Emergent Architecture Design

Introduction

In this document we will go over the details of how the system is going to be built and how our group will design the software architecture of the connector part of our context project.

Design goals

Throughout the project, the contributors will keep these design goals in mind when creating code for our project.

Extensibility

Keeping in mind that the Tygron Engine is not final and may be extended, we will design our Connector in such a way that features can easily be added or modified if the Tygron Engine changes.

Performance

The Connector should have low latency, so that agents can ask for environment information and be able to process this information without the interface being the bottleneck.

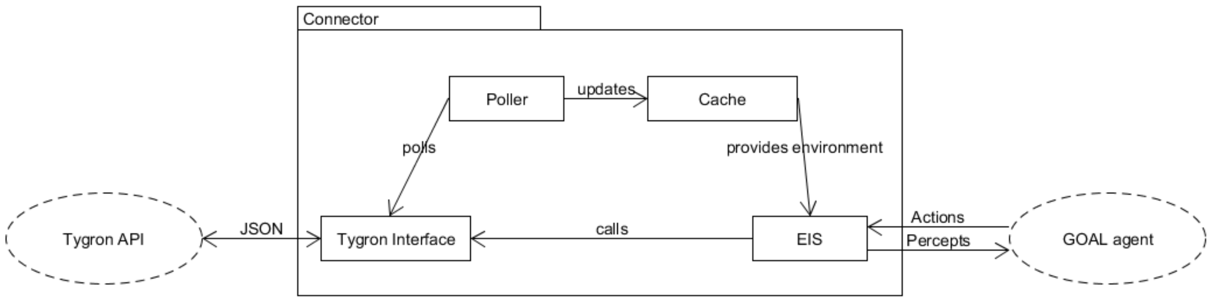
Code quality

Our connector’s code will be of high-quality, which means that code will be dynamically and statically tested.To ensure high quality we will be using JUnit tests, running CheckStyle, PMD, and FindBugs static testing and be using Travis integration. We will also be working on a pull-request based system where every addition or change to the code will be manually reviewed and discussed before being allowed to be pulled to the master branch.

Software architecture views

Subsystem decomposition

This part describes our Connector and how it is divided into subsystems.



Tygron Interface

The Tygron Interface is a subsystem that will communicate with Tygron’s REST endpoint. This interface’s responsibility is to translate functions to REST requests, and to translate REST responses back to Java objects.

Cache

The Cache is a buffer between the EIS and the Tygron Interface. It’s responsibility is to keep a copy of data from the Tygron Interface which can be updated. GOAL agents will be able to access this data via the EIS. The Cache is necessary because otherwise the GOAL agent will be have to wait for the duration of the API request.

EIS

We will also develop an interface according to the Environment Interface Standard. This interface is what provides GOAL agents with percepts which will provide GOAL agents with the required information to perform and exposes functions that the agent can call.

Poller

It’s job is to poll our JSON interface with some interval, and the results of the polling is used to update the Cache.

Hardware/software mapping

Our software will communicate over the internet with Tygron’s server. This communication will be layered on HTTP, using the REST protocol.

Persistent data management

Our connector does not have the responsibility of storing persistent data, so it does not have external files or databases.

Concurrency

Our cache is a shared resource. It is updated by one subsystem and read by another, therefore we are not concerned about deadlocks.

Glossary

Cache
A collection of data which duplicates other data

EIS
Environment Interface Standard, a standard which facilitates connecting agents to environments

GOAL
Goal based agent programming language, used for programming intelligent agents

JSON
JavaScript Object Notation, a data representation format

Percept
A chunk of data that a GOAL agent receives from it’s environment

Polling
Sampling an external service

REST
Representational State Transfer, a software architecture which is used for creating web services