

INTO THE ROS

ADVANCED ROS NETWORK INTROSPECTION

Praxis der Softwareentwicklung
Sommerterm 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 Anthropomatik 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	3
1.1	Architecture	3
1.1.1	Monitoring	3
1.1.2	GUI	4
2	Classes Description	5
2.1	Processing	5
2.1.1	MonitoringNode	5
2.1.2	Metadata	5
2.1.3	MetadataTuple	5
2.1.4	MetaDataStorage	5
2.1.5	Specification	5
2.1.6	SpecificationHandler	5
2.1.7	ComparisonResult	5
2.2	NodesInterface	6
2.2.1	HostStatistic	6
2.2.2	NodeManager	6
2.3	Countermeasures	6
2.3.1	ConstraintHandler	6
2.3.2	Constraint	7
2.3.3	ConstraintItem	7
2.3.4	ConstraintLeaf	8
2.3.5	ConstraintAnd	8
2.3.6	ConstraintOr	9
2.3.7	ConstraintNot	9
2.3.8	Enum Outcome	9
2.3.9	CountermeasureNode	10
2.4	GUI	10
2.4.1	Model	11
2.4.2	View	12

1 Composition

1.1 Architecture

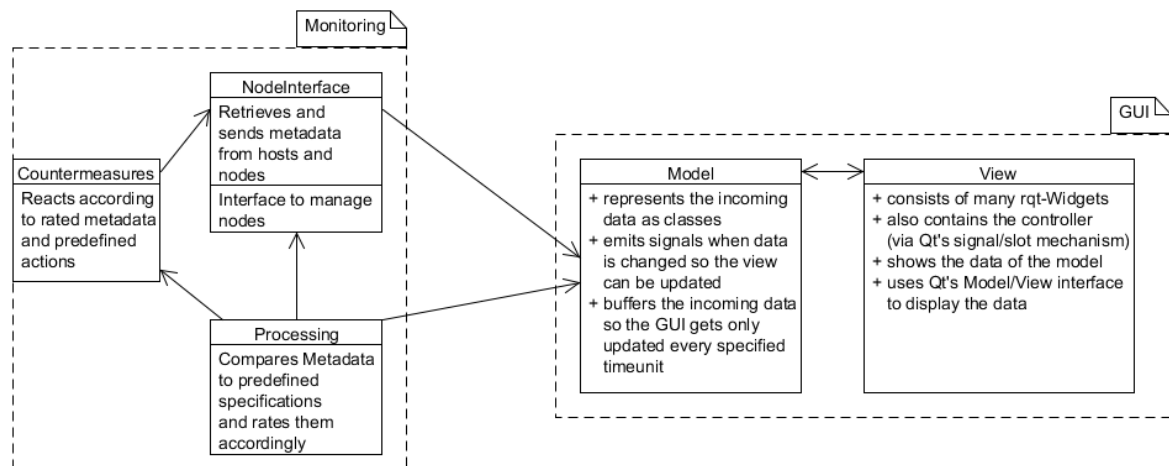


Abbildung 1.1: architecture

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 specifcaton.

2.2 NodesInterface

2.2.1 HostStatistic

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 ConstraintHandler

Attributes

- private Constraint constraint[]
contains a list of all constraints
- private RatedStatistic rated_statistic
contains all incoming rated statistic
- private int global_constraint_level
only constraints with an constraint_level \leq current_constraint level get evaluated

Methods

- public addConstraint(Constraint)
adds an constraint to this list
- public setStatistic(RatedStatistic)
sets the Statistic to use. Should only be needed on initialisation
- public evaluateConstraints()
evaluates every constraint
- public executeReactions()
checks if there are any new reactions to do and executes them.
- private reactToConstraint(Constraint)
executes an single Reaction and updates the attributes of the Constraint

2.3.2 Constraint

TODO: Description

Attributes

- private ConstraintItem constraint
first constraint in the chain of ConstraintItems.
- private int true_since
epoch time in milliseconds since the constraint is true, if the constraint is not true it is 0
- private Reaction planned_reaction
the reaction that should be executed if the constraint has been true longer than min_reaction_interval milliseconds
- private int min_reaction_interval
the minimum time needed in ms that the constraint needs to be true to execute the planned_reaction
- private int last_reaction
contains the epoch time in ms when the reaction corresponding to this constraint has been executed for the last time. is 0 if it has never been executed
- private int reaction_timeout
minimum duration in ms needed before an reaction can happen again
- private int constraint_level
this constraint only gets evaluated if the global constraint level is \geq constraint_level

Methods

- public evaluateConstraint(RatedStatistic)
evaluates this constraint and sets the attributes according to the result of the evaluation

2.3.3 ConstraintItem

Abstract description of a Constraint, can be an logical operation on constraints or an actual constraint

Attributes

Methods

- public abstract boolean evaluateConstraint(RatedStatistic)
evaluates if this constraint, given the available RatedStatistic, is true.

2.3.4 ConstraintLeaf

Contains an actual constraint.

Attributes

- private String name
contains the name of the statistic data
- private Outcome outcome
contains the outcome needed for this constraint to be true
- private String seuid
contains the unique identifier of the corresponding StatisticEntity

Methods

- public abstract boolean evaluateConstraint(RatedStatistic)
returns true if this constrain is true for the RatedStatistic

2.3.5 ConstraintAnd

TODO: Description

Attributes

- private Constraint constraint[2]
contains the two constraints to be evaluated with a logical and

Methods

- public boolean evaluateConstraint(RatedStatistic)
returns true if the evaluation of both constains returns true

2.3.6 ConstraintOr

TODO: Description

Attributes

- private Constraint constraint[2]
contains the two constraints to be evaluated with an logical or

Methods

- public boolean evaluateConstraint(RatedStatistic)
returns true if the evaluation of at least one constraint returns true

2.3.7 ConstraintNot

TODO: Description

Attributes

- private constraint: Constraint
the constraint to be evaluated negated

Methods

- public boolean evaluateConstraint(RatedStatistic)
returns true if the evaluation of the constraint returns false

2.3.8 Enum Outcome

TODO: Description

Types

- high
data value is too high
- low
data value is too low
- unknown
data value is unknown
- out_of_bounds
data value is either too high or too low

2.3.9 CountermeasureNode

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

2.4 GUI

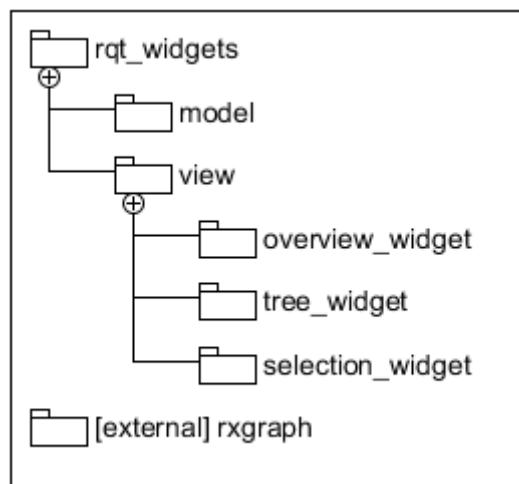


Abbildung 2.1: The package structure of the GUI

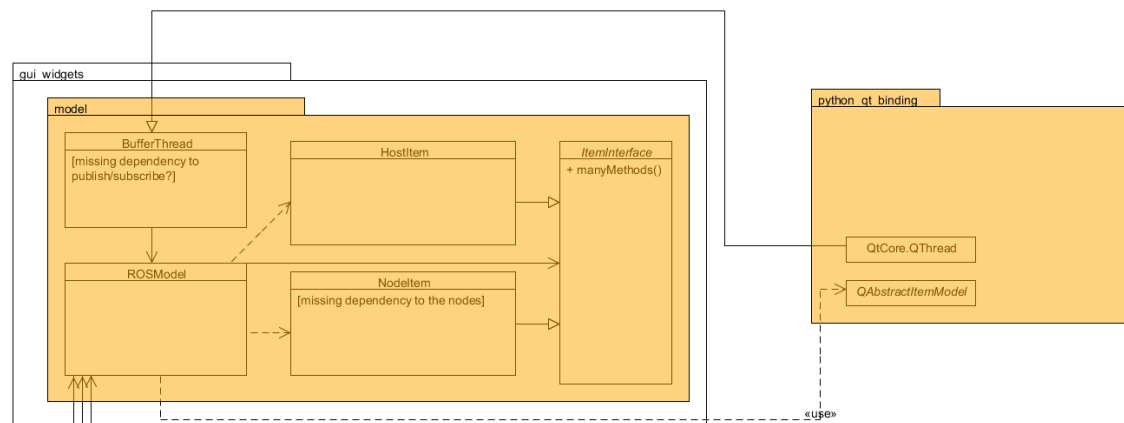


Abbildung 2.2: The model class diagram

2.4.1 Model

BufferThread

This thread should buffer the incoming data and regularly 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

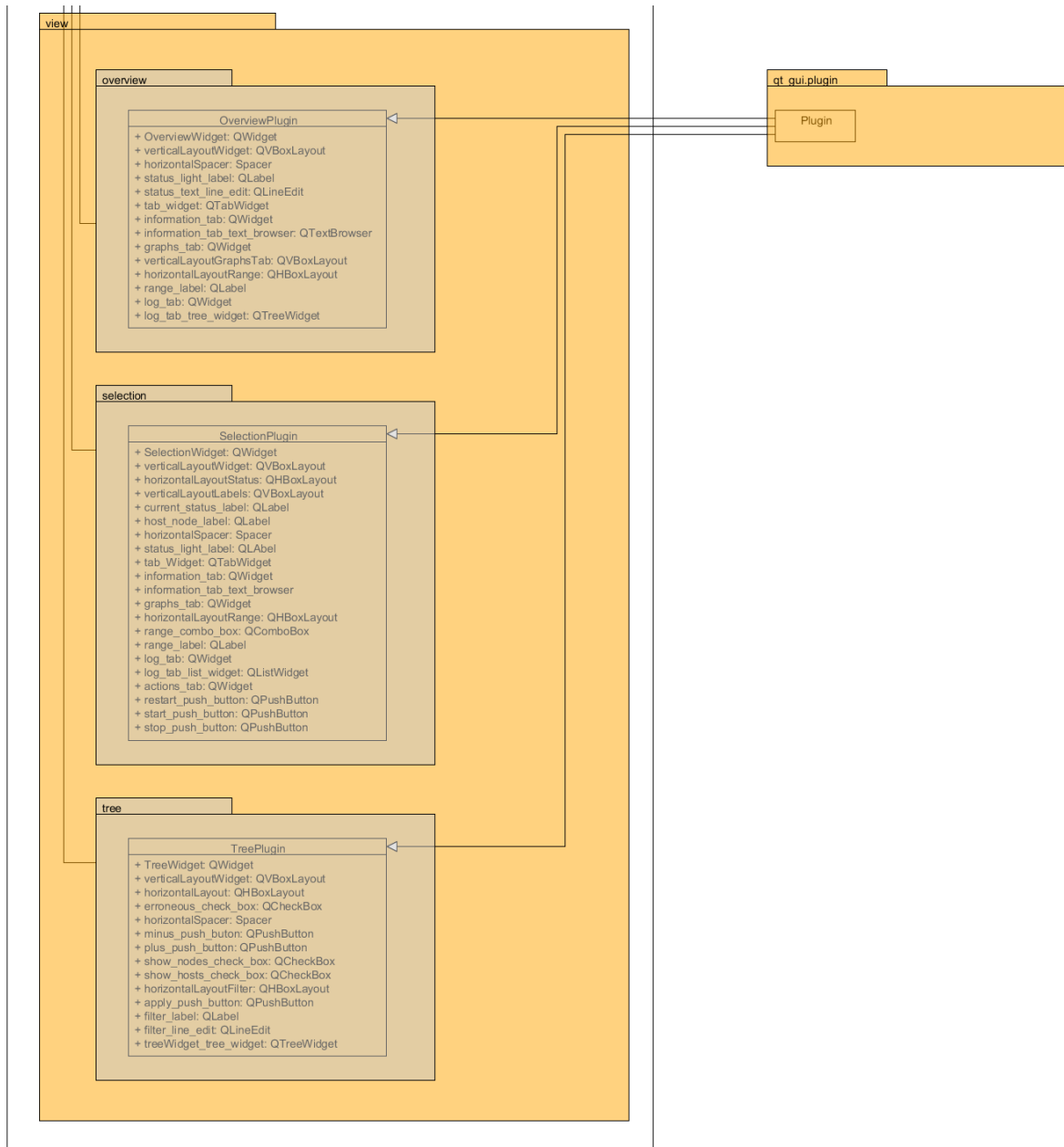


Abbildung 2.3: The view class diagram

OverviewPlugin

The class OverviewPlugin is the core of the graphical user interface, which contains most of the relevant information in a small and fancy area.

TreePlugin

TreePlugin is very simply and shows only the actual active hosts and nodes.

SelectionPlugin

A class which shows detailed information in a Tree-Layout about the currently selected host or node.