Common Platform Architecture

A Simple and Clean Architecture for Participation in SPL and Simulation 3D

Heinrich Mellmann, Yuan Xu, and Thomas Krause

Institut für Informatik, LFG Künstliche Intelligenz, Humboldt-Universität zu Berlin, Rudower Chaussee 25, 12489 Berlin, Germany

http://www.naoth.de
nao-team@informatik.hu-berlin.de

1 Introduction

An appropriate architecture (i.e., framework) is the base of each successful heterogeneous software project. It enables a group of developers to work at the same project and to organize their solutions. From this point of view, the artificial intelligence and/or robotics related research projects are usually more complicated, since the actual result of the project is often not clear. In particular, a strong organization of the software is necessary if the project is involved in education.

At present, there are several different leagues in RoboCup, which focus on different subtasks. However, there are only few solutions working simultaneously in different leagues. A framework, which is able to run on different platforms can help to achieve this. Obviously, there is no perfect framework which could satisfy all the needs of the developers. In this project we propose a modular software architecture designed to implement an autonomous agent. In particular, it is used to develop software which is used simultaneously at several platforms, e.g., humanoid robot, simulated agent. One of the main aspects considered in our design is a strong code modularization which allows for re-usability, transparency and easily testing. Other important aspects are real-time applicability and simple usage.

As a former member of the German Team our team has more than 10 years history in RoboCup and thus also large experience in building of software frameworks for autonomous robots. At the moment, our team — Nao Team Humboldt participates in both, SPL and S3L with the common core of our program [1]. Therewith, we want to foster the cooperation between the two leagues and to improve both of them. In the S3L, we won the German Open and the AUTCup competitions and achieved the 2. place at the RoboCup World Championship 2010 in Singapore; in the SPL, we won the Greek RoboCup Championship SETN 2010 and achieved 2. place at Mediterranean Open in Rome 2010, and 4. place GermanOpen Magdeburg 2010.

2 NaoTH Architecture

In this project we will design end implement a C++ software architecture which is able to run on different robot platforms, in particular on the robot Nao and with the simulator SimSprak. The framework will allow to exchange the platform without changing the actual code of the agent. Thus, it will be transparent for the agent, e.g., the whole code can be run with recorded data as if it would run of the real robot. In more detail, it will consist of the following parts:

communication - between the agents and with monitoring or debugging software;

high-level runtime debug - generic layer based on the communication, which allows for an easy implementation of custom concepts for debugging during the runtime, e.g., monitor, change or plot values, monitor the execution time of code parts, switch on/off parts of code;

platform interface - a generic interface allowing to exchange the platform implementation without touching the actual core part of the code. We plan to implement platforms for SimSpark, Nao, Webots, Webcam and a LogSimulator. However, further custom platforms can be easily implemented;

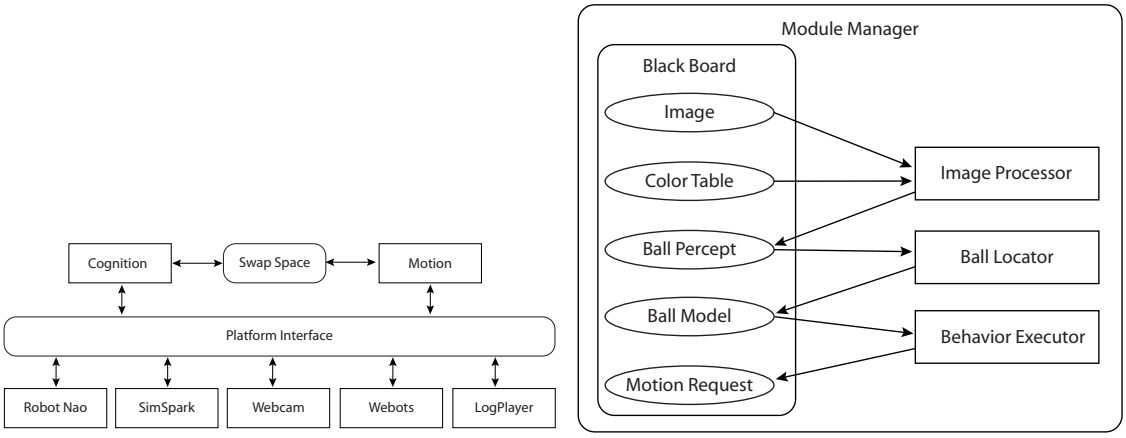

Fig. 1. Schematical illustration of the *platform interface* (left) and the *module architecture* (right).

module architecture - a blackboard architecture for the organization of the core part of the code;
logging - record the state of the agent, e.g., images, and replay them with using the LogSimulator;
testing infrastructure - infrastructure which allows to implement and run automated tests;

The whole architecture will be structured in a way, that every of this components can be easily replaced by a custom one, e.g., in our code we use *Protobuf* for serialization in our communication, but it is also possible to use any other solution.

The main design of the architecture is already finished and the first implementation is working well, as already mentioned. However, there is still a lot of work to be done to create a release version. The current state of our architecture is presented in [2]. We also published a *Simple Soccer Agent* — an implementation of an soccer agent based on our framework. It can be downloaded from our homepage [3]. In particular this agent can be run in S3D and SPL. At the current state, this program has two aims: to demonstrate how our framework works, but also to enable new teams easily to access the Simulation League. The current version of *Simple Soccer Agent* and the framework can be downloaded from our homepage [3]. Fig. 1 illustrates schematically the current state of the *platform interface* (left) and the *module architecture* (right).

3 Impact on the RoboCup Community

With our efforts in the proposed project, we hope to foster the cooperation between the two leagues and to improve both of them. There are mainly two obvious benefits which can be expected in a short time:

1. New teams can easy enter the simulation league and start developing their own teams;
2. The transfer of solutions between SPL and S3D becomes very easy;

On the long time scale we hope to contribute to the development of a *standard* software architecture for humanoid robots, which could be used in all RoboCup leagues.

References

1. Xu, Y., Mellmann, H., Burkhard, H.D.: An approach to close the gap between simulation and real robots. In Ando, Balakirska, Reggiani, von Stryk, eds.: 2nd International Conference on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR), Darmstadt (November 2010)
2. Mellmann, H., Xu, Y., Krause, T., Holzhauer, F.: Naoth software architecture for an autonomous agent. In: International Workshop on Standards and Common Platforms for Robotics (SCPR 2010), Darmstadt (November 2010)
3. Humboldt, N.T.: NaoTH web site. (2010) http://www.naoth.de.