crystal_facet_uml user documentation

COLLABORATORS			
	TITLE : crystal_facet_uml user documentation		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Andreas Warnke	2019-03-13	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Introduction **1**

1.1 Goal . . . 1

1.2 Features . . . 1

1.3 Usage Overview . . . 2

2 Example Diagrams **2**

2.1 Feature List . . . 2

2.2 Example UML Behavioral Views . . . 5

2.3 Example UML Static Views . . . 8

2.4 Example SysML Views . . . 10

2.5 Other Examples . . . 12

3 GUI / Usage Manual **12**

3.1 Window Area Overview . . . 12

3.2 Tool Bar . . . 13

3.2.1 Create/Use DB . . . 13

3.2.2 Export . . . 13

3.2.3 New Window . . . 14

3.2.4 Navigate . . . 14

3.2.5 Edit . . . 14

3.2.6 Create . . . 14

3.2.7 Cut . . . 14

3.2.8 Copy . . . 14

3.2.9 Paste . . . 15

3.2.10 Delete . . . 15

3.2.11 Instantiate . . . 15

3.2.12 Highlight . . . 15

3.2.13 Reset Selection . . . 15

3.2.14 Undo . . . 15

3.2.15 Redo . . . 16

3.2.16 About . . . 16

3.3 Drawing Area . . . 16

3.3.1 Navigate . . . 16

3.3.2 Edit . . . 16

3.3.3 Create . . . 17

3.4 Element Configuration Area . . . 17

3.4.1 Maximum stringlengths . . . 17

3.4.2 Commit . . . 18
3.5 Notification Bar . . . 18
3.5.1 Information . . . 18
3.5.2 Warning . . . 19
3.5.3 Error . . . 19

4 Modelling Guidelines **19**
4.1 crystal_facet_uml Hints . . . 19
4.1.1 Tree Structure . . . 19
4.1.2 Focus . . . 19
4.1.3 Namespaces . . . 20
4.1.4 Attic/Storage room . . . 20
4.2 General Hints on Architecture Documentation . . . 20
4.2.1 Problem vs. Solution . . . 20
4.2.2 Names . . . 20
4.2.3 Description . . . 20
4.2.4 Precise sentences . . . 20
4.2.5 Distinguish similar things . . . 20

A Download Information **20**
A.1 Download Links . . . 20
A.1.1 Install . . . 21
A.1.2 License . . . 21

1 Introduction

crystal_facet_uml is an application to support software architects.

1.1 Goal

crystal_facet_uml creates a set of uml diagrams.

It ensures consistency of uml elements: Their names and their relationships are in sync in all diagrams.

crystal_facet_uml exports diagrams in various vector and pixel-based image formats.

1.2 Features

crystal_facet_uml provides a graphical user interface to

- create diagrams
  (use-case, deployment, component, composite-structure, package, class, activity, state, timing, communication, sequence)
- create uml elements
  (actor, system-boundary, use-case, node, component, part, interface, package, class, activity, state, object, artifact, comment, requirement)
- move, modify and delete uml elements
- create, modify and delete relationships
  (dependency, association, aggregation, composition, generalization, realization, contains, sync-call, return-call, async-message, communication-path, control-flow, object-flow, deployment, manifest, include, extend)
- create, modify and delete features
  (port, field, operation)
- cut, copy, paste uml elements between diagrams
- undo and redo are supported
- multiple windows can show different or same parts of the uml model

Diagrams are layouted part-automatically:

- The user chooses the relative location of uml elements towards others
- crystal_facet_uml selects the exact locations of uml elements
- The user controls the positions of messages/transitions in sequence and timing diagrams

- crystal_facet_uml auto-layouts relationships in other diagrams

crystal_facet_uml manages a meta model:

- Diagrams are organized as a tree, similar to a book's table-of-contents
- Uml elements exist only once even if shown in many diagrams
- Relationships and features are consistent between all diagrams
- Diagram-local messages/transitions are supported in scenario-based diagrams
  (sequence, communication, timing)

crystal_facet_uml exports diagrams as

- vector graphics
  (pdf, ps, svg)
- pixel graphics
  (png)
- textual representation
  (utf-8-txt, docbook, xhtml)

crystal_facet_uml can also be started from command line to check and repair database files.

1.3 Usage Overview

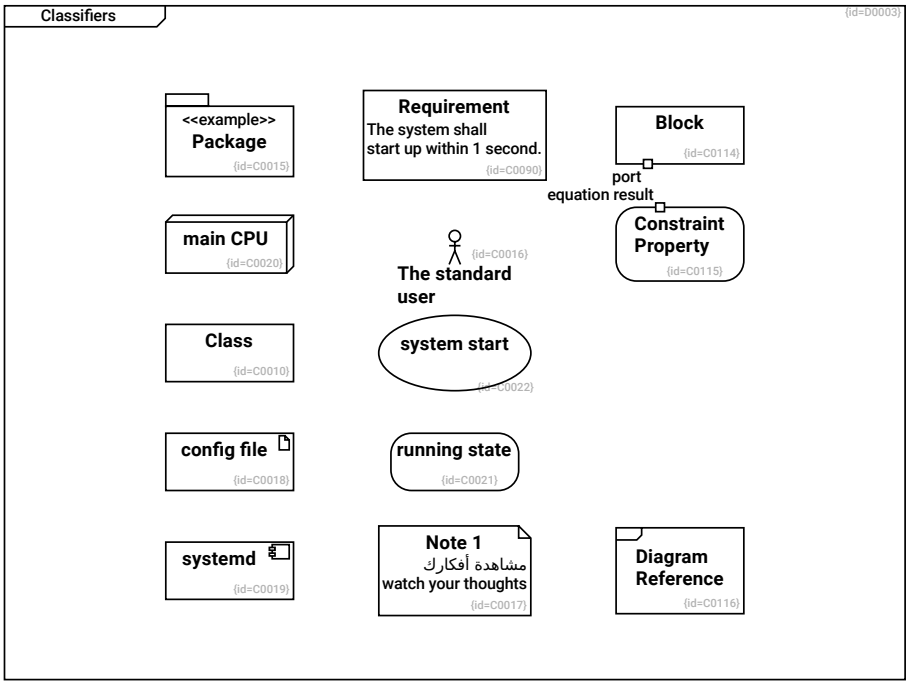
crystal_facet_uml can be started in graphical mode (see Section 3) or from command line (for help run **`crystal_facet_uml -h`**).

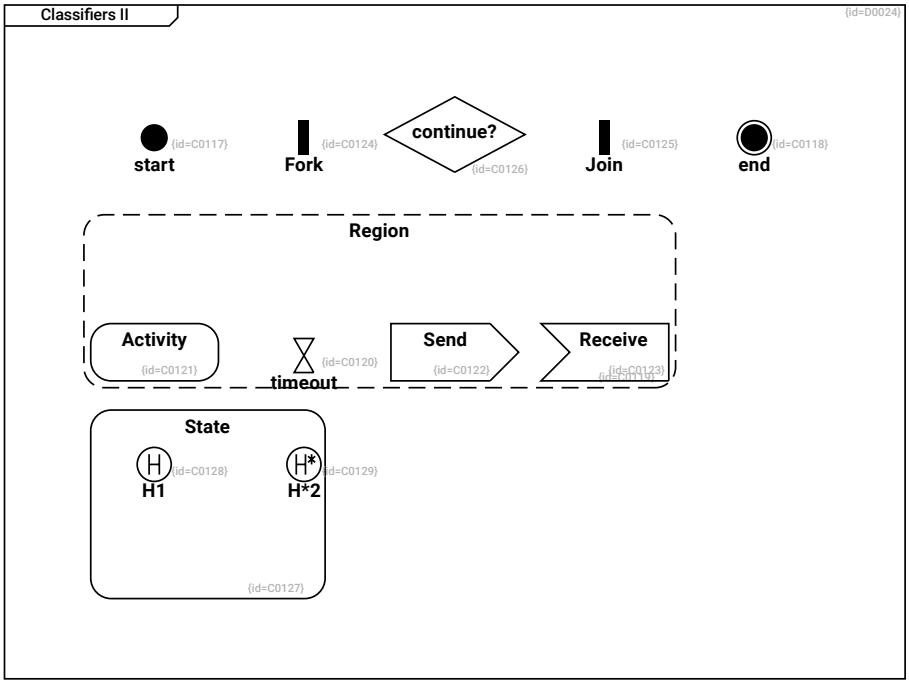
2 Example Diagrams

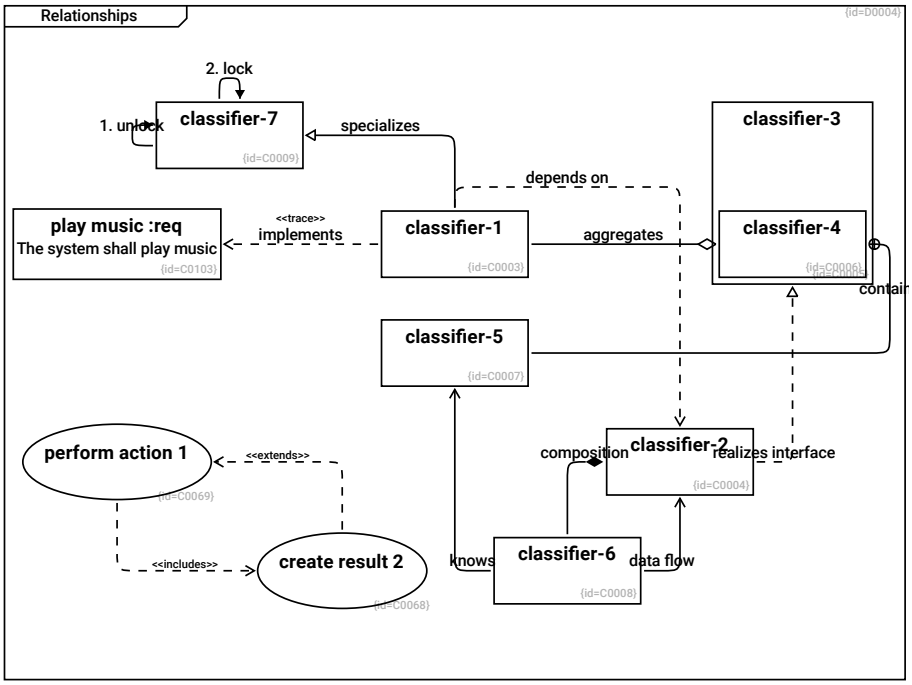
This sections presents the features of crystal_facet_uml.

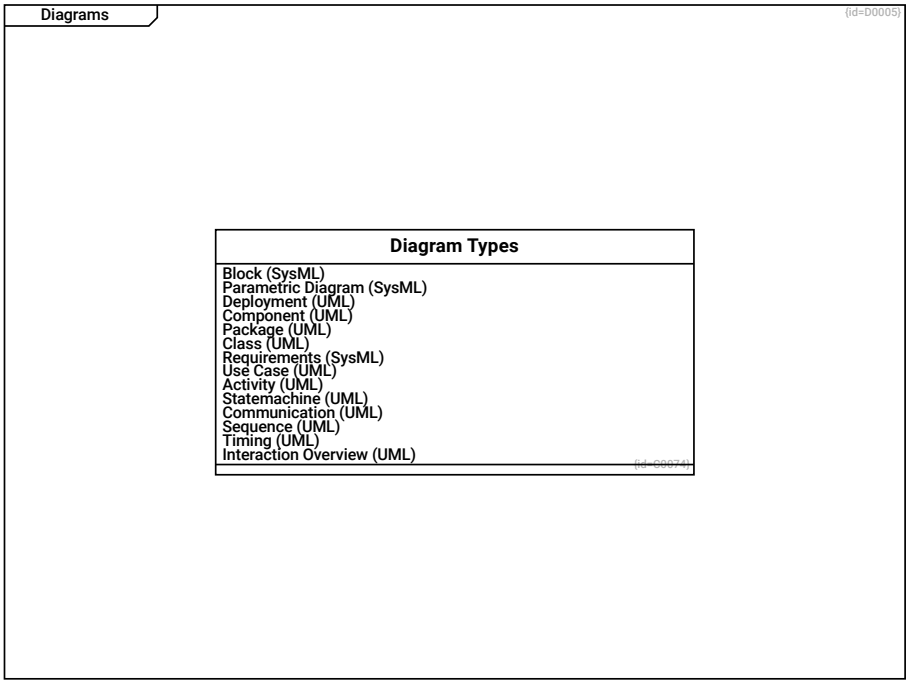
2.1 Feature List

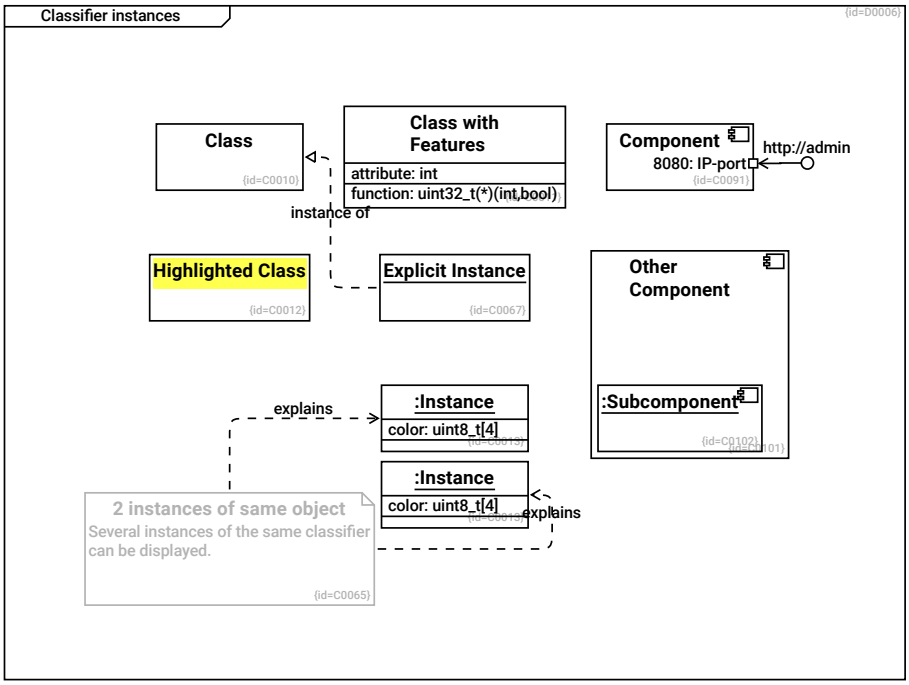
This section lists what kind of elements crystal_facet_uml can draw in diagrams.


Classifiers
{id=D0003}
<<example>>
Package
{id=C0015}
Requirement
The system shall
start up within 1 second.
{id=C0090}
Block
{id=C0114}
port
equation result
Constraint
Property
{id=C0115}
main CPU
{id=C0020}
{id=C0016}
The standard
user
Class
{id=C0010}
system start
{id=C0022}
config file
{id=C0018}
running state
{id=C0021}
systemd
{id=C0019}
Note 1
مشاهدة أفكارك
watch your thoughts
{id=C0017}
Diagram
Reference
{id=C0116}


Classifiers II
{id=D0024}
{id=C0117}
start
{id=C0124}
Fork
continue?
{id=C0126}
{id=C0125}
Join
{id=C0118}
end
Region
Activity
{id=C0121}
{id=C0120}
timeout
Send
{id=C0122}
Receive
{id=C0123}
State
{id=C0128}
H1
{id=C0129}
H*2
{id=C0127}

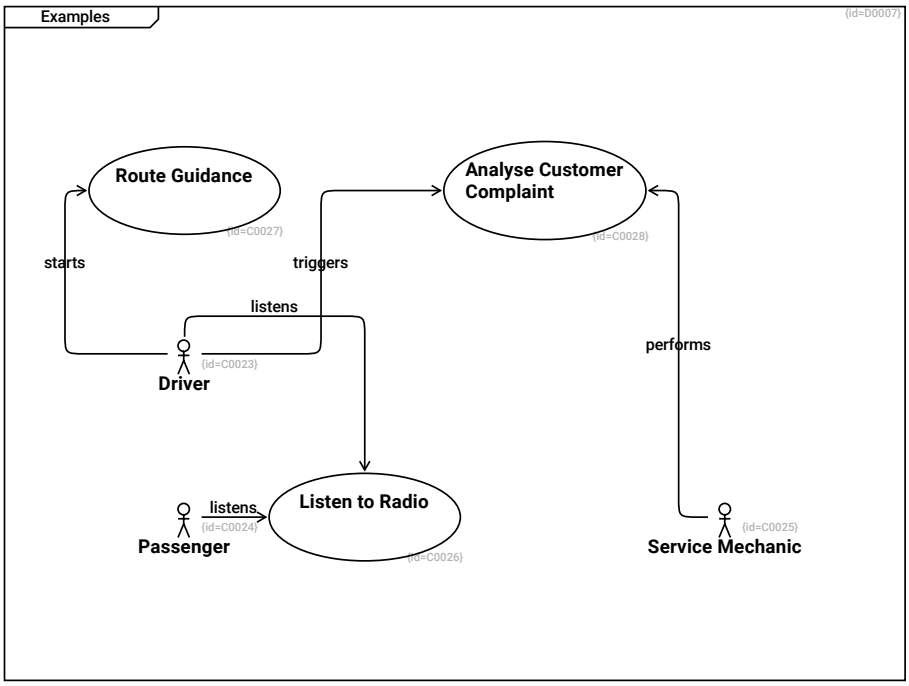

Relationships
{id=D0004}
2. lock
1. unlock
classifier-7
{id=C0009}
specializes
classifier-3
depends on
play music :req
The system shall play music
{id=C0103}
<<trace>>
implements
classifier-1
{id=C0003}
aggregates
classifier-4
{id=C0006}
contain
classifier-5
{id=C0007}
perform action 1
{id=C0069}
<<extends>>
composition
classifier-2
{id=C0004}
realizes interface
<<includes>>
create result 2
{id=C0068}
knows
classifier-6
{id=C0008}
data flow

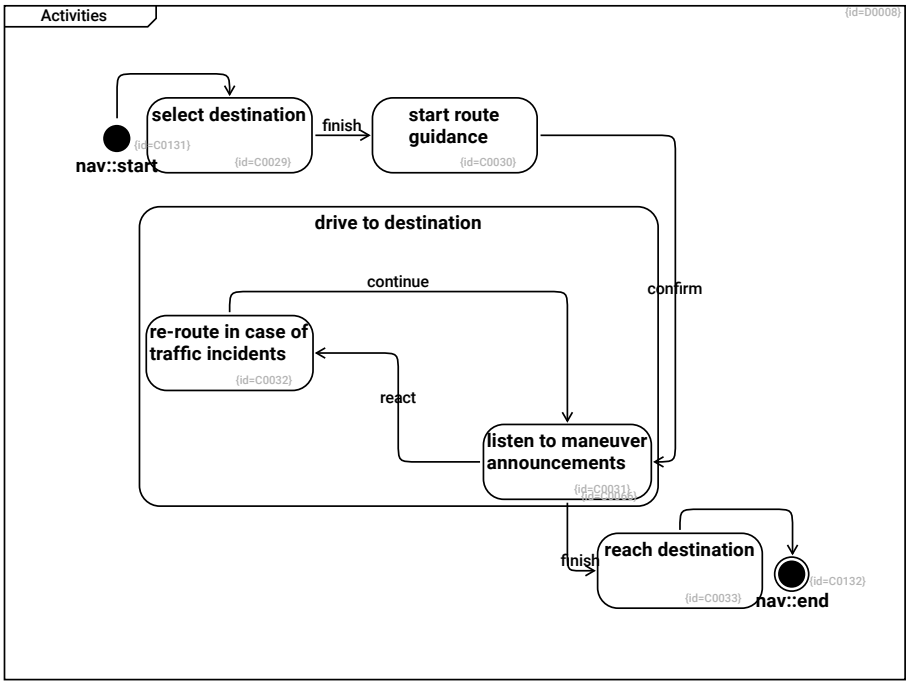

Diagrams
{id=D0005}
Diagram Types
Block (SysML)
Parametric Diagram (SysML)
Deployment (UML)
Component (UML)
Package (UML)
Class (UML)
Requirements (SysML)
Use Case (UML)
Activity (UML)
Statemachine (UML)
Communication (UML)
Sequence (UML)
Timing (UML)
Interaction Overview (UML)
{id=C0074}

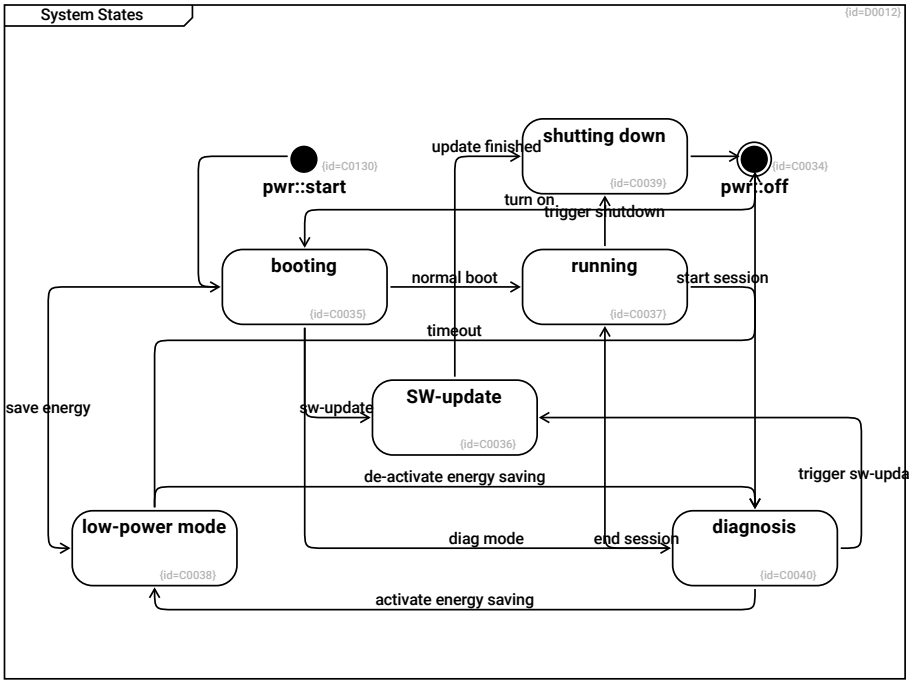


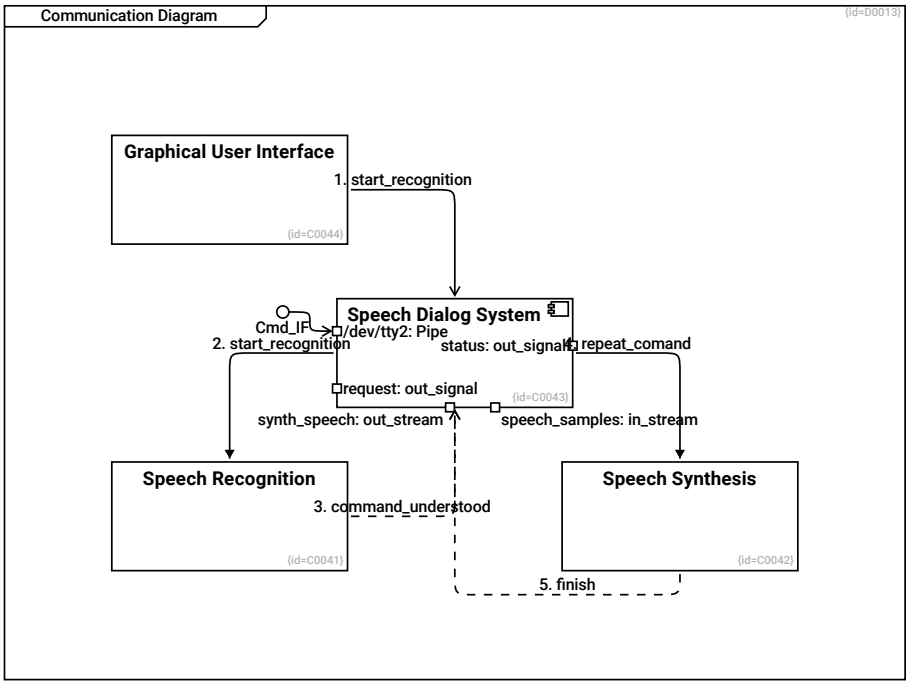
2.2 Example UML Behavioral Views

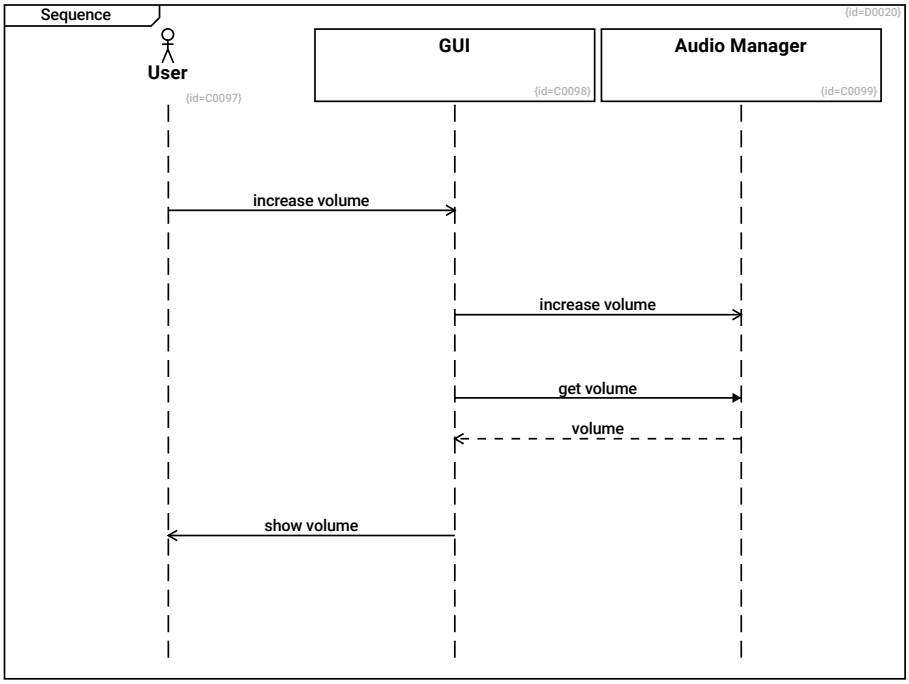
This section lists what kind of elements crystal_facet_uml can draw in diagrams.

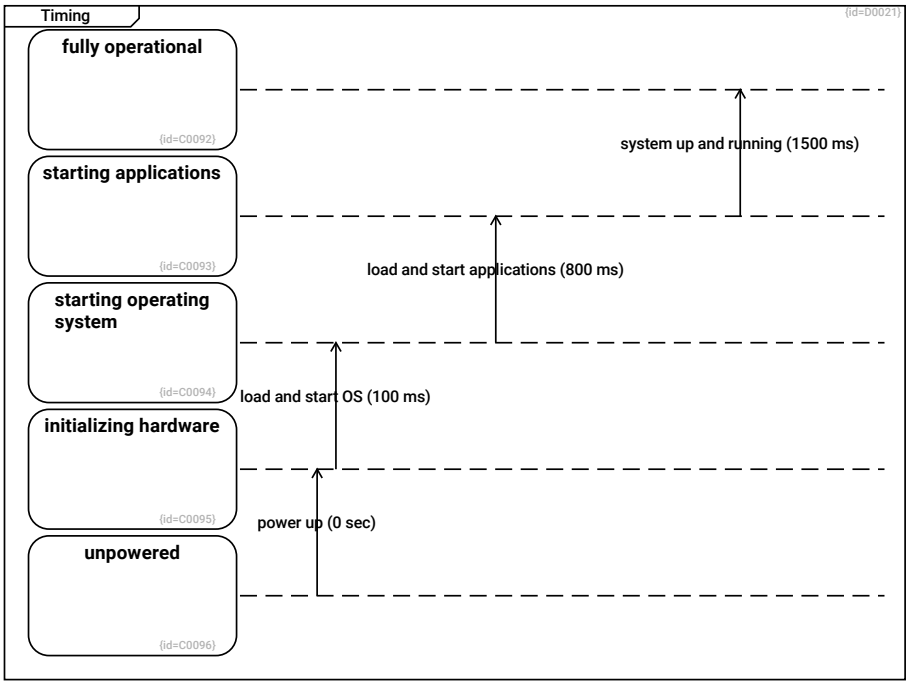


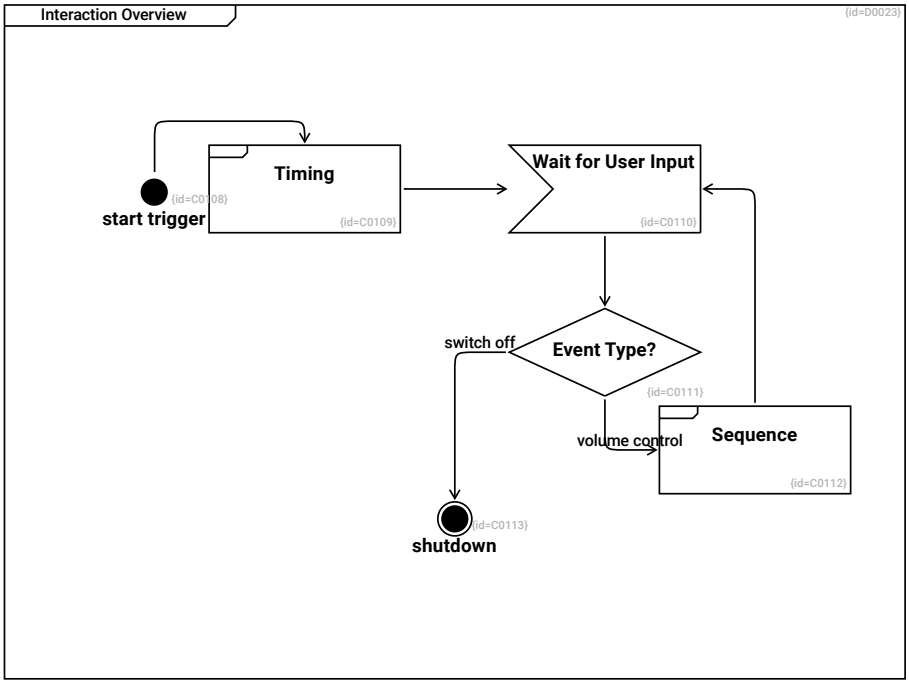

Activities
{id=D0008}
select destination
{id=C0131}
{id=C0029}
finish
start route guidance
{id=C0030}
nav::start
drive to destination
continue
confirm
re-route in case of traffic incidents
{id=C0032}
react
listen to maneuver announcements
{id=C0031}
finish
reach destination
{id=C0033}
{id=C0132}
nav::end


System States
{id=D0012}
{id=C0130}
pwr::start
update finished
shutting down
{id=C0039}
{id=C0034}
pwr::off
turn on
trigger shutdown
booting
{id=C0035}
normal boot
running
{id=C0037}
start session
timeout
save energy
sw-update
SW-update
{id=C0036}
de-activate energy saving
trigger sw-upda
low-power mode
{id=C0038}
diag mode
end session
diagnosis
{id=C0040}
activate energy saving


Communication Diagram
{id=D0013}
Graphical User Interface
{id=C0044}
1. start_recognition
Speech Dialog System
Cmd_IF
/dev/tty2: Pipe
2. start_recognition
status: out_signal
repeat_comand
request: out_signal
{id=C0043}
synth_speech: out_stream
speech_samples: in_stream
Speech Recognition
{id=C0041}
3. command_understood
Speech Synthesis
{id=C0042}
5. finish

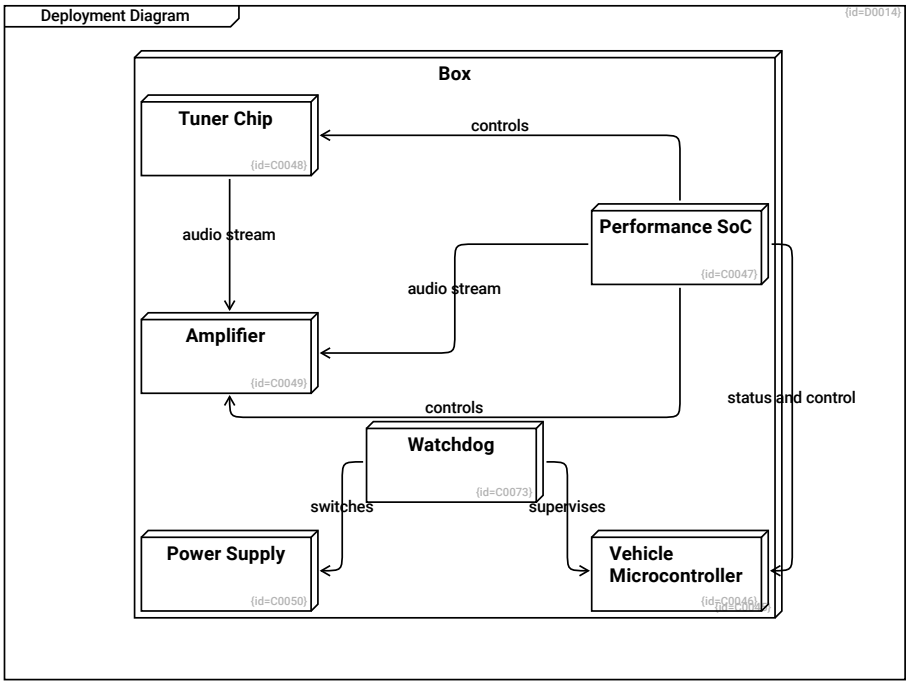

Sequence
{id=D0020}
User
{id=C0097}
GUI
{id=C0098}
Audio Manager
{id=C0099}
increase volume
increase volume
get volume
volume
show volume

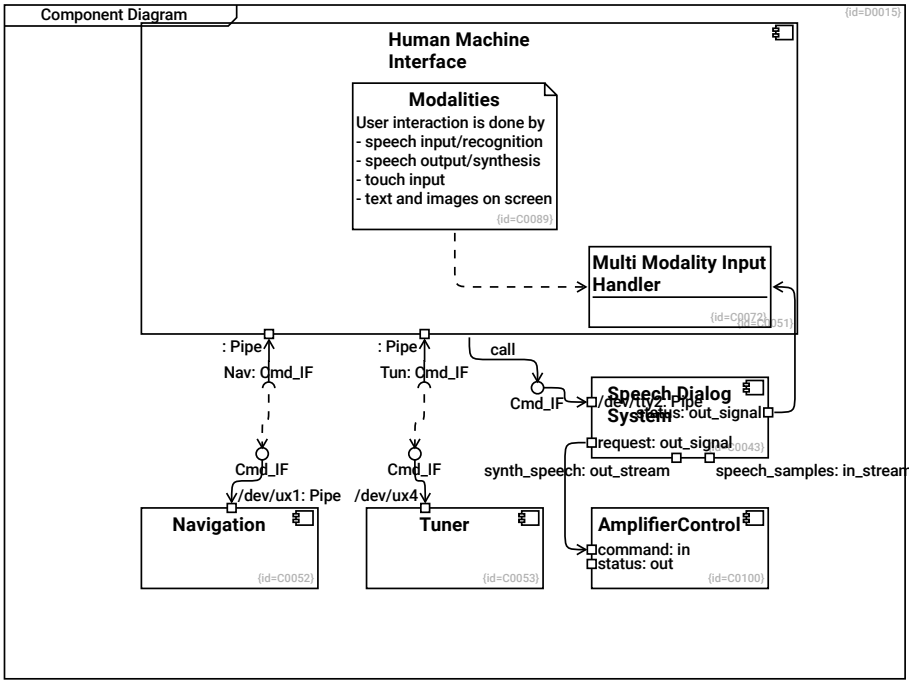


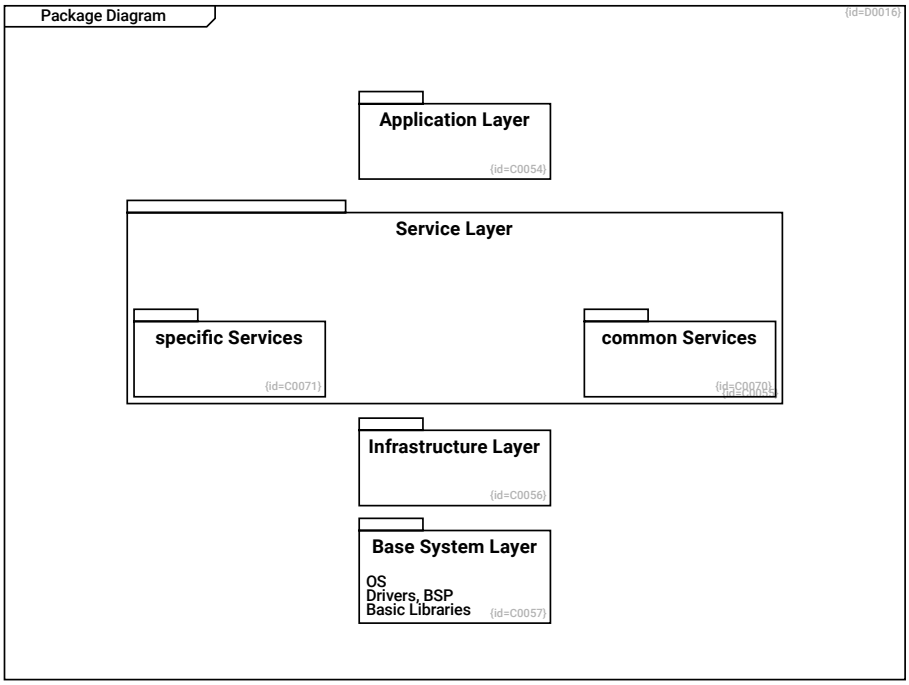


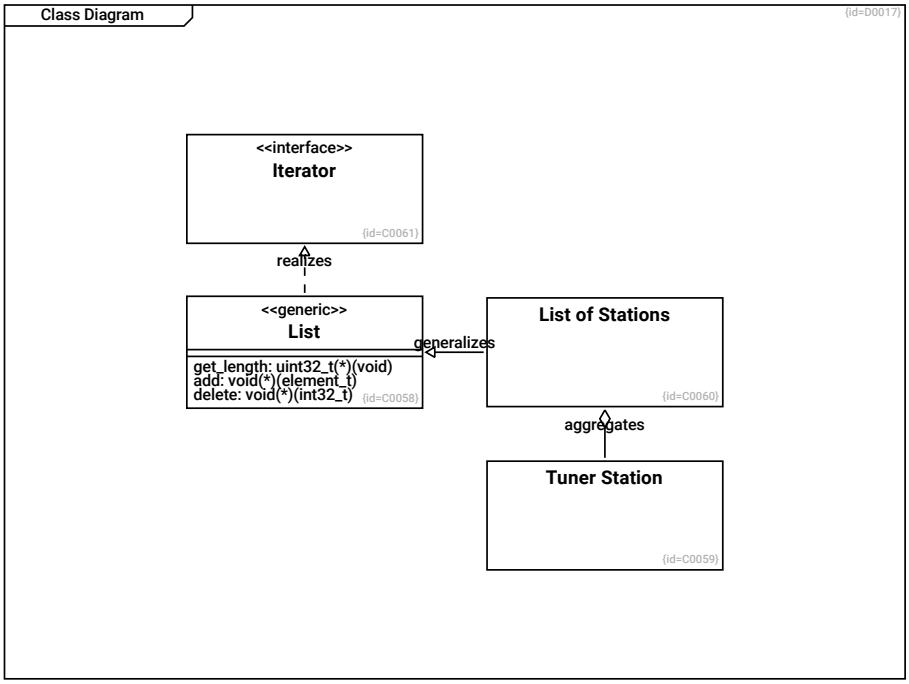
2.3 Example UML Static Views

This section lists what kind of elements crystal_facet_uml can draw in diagrams.


Deployment Diagram
{id=D0014}
Box
Tuner Chip
{id=C0048}
controls
audio stream
Performance SoC
{id=C0047}
audio stream
Amplifier
{id=C0049}
controls
status and control
Watchdog
{id=C0073}
switches
supervises
Power Supply
{id=C0050}
Vehicle Microcontroller
{id=C0046}

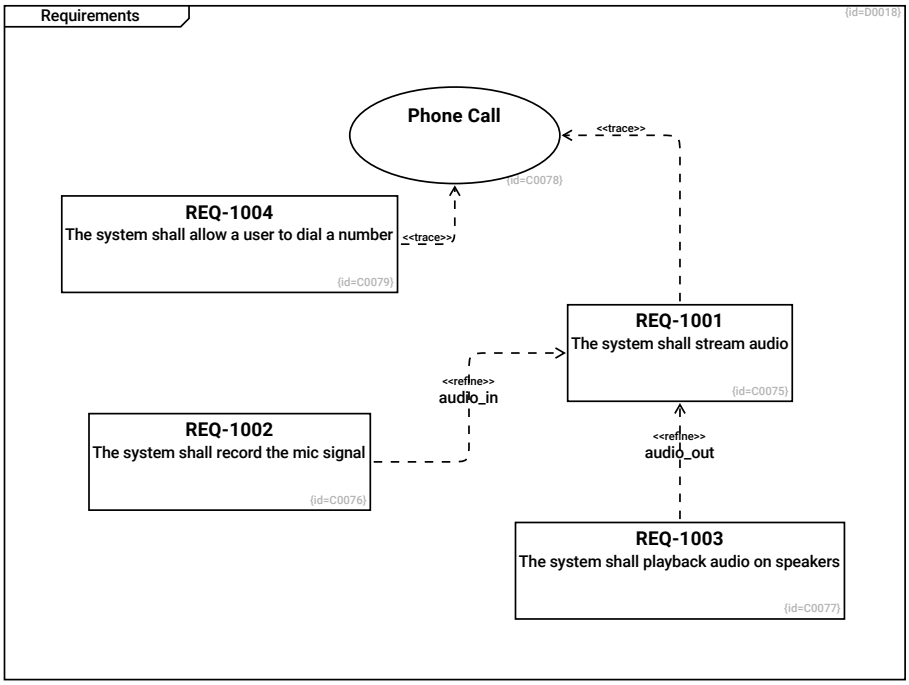

Component Diagram
{id=D0015}
Human Machine Interface
Modalities
User interaction is done by
- speech input/recognition
- speech output/synthesis
- touch input
- text and images on screen
{id=C0089}
Multi Modality Input Handler
{id=C0072}
: Pipe
Nav: Cmd_IF
: Pipe
Tun: Cmd_IF
call
Cmd_IF
Speech Dialog System
/dev/ttyZ: Pipe
status: out_signal
request: out_signal
Cmd_IF
Cmd_IF
synth_speech: out_stream
speech_samples: in_stream
/dev/ux1: Pipe
/dev/ux4
Navigation
{id=C0052}
Tuner
{id=C0053}
AmplifierControl
command: in
status: out
{id=C0100}

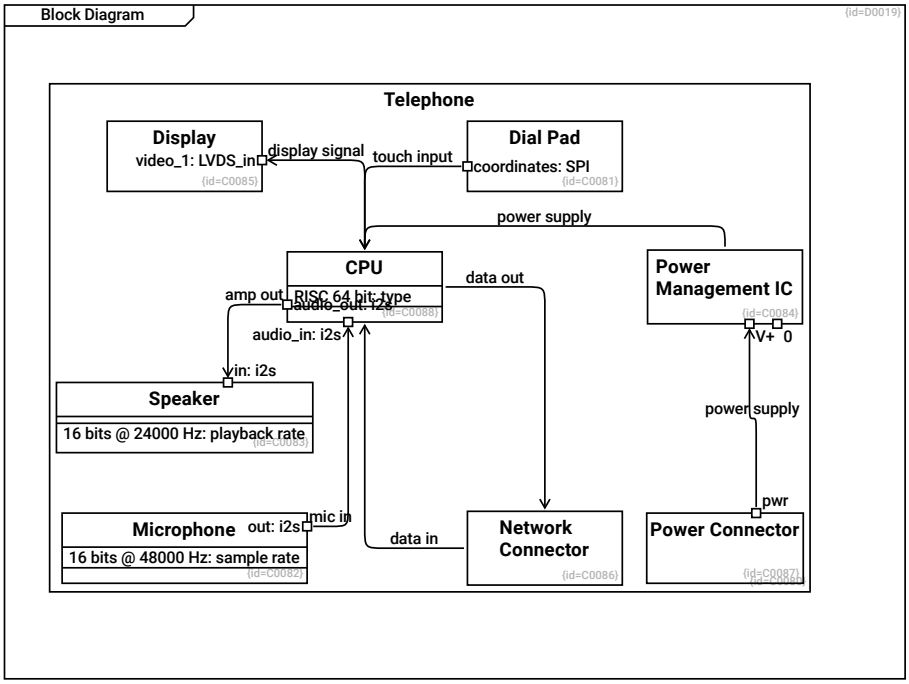


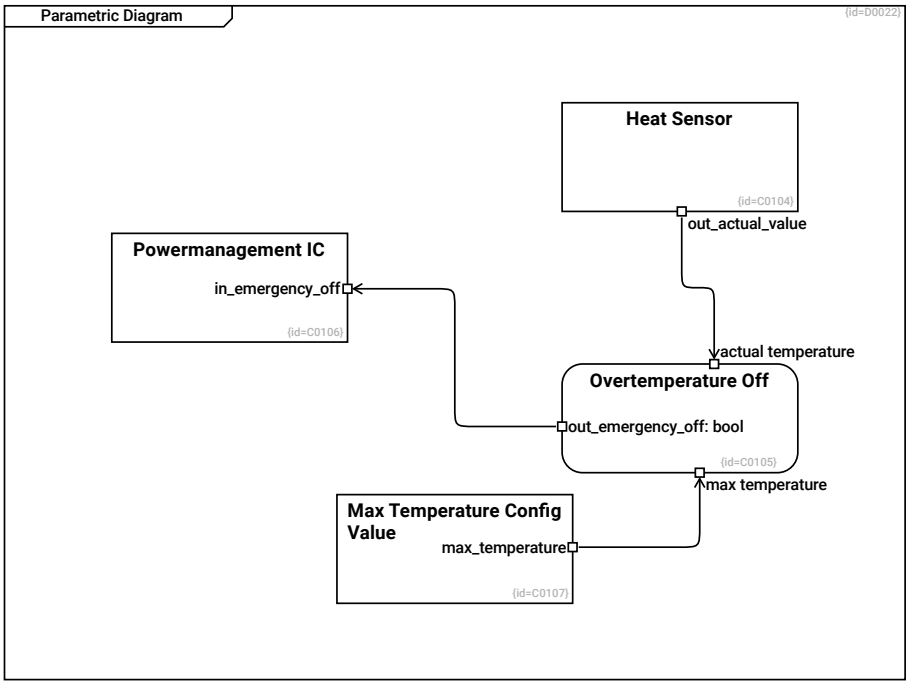


2.4 Example SysML Views

This section lists what kind of elements crystal_facet_uml can draw in diagrams.


Requirements
{id=D0018}
Phone Call
{id=C0078}
<<trace>>
REQ-1004
The system shall allow a user to dial a number
{id=C0079}
<<trace>>
REQ-1001
The system shall stream audio
{id=C0075}
<<refine>>
audio_in
REQ-1002
The system shall record the mic signal
{id=C0076}
<<refine>>
audio_out
REQ-1003
The system shall playback audio on speakers
{id=C0077}


Block Diagram
{id=D0019}
Telephone
Display
video_1: LVDS_in
{id=C0085}
display signal
touch input
Dial Pad
coordinates: SPI
{id=C0081}
power supply
CPU
RISC 64 bit type
{id=C0088}
amp out
audio_out: i2s
audio_in: i2s
Power
Management IC
{id=C0084}
V+ 0
in: i2s
Speaker
16 bits @ 24000 Hz: playback rate
{id=C0083}
power supply
pwr
Microphone
out: i2s
mic in
16 bits @ 48000 Hz: sample rate
{id=C0082}
data out
data in
Network
Connector
{id=C0086}
Power Connector
{id=C0087}



2.5 Other Examples

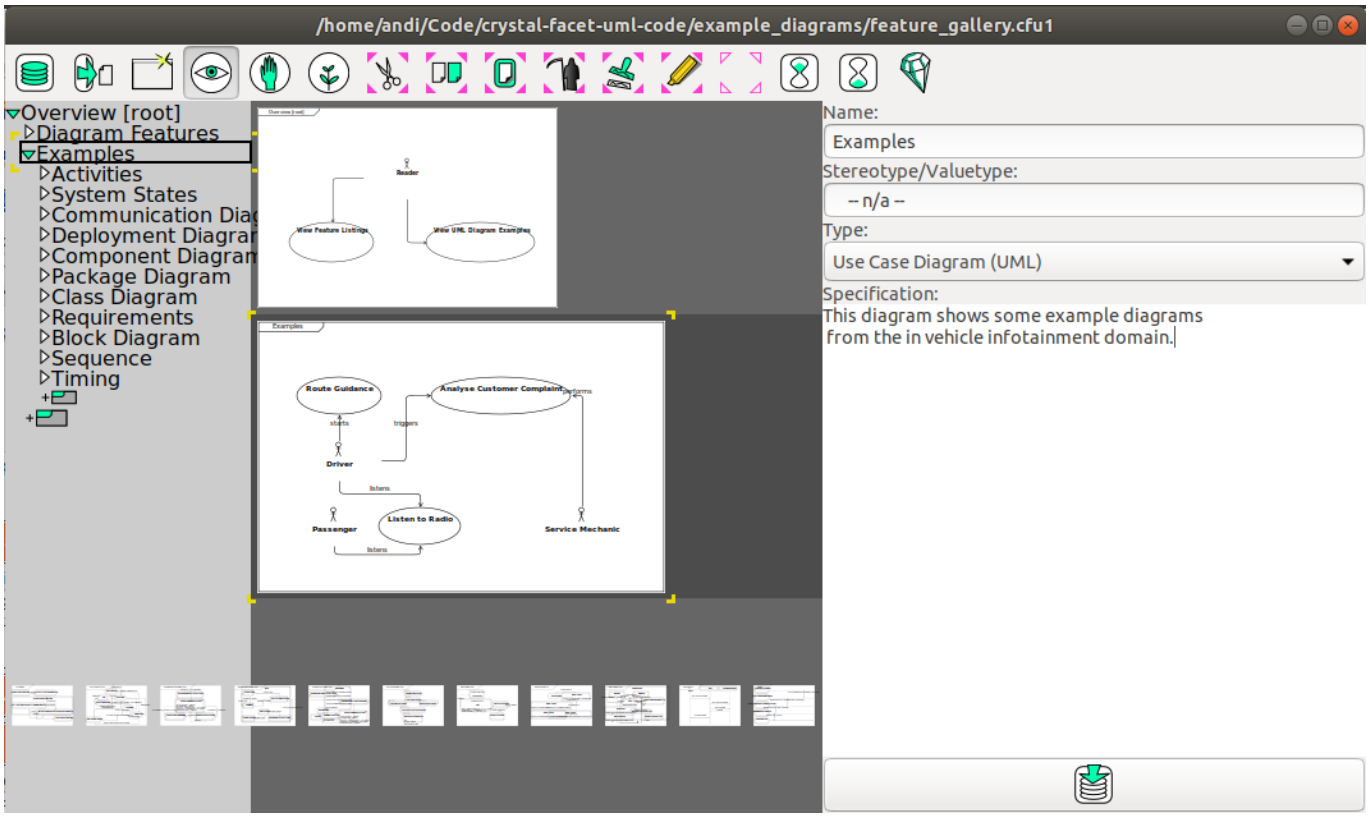
See https://github.com/awarnke/crystal_facet_uml/blob/master/example_diagrams/quality.pdf to get another view on how to use crystal_facet_uml.

3 GUI / Usage Manual

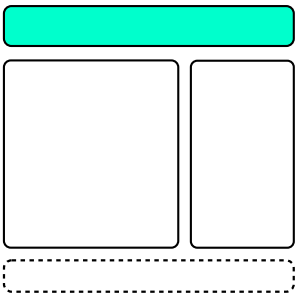
3.1 Window Area Overview

If started in graphical mode, crystal_facet_uml shows a window with

- toolbar on top,
- drawing area in the center,
- element configuration widgets to the right and
- an optional notification bar at the bottom.



3.2 Tool Bar



3.2.1 Create/Use DB

- Opens an existing database file or creates a new database file

3.2.2 Export

- Exports all diagrams to the selected folder (supported formats are txt, png, pdf, ps and svg)

3.2.3 New Window

- Opens another window on the same database.

This option allows you to work reliably with multiple windows on the same database.

3.2.4 Navigate

- Navigate to parent or child diagrams
- Create a new diagram (see Section 3.3.1)

3.2.5 Edit

- Modify elements in the diagram (see Section 3.3.2)

3.2.6 Create

- Create elements in the diagram (see Section 3.3.3)

3.2.7 Cut

- Cut all pink-cornered elements to the clipboard (features of classifiers are cut independantly of their corner-colors)

3.2.8 Copy

- Copy all pink-cornered elements to the clipboard (features of classifiers are copied independantly of their corner-colors)

3.2.9 Paste

- Pastes diagrams and classifiers from the clipboard to the uml model. (Relationships are not pasted) If id and name are identical to an existing element, an instance of the existing element is pasted to the diagram. Otherwise a new element is created.

3.2.10 Delete

- Deletes all pink-cornered elements. This operation may fail if a marked diagram contains unmarked elements.

3.2.11 Instantiate

- Toggles the pink-cornered classifiers between classes and anonymous instances.
- No effect on classifiers that are already instances: Object, Part.
- No effect on relationships and features.

3.2.12 Highlight

- Toggles the pink-cornered classifiers between yellow-marked, greyed-out and normal. (Does not work for relationships and features)

3.2.13 Reset Selection

- Resets the pink-cornered selection

3.2.14 Undo

- Un-does the last operation (Opening a database and exporting files cannot be undone)

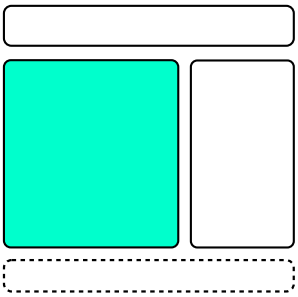
3.2.15 Redo

- Re-does the last un-done operation

3.2.16 About

- Shows version, license and copyrights

3.3 Drawing Area



Diagrams are layouted automatically. You can influence the locations of classifiers only. When adding too many classifiers or relations, auto layouting may not achieve the expected results. In many cases, splitting the diagram into two or more diagrams solves the layouting issues and at the same time improves understandability by focusing on one aspect/topic per diagram.

3.3.1 Navigate

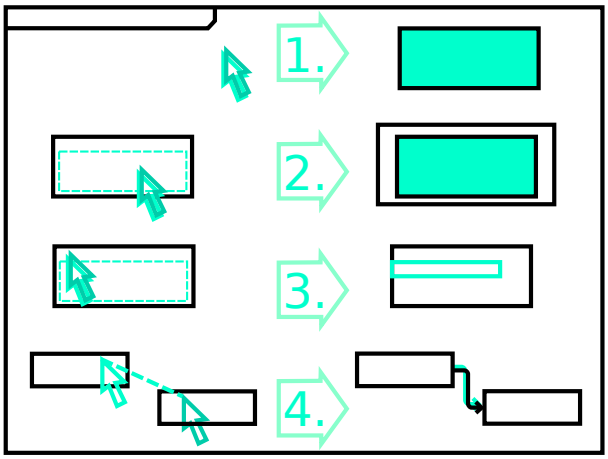
- To navigate to parent, sibling or children diagrams, click on the diagram.
- To create a new diagram, click on the icon, or the smaller icon for a new child-diagram.
- To restructure the diagram tree, drag a diagram name to the new location.

3.3.2 Edit

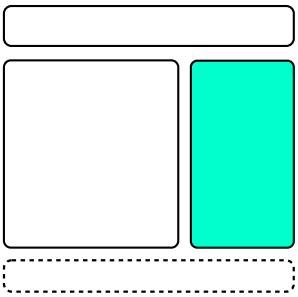
- To select the diagram or a classifier or a feature or a relationship with yellow corners, click on this object.
- To mark an element with pink corners, click on these objects twice.
- To move classifiers within the diagram, 1.) press, 2.) drag and 3.) release the mouse button.

3.3.3 Create

1. To create a classifier, click at an empty space in the diagram.
2. To create a child classifier, click into the white space of a classifier. (Alternatively, create a classifier (see 1) and a containment relationship (see 4).)
3. To create a feature, click onto a classifier (name or border).
4. To create a relationship, press on the source classifier and drag it to the destination classifier.



3.4 Element Configuration Area



Edit the properties of the yellow-cornered object.

- name of the focused object
- stereotype/valuetype of the focused object (deactivated depending on object-type)
- type of the focused object
- description of the focused object

3.4.1 Maximum stringlengths

All strings (names, descriptions, stereotypes) have a maximum length. Ascii characters require one, most other characters two bytes. Current sizes in bytes are: Classifiers:

- DATA_CLASSIFIER_MAX_NAME_LENGTH = 47,
- DATA_CLASSIFIER_MAX_STEREOTYPE_LENGTH = 47,
- DATA_CLASSIFIER_MAX_DESCRIPTION_LENGTH = 4095,

Features:

- DATA_FEATURE_MAX_KEY_LENGTH = 47, (name)
- DATA_FEATURE_MAX_VALUE_LENGTH = 255, (type)
- DATA_FEATURE_MAX_DESCRIPTION_LENGTH = 1023,

Relationships:

- DATA_RELATIONSHIP_MAX_NAME_LENGTH = 47,
- DATA_RELATIONSHIP_MAX_DESCRIPTION_LENGTH = 1023,

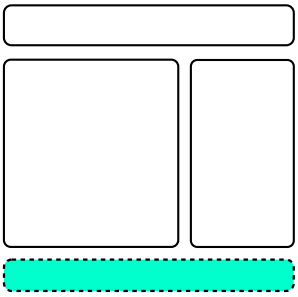
Diagrams:

- DATA_DIAGRAM_MAX_NAME_LENGTH = 47,
- DATA_DIAGRAM_MAX_DESCRIPTION_LENGTH = 8191,

3.4.2 Commit

- Stores the latest changes to the database immediately. This feature is optional, it is not necessary to explicitly save the file.

3.5 Notification Bar



3.5.1 Information

- Informs on success of an operation, e.g. an export

3.5.2 Warning

- Informs on a possible problem

3.5.3 Error

- Informs on an error

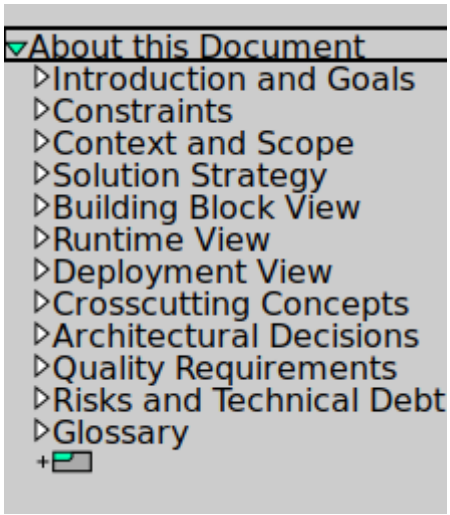
4 Modelling Guidelines

This page lists remarks on creating a software architecture and design document in general and it lists hints on getting along with the tool crystal_facet_uml. As all tools, this program has its strengths and weaknesses. This page helps in making use of the strenghts.

4.1 crystal_facet_uml Hints

4.1.1 Tree Structure

Diagrams are organized as a tree. Start the root of the tree explaining the document structure. At the second level of the tree, list the main areas to be shown, for example based on the arc42 template https://arc42.org/overview/ :



4.1.2 Focus

Put only few elements into each diagram. This increases understandability of the main purpuse of the diagram. Put further aspects of a topic into a separate diagram. Do not hesitate to copy an element from one diagram to the next. This is what crystal_facet_uml is good at: it keeps the model in sync.

4.1.3 Namespaces

Put a prefix to all your elements denoting its namespace. You can then distinguish a GLOBAL_START_STATE from an AUDIO_START_STATE. Or global::start from audio::start.

4.1.4 Attic/Storage room

If you are not sure if you really want to delete elements, 1) copy them to an attic-diagram and then 2) delete them from the original diagram.

4.2 General Hints on Architecture Documentation

4.2.1 Problem vs. Solution

Distinguish things that are

- given constraints (problem space),
- decisions, chosen and rejected alternatives and
- the designed solution

4.2.2 Names

Names of things are crucial: If the reader gets a wrong understanding by the name of an element, a hundred correct sentences of describing text cannot set this straight again.

4.2.3 Description

Every design element needs a description, maybe a list of responsibilities: What shall this element do, what is it for? Names alone cannot explain a system part.

4.2.4 Precise sentences

Be precise: Write in active form, e.g. The persistence component shall store and retrieve binary data records indentified by string-based keys.

4.2.5 Distinguish similar things

Things that are similar but not the same shall be different entities when modelling. E.g. The process in which an example application runs may be different from the storage location and may be different from the software-component. These are three things: Example_App_Process (Type: Node), Example_App_ObjectFile (Type:Artifact) and Example_App_SWComponent (Type:Component).

A Download Information

A.1 Download Links

Find the latest version at:

- https://sourceforge.net/projects/crystal-facet-uml/

- https://github.com/awarnke/crystal_facet_uml
- https://build.opensuse.org/package/show/home:awarnke/crystal_facet_uml

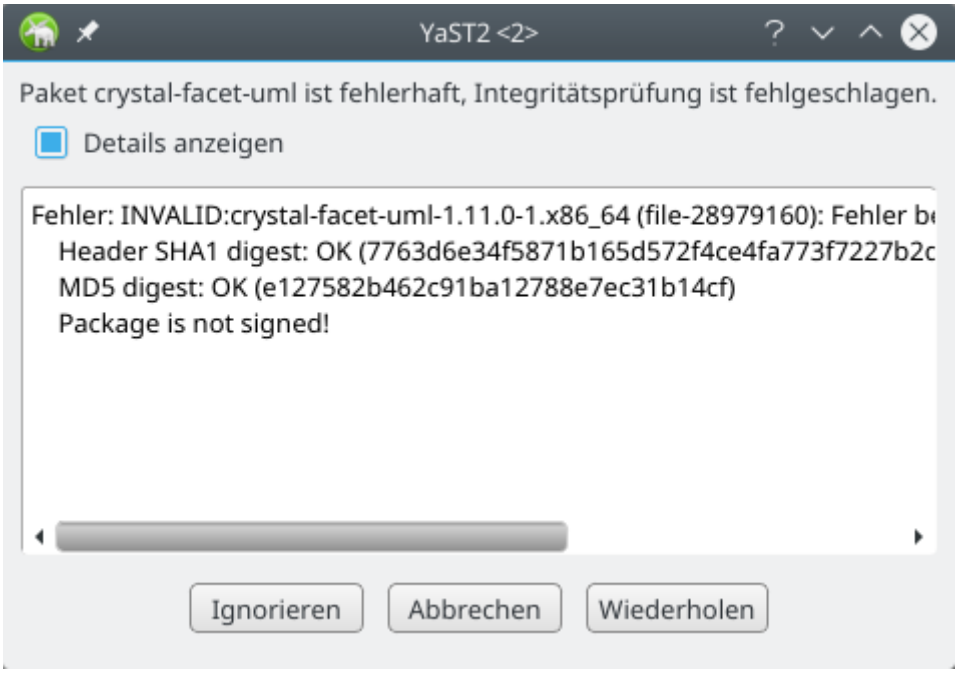
User documentation is available here:

- http://www.andreaswarnke.de/crystal_facet_uml/crystal_facet_uml_user_documentation.pdf
- https://github.com/awarnke/crystal_facet_uml/blob/master/user_doc/crystal_facet_uml_user_documentation.pdf

A.1.1 Install

The .deb and .rpm packages can be installed by the package installers of your system.

Because the packages are not signed, you may want to ignore the warning.



Alternatively, you may want to build the software from the .orig source-package and then install it by **`sudo make install`**; see the readme file for more information.

A.1.2 License

License of crystal_facet_uml is Apache-2.0. (c) 2016-2019 Andreas Warnke; Email-contact: cfu-at-andreaswarnke-dot-de