CARMEN: A software structure for control and monitoring of flexible automation.

Project Introduction

The project is associated with the overall project CARMEN that in general is concerned with reducing the setup time for automation systems.

The project will more specifically be related to the software design and overall system design of an infrastructure on which an action library can be applied.

Key subjects for the project

- Infrastructure
- System monitoring and surveillance
- Error recovery
- Action selection and chain checking
- Concurrent programming

System monitoring and error detection

One of the key focus points besides the infrastructure in general, will be on how to integrate system monitoring and surveillance natively into an infrastructure. As robot and automation systems are moving away from deterministic actions and applications and starts becoming more generic the likelihood of failures and errors increases.

The problems concerning errors are three fold
- Detecting the error
- Reacting on the error
- Passing of the error

Combining the touching relations with the semantic event chains and other sensor input to produce a system that can constantly monitor if the outcome of an action is as expected and integrating this into the infrastructure could provide valuable information.

Model driven software development

- Programming in a tailored language

A key advantage of developing applications in modern day programming languages is their ability to give meaningful and helpful errors massages. This allows creators to create more complicated programs in shorter time. Theses messages however are only concerned with the programming and not coupled to the physical world. Providing developers with the errors detected by the system in a useful format would hopefully help the programmers and developers reduce they setup time of automation systems even more.

This system should be able to provide the users with selections of the proper actions. This system should be able to provide the users with selections of the proper actions. This system becomes relevant when a programming containing a sequence of actions is made. It is the desired to be able to use pre and post conditions of the actions to suggest actions based on whether they are feasible in a given context.

Example of DSL