The Tekkotsu State Machine Composer

Authors: Albert Berk Toledo, Roger Smith II, Hampton University

Advisor: Professor David S. Touretzky, Carnegie Mellon

Abstract

The Tekkotsu State Machine Composer is a graphical editing tool for composing state machines in the Tekkotsu robot programming framework. Tekkotsu is widely used in undergraduate schools for teaching robotics. It provides a hierarchical, event based state machine language built on top of C++. The State Machine Composer consists of graphical components, textual components, and wizards. A graphical component lets users create state machines by dragging and dropping shapes representing nodes and transitions. Users are also allowed to edit the code directly to manipulate advanced parts of the Tekkotsu code such as custom method bodies. Wizards facilitate the creation of specialized state nodes, such as MapBuilder nodes used for robot vision. We have built Java classes that represent the components of a Tekkotsu program, and methods to translate these components into compilable Tekkotsu code that can successfully run on robots. Currently users can create simple state machines using the graphical tool and generate Tekkotsu code for their state machines. The graphical interface of the program is built on top of the Eclipse Rich Client Platform and Graphical Editing Framework. The program will have a novice mode in which the more advanced features are suppressed, and an expert mode that makes the full power of the framework available. We expect the tool to be adopted in robotics classes and competitions where Tekkotsu is used. Member institutions of the ARTSI Alliance will be the first schools to use the tool in their educational activities.

Description

Graphical display and editing of state machines is common in robot programming frameworks. Examples include LEGO Mindstorms (NXT-G, based on LabView), Nao robots (Choregraph), ROS (Smach), and Gostai URBI. Graphical interfaces make it possible for even non-programmers to create simple robot behaviors, as LEGO has demonstrated.

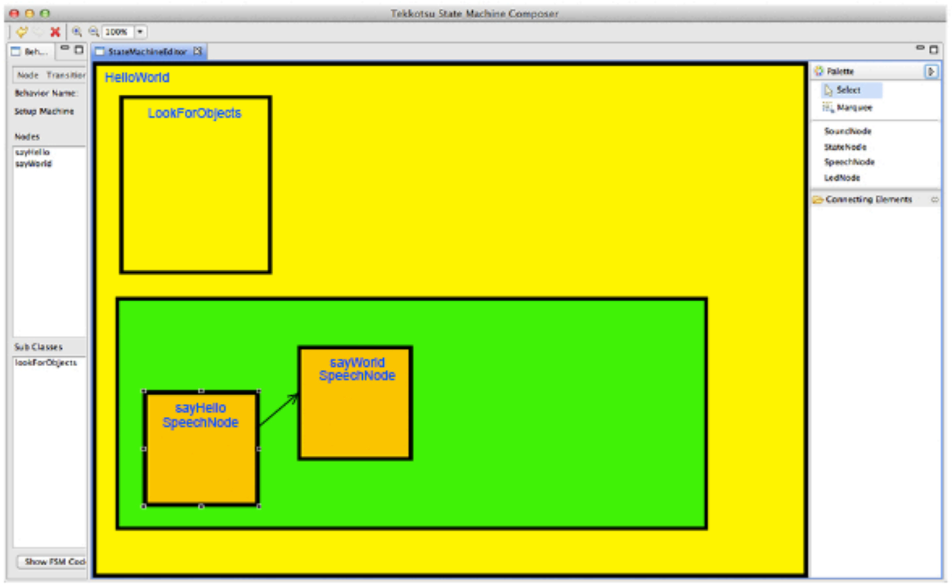
Tekkotsu is an open source robot programming framework built on top of C++. It includes a hierarchical concurrent state machine formalism with a special shorthand notation that is translated into pure C++ code by a Perl script. The Tekkotsu State Machine Composer is a new graphical editing tool for composing state machines, built using the Eclipse Rich Client Platform. The Eclipse Graphical Editing Framework is used for the graphical component.

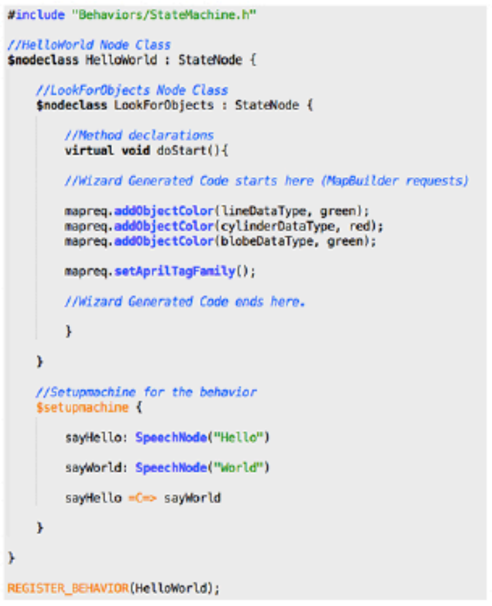
Node classes in Tekkotsu are more complex than in some other state machine formalisms, as they can contain not only nested subclasses but also C++ methods and variables. In the Tekkotsu State Machine Composer, all these components are visualized as editable graphics. The Composer also allows users to edit more complex parts of a Tekkotsu behavior such as MapBuilder requests used for robot vision. Wizards and integrated code editors are used for advanced modification of a behavior.

The Tekkotsu State Machine Composer is tailored for the specific needs of Tekkotsu programming. Targeting Tekkotsu programmers with a wide range of skill levels, it will have a novice and an expert mode to provide the right interface for each user. The expert mode allows the tool to expose the full power of Tekkotsu for advanced Tekkotsu programmers, while the novice mode suppresses complex components.

We have built data structures in Java that represent the components of a Tekkotsu behavior. The components represented include NodeClass, TransitionClass, NodeInstance, TransitionInstance, SetupMachine, Method, Variable, ConstructorCall, Parameter etc. We have also built a Java class with appropriate methods to parse the information represented in these structures and convert them to compilable Tekkotsu code.

The Tekkotsu State Machine Composer will be used in robotics classes and competitions. The member institutions of the ARTSI (Advancing Robotics Technology for Societal Impact) Alliance will be among the first to use the tool. Feedback from these users will help us improve the quality of the tool in the future.




```
#include "Behaviors/StateMachine.h"

//HelloWorld Node Class
$nodeclass HelloWorld : StateNode {

    //LookForObjects Node Class
    $nodeclass LookForObjects : StateNode {

        //Method declarations
        virtual void doStart(){

        //Wizard Generated Code starts here (MapBuilder requests)

        mapreq.addObjectColor(lineDataType, green);
        mapreq.addObjectColor(cylinderDataType, red);
        mapreq.addObjectColor(blobeDataType, green);

        mapreq.setAprilTagFamily();

        //Wizard Generated Code ends here.

        }

    }

    //Setupmachine for the behavior
    $setupmachine {

        sayHello: SpeechNode("Hello")

        sayWorld: SpeechNode("World")

        sayHello =C=> sayWorld

    }

}

REGISTER_BEHAVIOR(HelloWorld);
```