Into the ROS

Advanced ROS Network Introspection

Praxis der Softwarentwicklung
Summerterm 2014

S o f t w a r e d e s i g n

Client

KIT - Karlsruher Institut für Technologie
Fakultät für Informatik
Institut für Anthropromatik und Robotik (IAR)
Intelligente Prozessautomation und Robotik (IPR)

Advisor: Andreas Bihlmaier
andreas.bihlmaier@gmx.net

Contributors

Name	E-Mail-address
Alex Weber	alex.weber3@gmx.net
Matthias Hadlich	matthias.hadlich@student.kit.edu
Matthias Klatte	matthias.klatte@go4more.de
Micha Wetzel	micha.wetzel@student.kit.edu
Sebastian Kneipp	sebastian.kneipp@gmx.net

Karlsruhe, 07.06.2014

Inhaltsverzeichnis

1 Composition

1.1 Architecture

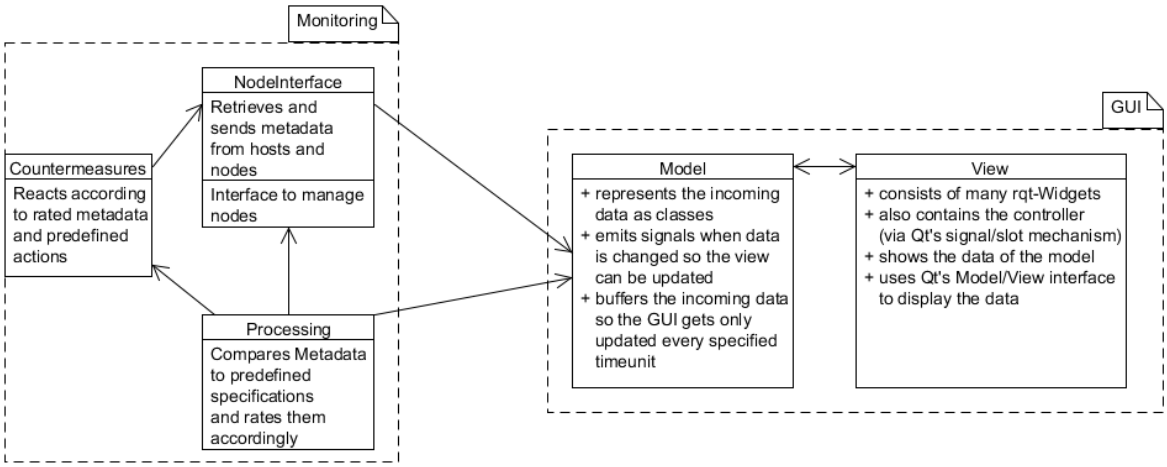

Abbildung 1.1: architecture

Figure 1.1 shows the general architecture of our software. It is divided into two parts, one for the graphical user interface and one for the monitoring aspect. The right part depicts the GUI. It is designed using the MVC architecture, consisting of the usual three elements: model, view and controller. It will handle user-interaction. The left part depicts the monitoring aspect. It consists of three elements : NodeInterface, Countermeasure and Processing. It will take care of collecting metadata, processing it and taking appropriate action in case of an error.

1.1.1 Monitoring

NodeInterface
- Retrieves and sends metadata from hosts and nodes
- Interface to manage nodes

Processing Compares Metadata to predefined specifications and rates the accordingly

Countermeasures Reacts according to rated metadata and predefined actions

1.1.2 GUI

Model
- Represents the incoming data as classes
- Emits signals when data is changed so the view can be updated

- Buffers the incoming data so the GUI gets only updated every specified timeunit

View

- Consists of many rqt-Widgets
- Also contains the controller (via Qt's signal/slot mechanism)
- Shows the data of the model
- Uses Qt's Model/View interface to display the data

2 Classes Description

2.1 Processing

2.1.1 MonitoringNode

Main Class wrapping the processing functionality.

2.1.2 Metadata

Wraps metadata of exactly one host or node, a topic or a node-topic-combination

2.1.3 MetadataTuple

Contains the name of a metadata field and an object which can be a monitoring point or the bounds as a tuple.

2.1.4 MetaDataStorage

Saves recieved metadata packages for a given period of time and can provide them on request.

2.1.5 Specification

Wraps specification fields. Can contain multiple MetadataTuple objects from exactly one host or node

2.1.6 SpecificationHandler

Loads the specifications from the parameter server and compares them to the actual metadata.

2.1.7 ComparisonResult

Wraps the result of the comparison between the actual metadata and the specificaton.

2.2 NodesInterface

2.2.1 HostStastistic

Singleton per host which contains statistics about the host and nodes running on the it. Handles request regarding node management.

2.2.2 NodeManager

Is able to stop or restart nodes.

2.3 Countermeasures

2.3.1 CountermeasureNode

Handles incoming information about malfunctioning nodes and reacts according to defined countermeasures.

2.4 GUI

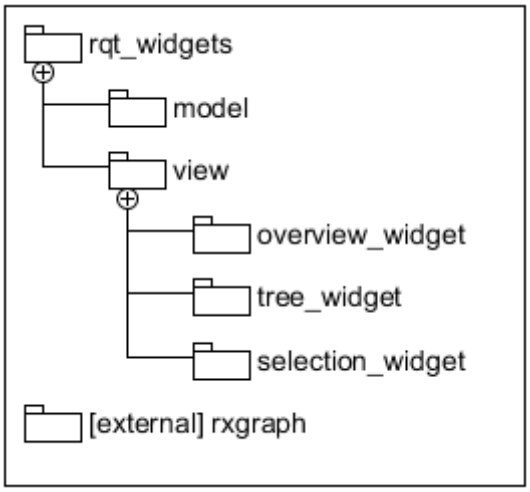

Abbildung 2.1: The package structure of the GUI

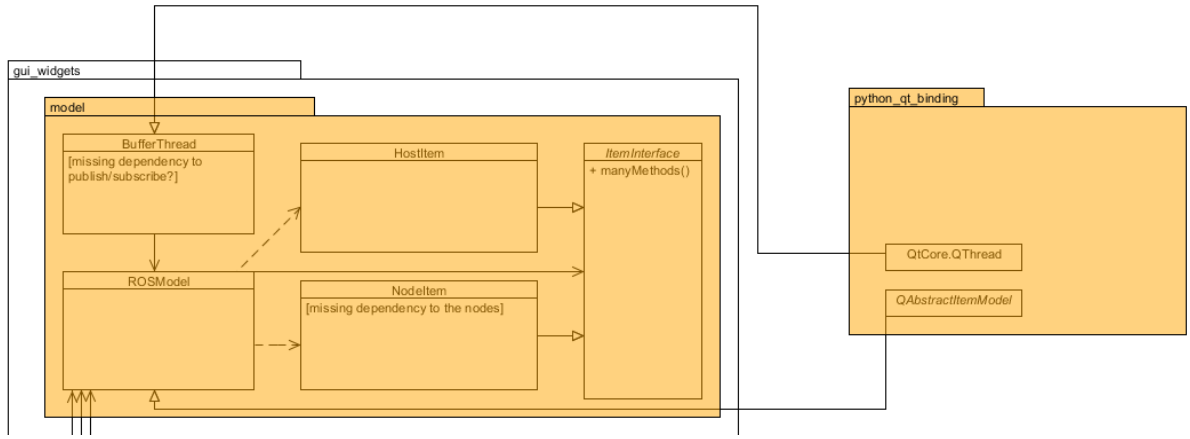

Abbildung 2.2: The model class diagram

2.4.1 Model

BufferThread

This thrad should buffer the incoming data and regulary update the model and hence also the model.

ROSModel

Represents the data as a QtModel. This enables automated updates of the View.

ItemInterface

Provides a unified interface to access the items of a model.

HostItem

A HostItem represents a host with all its data.

NodeItem

A NodeItem represents a node with all of its data. It also has a interface to start/stop/restart nodes.

2.4.2 View

OverviewPlugin

TreePlugin

SelectionPlugin

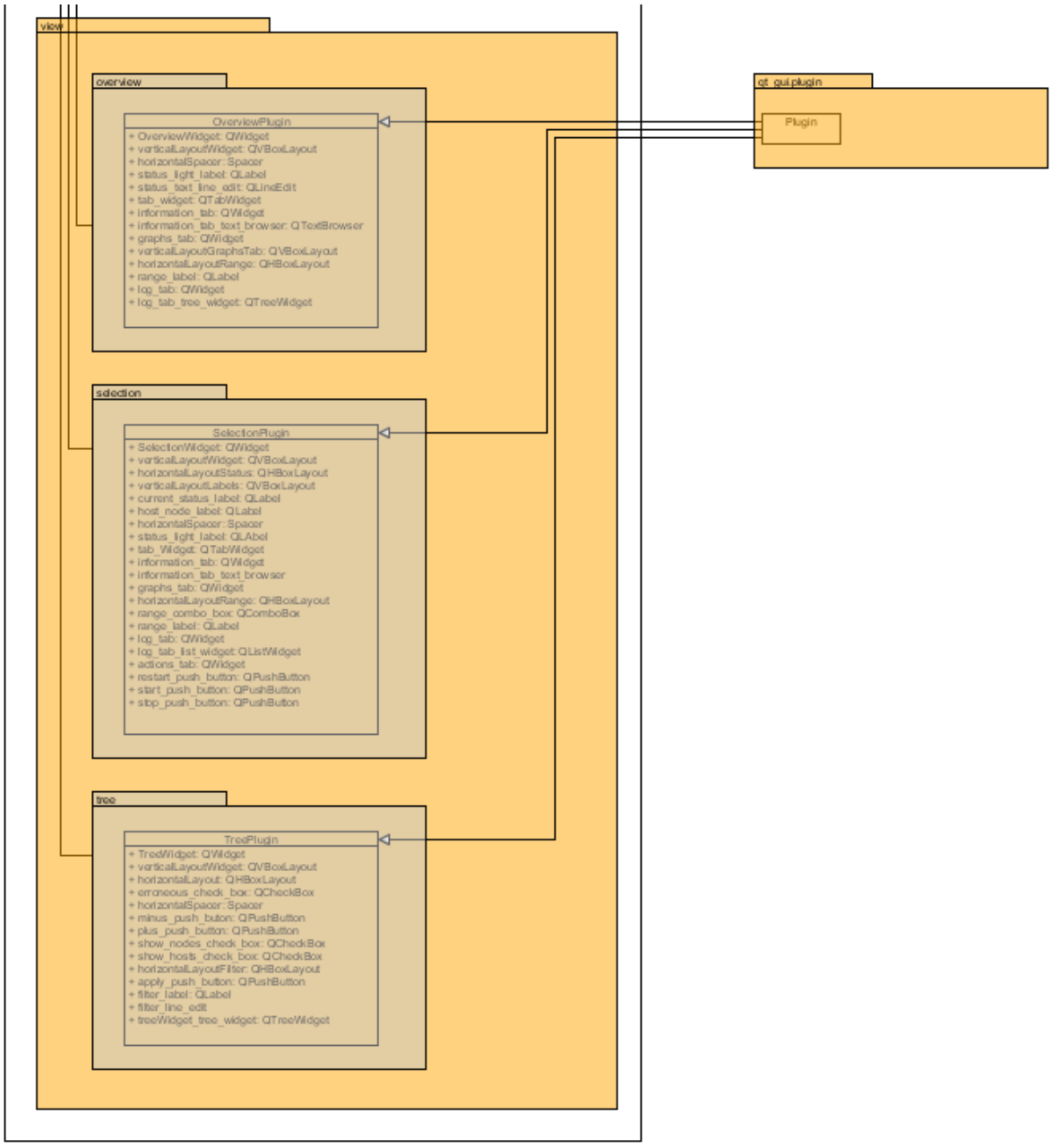

Abbildung 2.3: The view class diagram