solvers team and the basic architecture of the NP.Solvers software, whose development has been going on for just under one year.

1 Introduction

Due to technical improvements occurring the field of engineering and manufacturing, there is a steady increase of application areas for robots. The NP.Solvers Virtual Robots team of the University of Applied Sciences Technikum Wien was founded in October 2007. The foremost objective was the creation of a working robot management system enabling our team to compete in international contests, such as the Robocup. When participating in the German Open 2008 competition, the NP.Solvers team modified the existing sourcecode of the MrCS (Multirobot Control System), developed by the STEEL team, a cooperation of Carnegie Mellon University, Pittsburgh PA and University of Pittsburgh, Pittsburgh PA and the winner of 2007 Robocup in Atlanta. The aim after the competition was to implement a completly new system, following a strictly modular approach that could be ported to real robots by replacing the USARsim specific API. In the beginning the team consisted of two members only but grew steadily. Currently two students are involved in the NP.Solvers team.

2 Architecture

2.1 Overview

The general requirements on the architecture:

- simplicity
  One goal was to write maintainable, well documented and understandable sourcecode to provide successor developers with a programmer friendly environment, which allows fast understanding, hacking and further development.
- flexibility
  The flexibility of the architecture is essential for reusability as well as portability. Reusability was one of the most influential aspects during the software planning process.

2.2 System Architecture

The core of the software is an abstract framework providing a basic control and datamanagement facilities. It consists of five core modules that allow datamining, caching, robot control, messaging and networking. So called applications communicate with the software's core by using this abstract framework, hence enabling applications to run on other implementations than the USARsim specific one.
Robots are represented by agents; the so called RobotAgent object possesses a set of all modules meaning that all robots have separate mechanisms and operate accordingly.

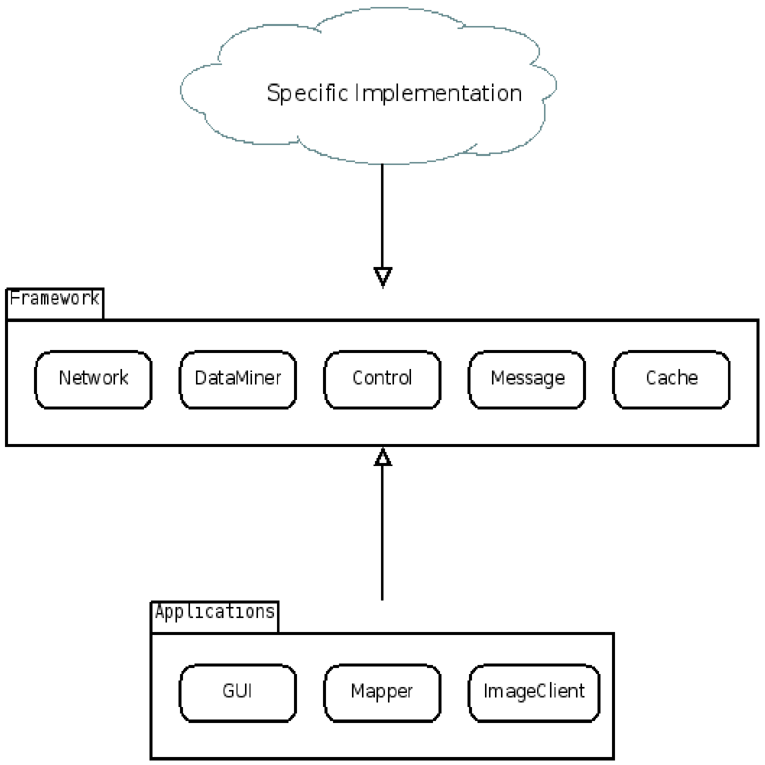

Fig. 1. Illustration: architecture.

2.3 Modules

Modules provide the core functionality of the robots such as controlling the robot and robot communication. As mentioned before, modules are completely interchangeable, as long as they follow the standards dictated by the framework. Each module was designed for one purpose only; strict separation of functionality enhances code readability and understanding. Modules are merely “primitive” wrappers that are not capable of making decisions and coordinating their actions. Note the following scenario:
The basestation commands a robot to move to a certain position. As soon as the command is received by the robot’s networking module, the data gets redirected to an arbitrary application handling such requests. The application registered with the networking module beforehand (observer) to receive networking data. The application itself knows how to handle drive requests and uses the control module to commence the driving procedure.

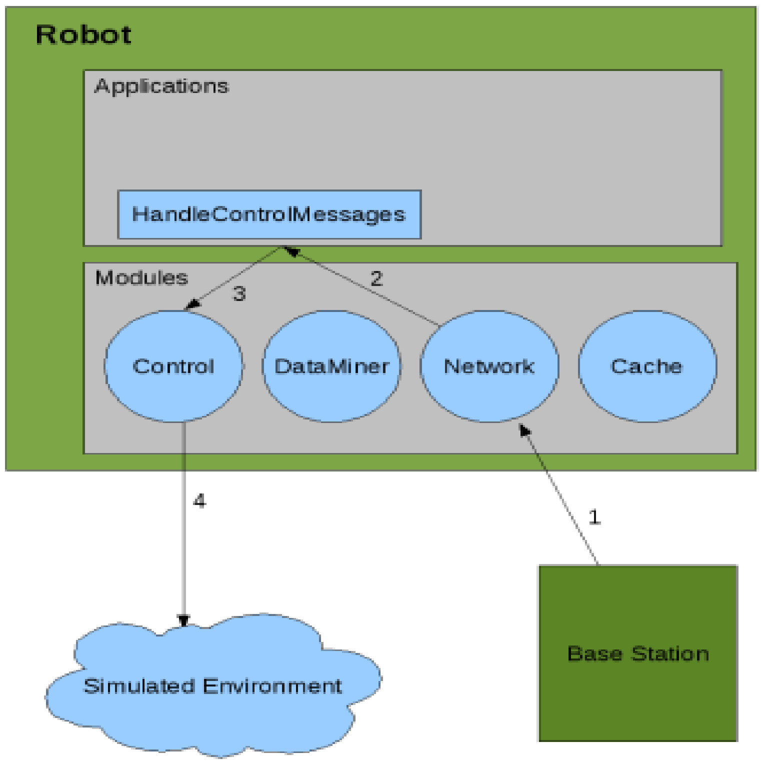

Fig. 2. Illustration: module application interaction.

A brief introduction of USARsim specific module implementation:

- Dataminer
  The dataminer is responsible for data exchange between the USARsim framework and the agent's applications. If USARsim provides any form of sensor related data, the data will be picked up by the dataminer and redirected to any application that registered with the dataminer's observer. If an application intends to communicate with USARsim, it needs to format the message using the provided method and command the dataminer to process the communication.
- Network
  The network module handles WSS related matters and operates similar to the dataminer. All WSS communication will be processed by the network module, applications that wish to engage in WSS communication need to register with the network module.
- Cache
  The cache is the exchange hub for applications. If an application wishes to profit from another application's knowledge, it does not request the information from the other application directly but will access the cache and retrieve the data. For instance: The operator requires an up to date map in order to control the robot properly. The mapper, that holds a recent image of the map deposits the map within the cache. The GUI than access the cache, retrieves the map and is able to display it to the operator.
- Control
  The control, as the name states handles control requests.

2.4 Applications

Applications are build on top of the abstract framework, therefore ensuring reusability. The current development state includes three applications, the GUI, a mapping application and an image client corresponding to the image server provided by the USARsim project. The communication in between applications is organized by a caching class, which is part of each agent and can be access by all applications equally. Essentially this means, that applications may communicate in an indirect manner but will never engange in a direct dialouge, thus preventing unexpected behaviour and providing data coordination.

2.5 Configuration

Robots are defined in XML files located in a directory dedicated for configuration files. The Graphical user interface is aware of all available configuration files and provides an easy to use drop down menu.

2.6 Implementation

The software was implemented in Java, following standard Sun coding conventions. The use of utility classes was minimized in order to produce reusable and pluggable code. The data exchange including USAR and WSS messages is handled by centralized modules; USAR messages will be processed by the DataMiner, while WSS messages are directly digested by the networking module. Communication with those parts will always involve the dedicated management facility and will never happen directly.
The mapper implementation was taken from the Steel Team code and modified to work as an application on top of the abstract framework. The configuration of the robots is saved using a XML format.

2.7 Future goals

Autonomic robot control Implementation of algorithms, which allow autonomous mapping. The operator should only interfere in extraordinary situations.

Mapping More detail especially concerning metadata such as hazardous environments, floor conditions and other environmental condition.

New applications Design an own mapper to replace the current version of the Steel Team mapper