# Traffic Light Control Simulation for Various Simulation Environments

*Behnecke D [1], Fischer M [1], Farkas B [2], Assmann D [1], Berekovic M [2], Köster F [1]*
*[1] German Aerospace Center (DLR), Institute of Transportation Systems, e-mail: danny.behnecke@dlr.de*
*[2] Technische Universität Braunschweig, Abteilung technische Informatik, E.I.S*

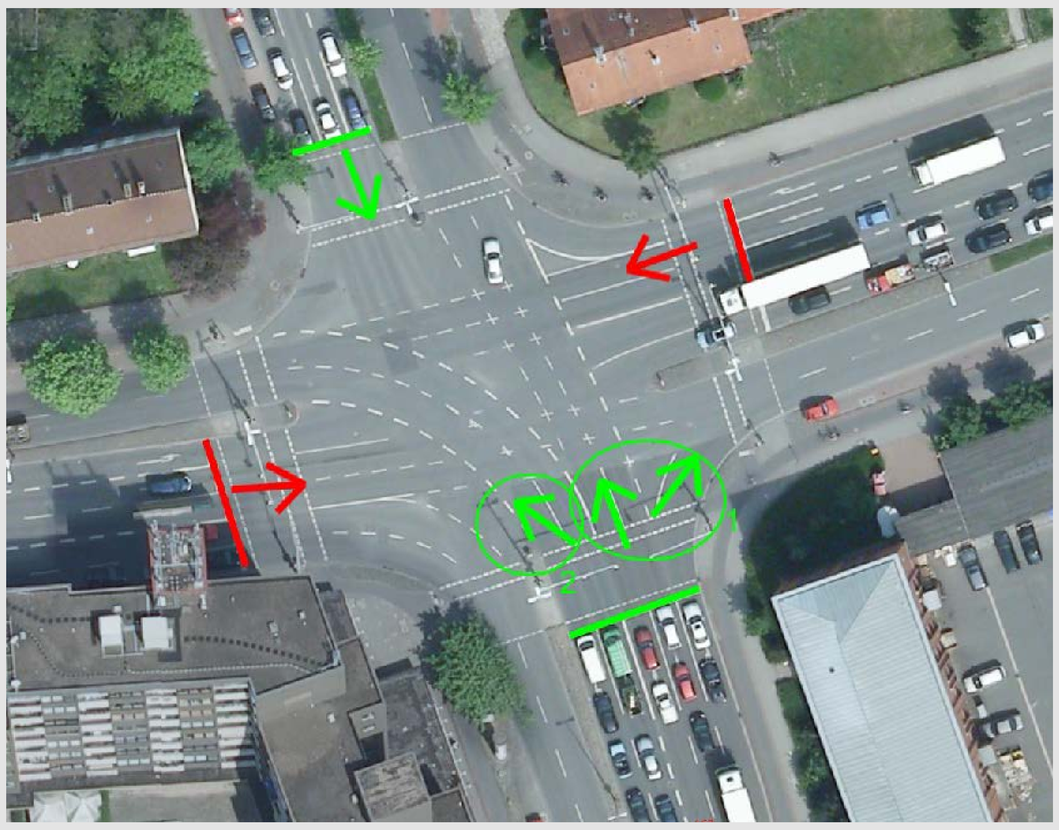

Figure 1: Real intersection – top view

## From reality to software

Figure 1 shows a typical intersection setup in reality. It also shows an example of a phase configuration. Phase control can be implemented arbitrarily complex and is a historically grown piece of technology. To satisfy the hereby existing conditions, simulation software have many different ways of implementing these structures. But this is not done uniform across different simulation environments. Figure 2 shows the representation of an intersection similar to Figure 1, created for the microscopic traffic simulation software SUMO*. SUMO for example controls traffic lights by the paths that will be taken across the intersection (as shown in Figure 2), while other software tools keeps a lane bound control within its structure (e.g. VTD**). To improve this we propose an abstract and more general layer in order to create one general interface for traffic light control. The basic architecture is shown in Figure 3.

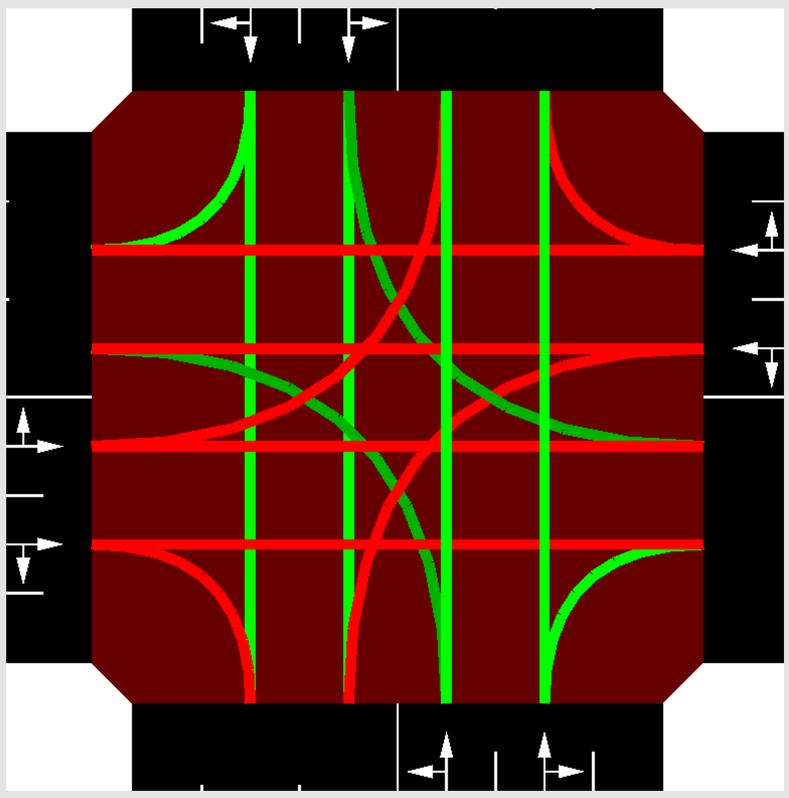

Figure 2: Intersection in SUMO

## Introduction

The ongoing digitalization of life would not be imaginable without software. Every aspect is now or will soon be software supported or already controlled. But the more complex a piece of software becomes, or the more software is used, the greater the probability of clutter and bugs becomes. In this environment, clear, direct and unraveled software designs are highly relevant. This becomes even more important when operating in domains that harbor possible dangers like it is the case in traffic control. With intelligent infrastructure design arises the need for safe, secure and reliable software. Key components here are the traffic lights and their control.

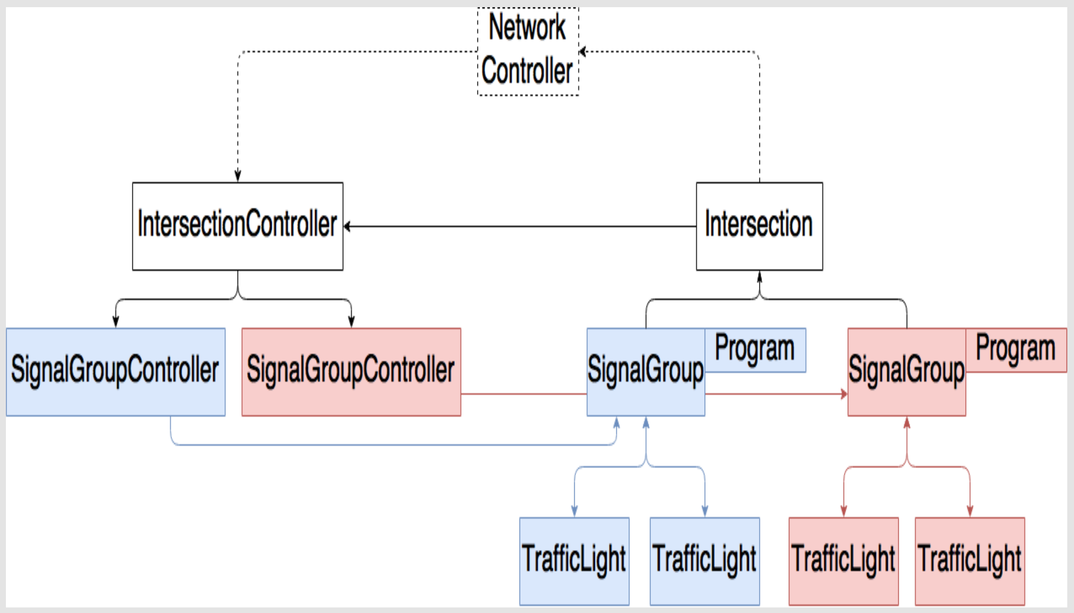


Figure 3: Software architecture design overview

## Design

It is a mixture of the Observer and the Model-View-Controller Pattern. The arrows indicate both control and notify structures as well as membership of the different modules among each other. This design allows the user to simulate every control scheme, from normal fixed time controls to more complicated, adaptive ones. Also it could improve interoperability between simulation frameworks, as every framework has its strengths and weaknesses in the context of traffic simulation.

Furthermore it defines a structure that can be tested in a straight-forward manner. When talking about distributed systems, testing and ensuring quality is a major task. This design presents a very modular structure and therefore a very cohesive architecture. Its modules are connected via narrow interfaces, partly given by the Observer pattern and partly domain specific. These interfaces reduce the test-cases to be tested and, again, unifies them. This would create transparency between traffic control simulation systems when it comes to traffic light controlling.

* http://sumo.dlr.de
** https://vires.com/vtd-vires-virtual-test-drive/



Deutsches Zentrum für Luft- und Raumfahrt