crystal-facet-uml documentation

Andreas Warnke

COLLABORATORS

	TITLE : crystal-facet-uml documentation	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Andreas Warnke	2026-11-01	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Introduction 1

2 Example Diagrams 3

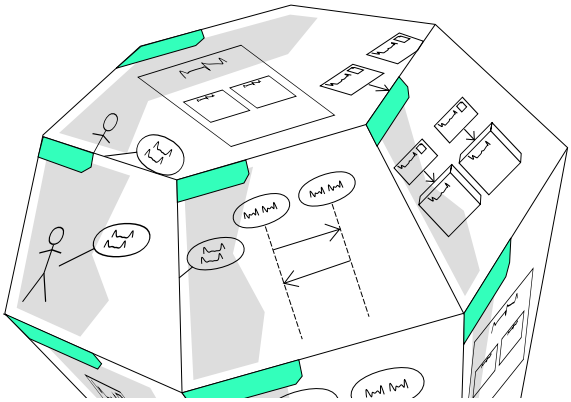
3 Graphical User Interface 13

4 Diagrams and Elements Spec 21

5 Modeling Guidelines 31

6 Command Line Interface 37

7 Download and Install 37

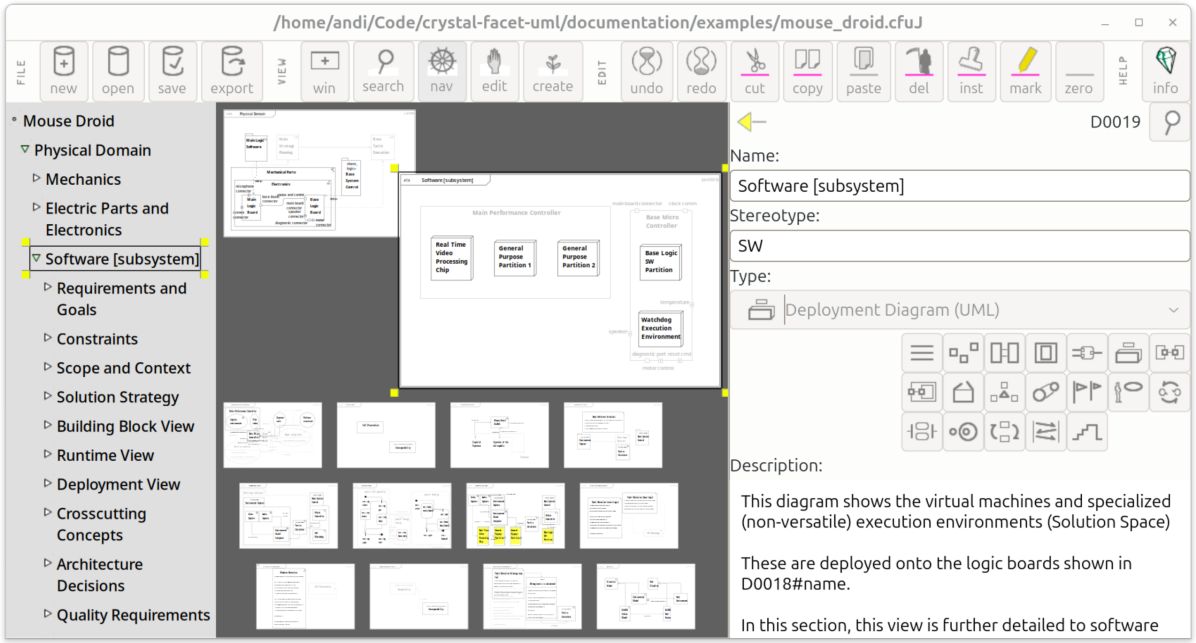


1 Introduction



crystal-facet-uml creates diagrams to document system and software architecture.

Like a crystal shows different facets of the same thing, the diagrams show different views of the same software or system.



1.1 Goal



As software architect, you create a set of diagrams describing use-cases, requirements, structural views, behavioral and deployment views.

These views show selected elements and their relationships. crystal-facet-uml keeps element names and element hierarchies consistent.

This tool runs on your local PC. It stores the model data in a json-based file which can be versioned in git, branched and merged together with your source code. crystal-facet-uml exports diagrams as svg, pdf, ps, png or the complete model as DocBook, html, json or xmi.

1.2 Features



crystal-facet-uml provides a graphical user interface to

- create, modify and delete diagrams and UML™/SysML™ elements
- cut, copy, paste elements between diagrams
- undo and redo are supported
- multiple windows can show different or same parts of the uml model
- search for elements

Diagrams are layouted part-automatically:

- The user chooses the relative location of elements towards others;
  crystal-facet-uml defines the exact locations of shown 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™/SysML™ 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 interaction diagrams:
  sequence, communication, timing, interaction overview
- Elements are automatically deleted when not visible anymore.
  Since version 1.70.0, crystal-facet-uml deletes invisible elements automatically
- These extension mechanisms of UML™ are supported:
  Tagged values and stereotypes including stereotype images

Diagrams can be exported as

- images: pdf, ps, svg, png
- text: utf-8, DocBook™, html
- machine-readable model: json, xmi™

crystal-facet-uml can also be started from command line

- to export all diagrams automatically
- to import a previously exported json file
- to check and repair database files

1.3 Usage Overview



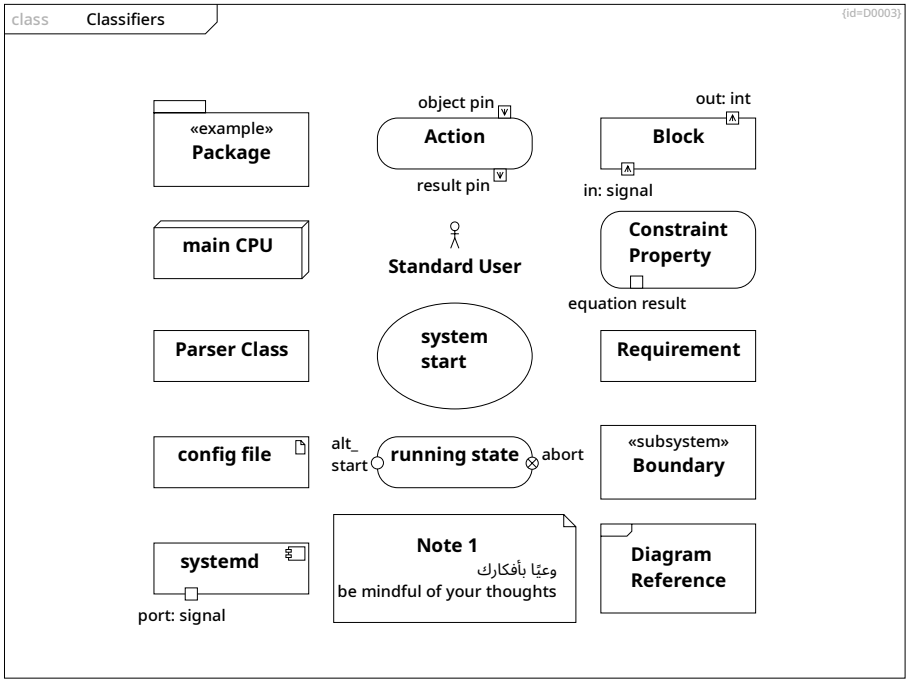
crystal-facet-uml can be started in graphical mode (see Section 3) or from command line (see Section 6).

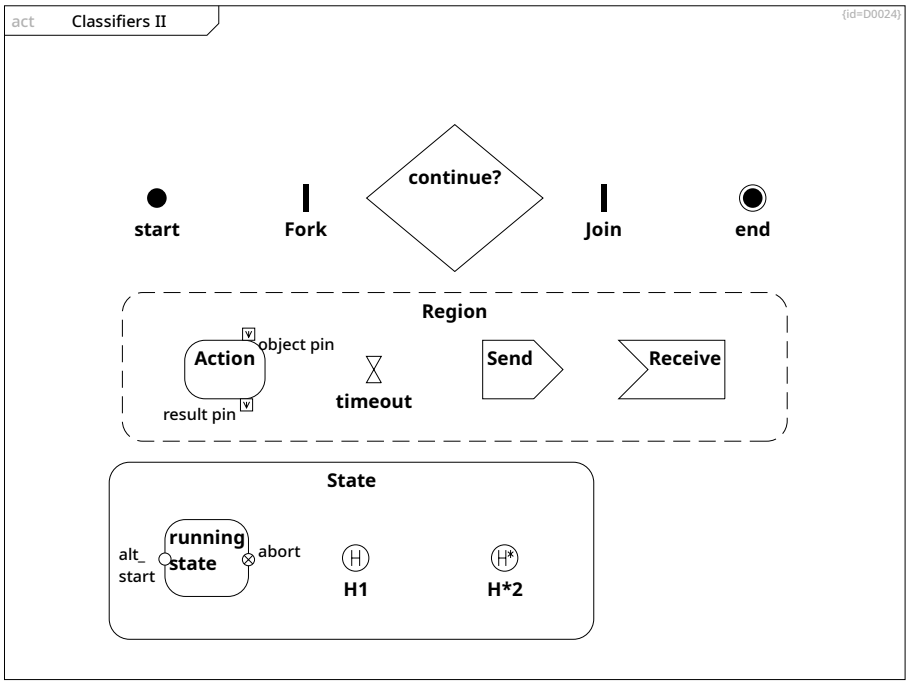
2 Example Diagrams

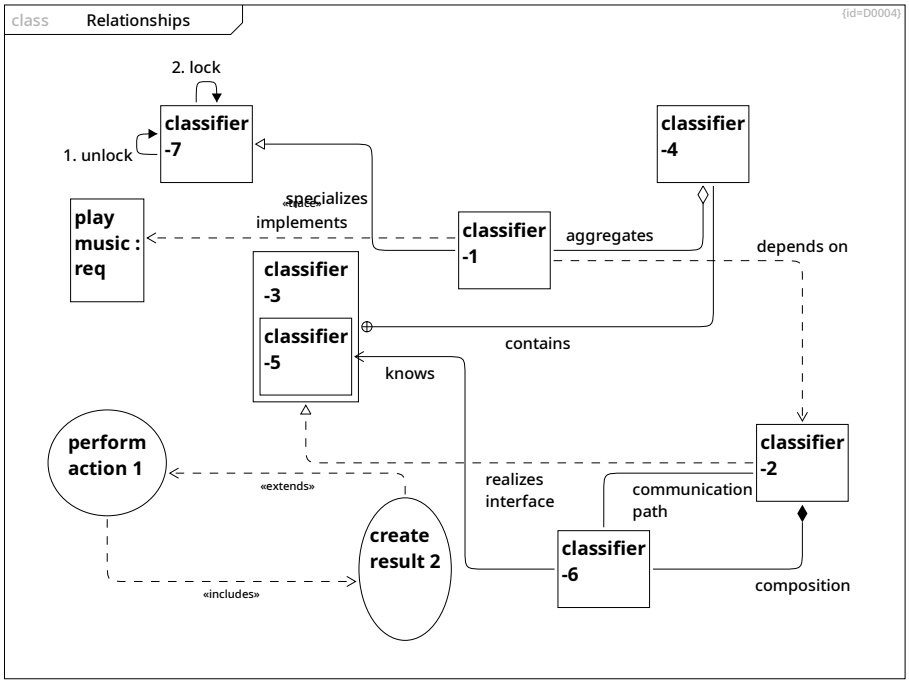
This sections presents the features of crystal-facet-uml.

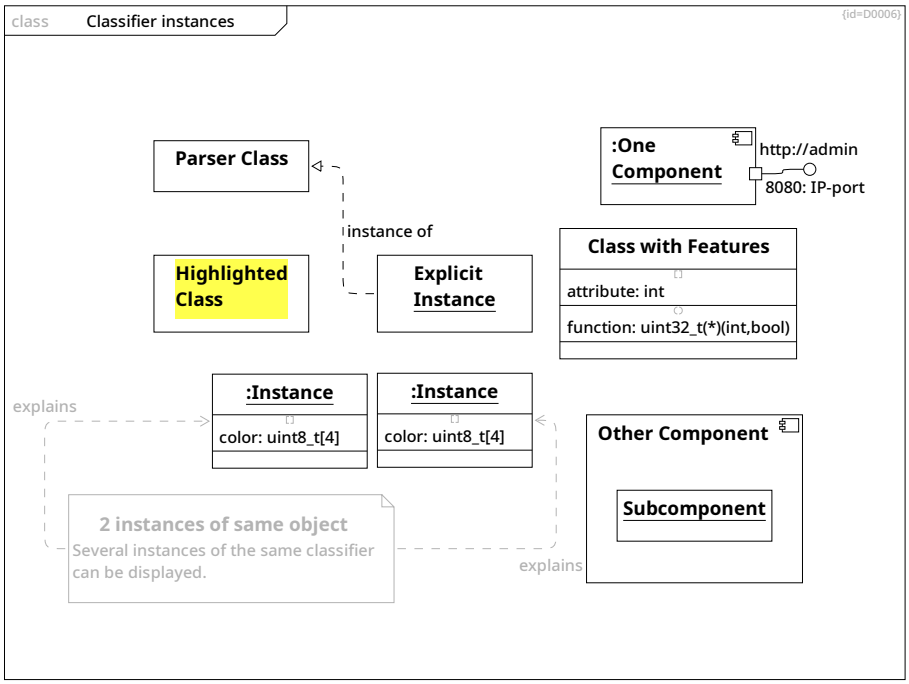
2.1 Feature List

This section lists what kind of elements crystal-facet-uml can draw in diagrams.



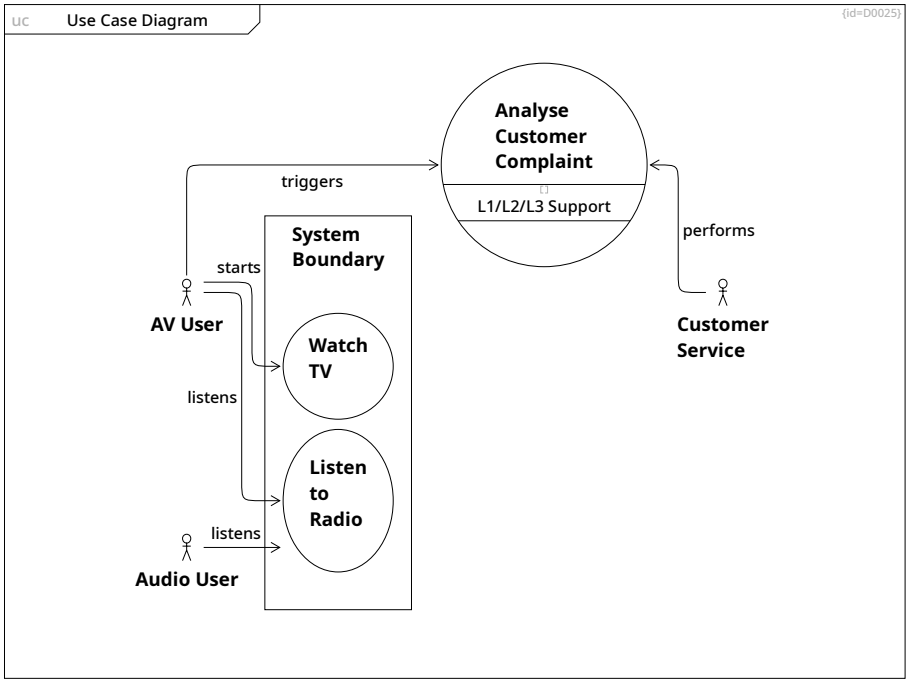

act
Classifiers II
{id=D0024}
start
Fork
continue?
Join
end
Region
Action
object pin
result pin
timeout
Send
Receive
State
alt_
start
running
state
abort
H1
H*2

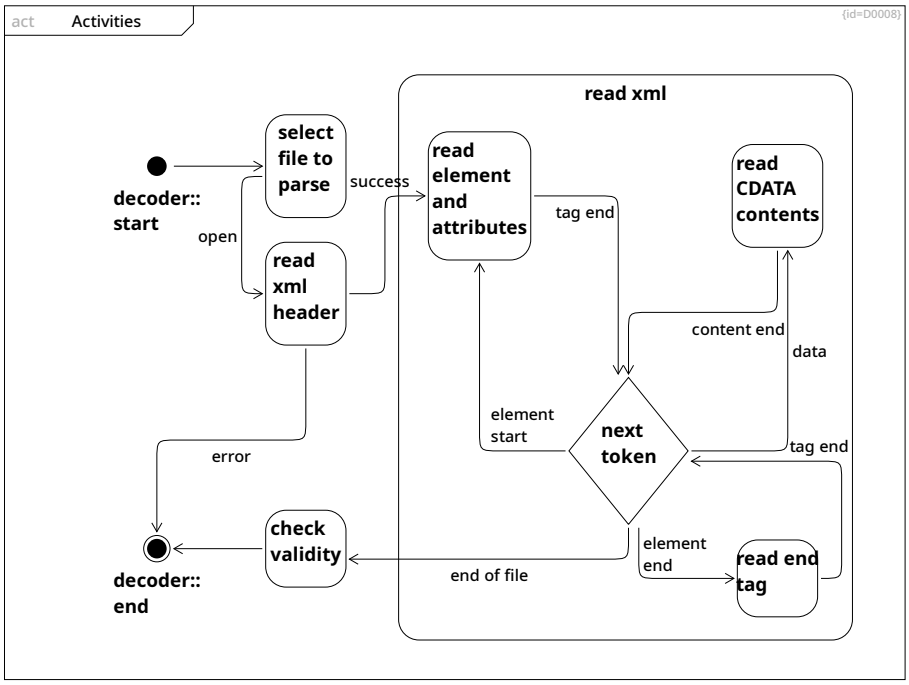

class
Relationships
{id=D0004}
2. lock
1. unlock
classifier
-7
classifier
-4
play
music :
req
specializes
implements
classifier
-1
aggregates
depends on
classifier
-3
classifier
-5
contains
knows
perform
action 1
«extends»
realizes
interface
communication
path
classifier
-2
create
result 2
classifier
-6
«includes»
composition

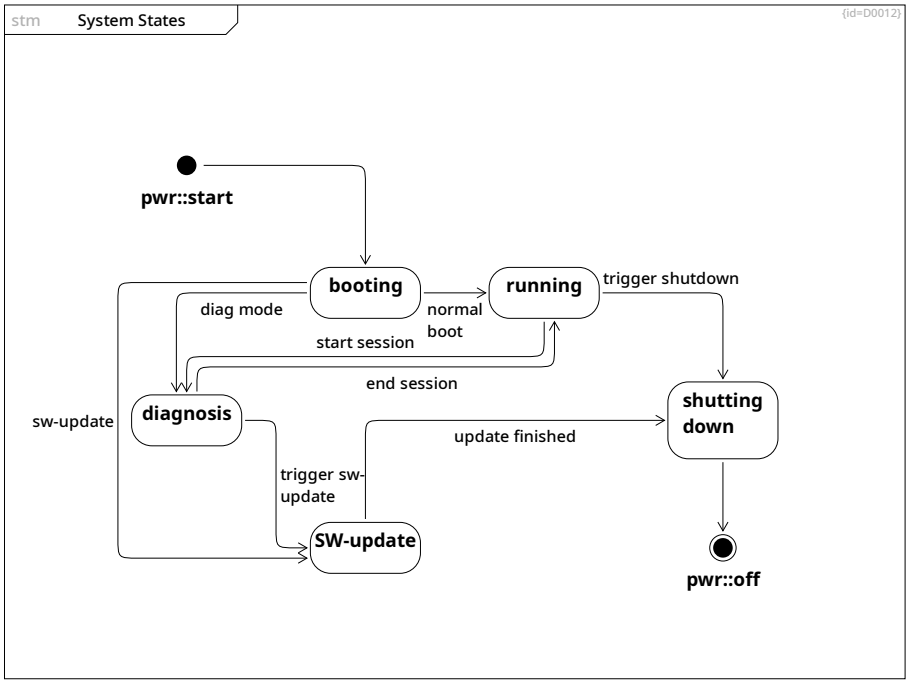


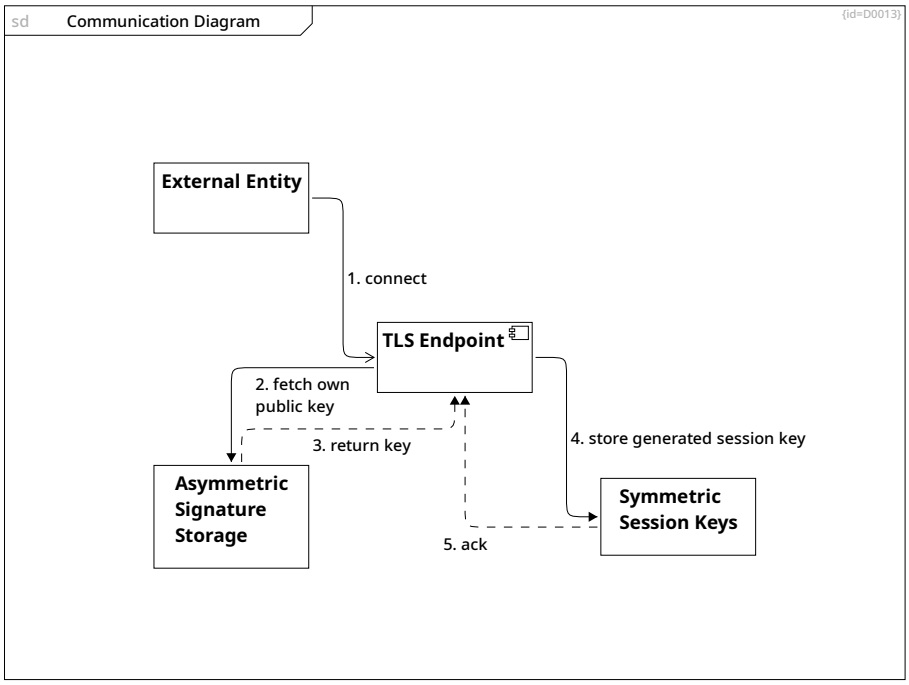
2.2 Example UML Behavioral Views

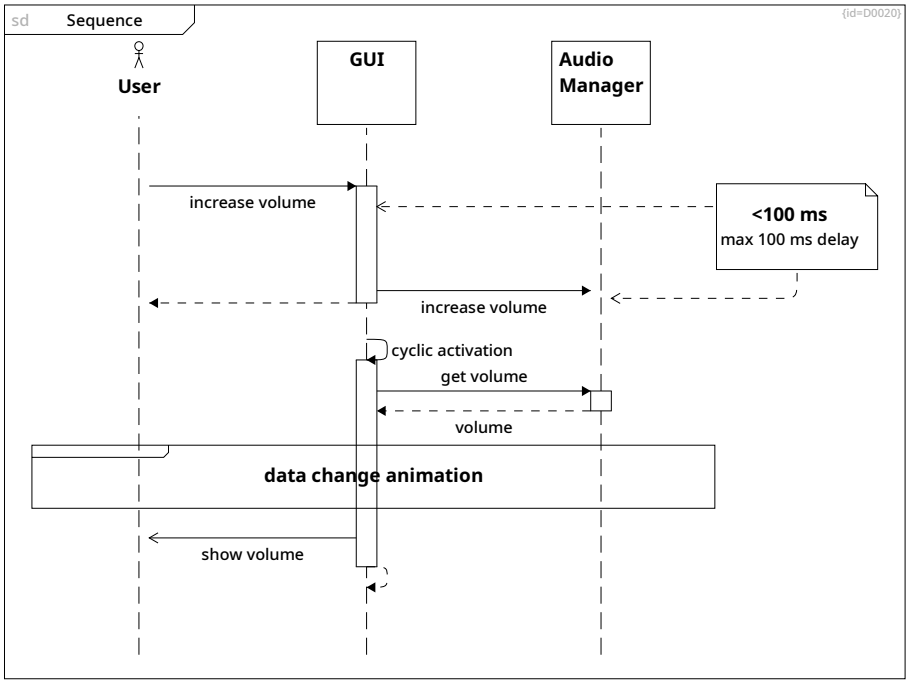
This section lists what kind of elements crystal-facet-uml can draw in diagrams.

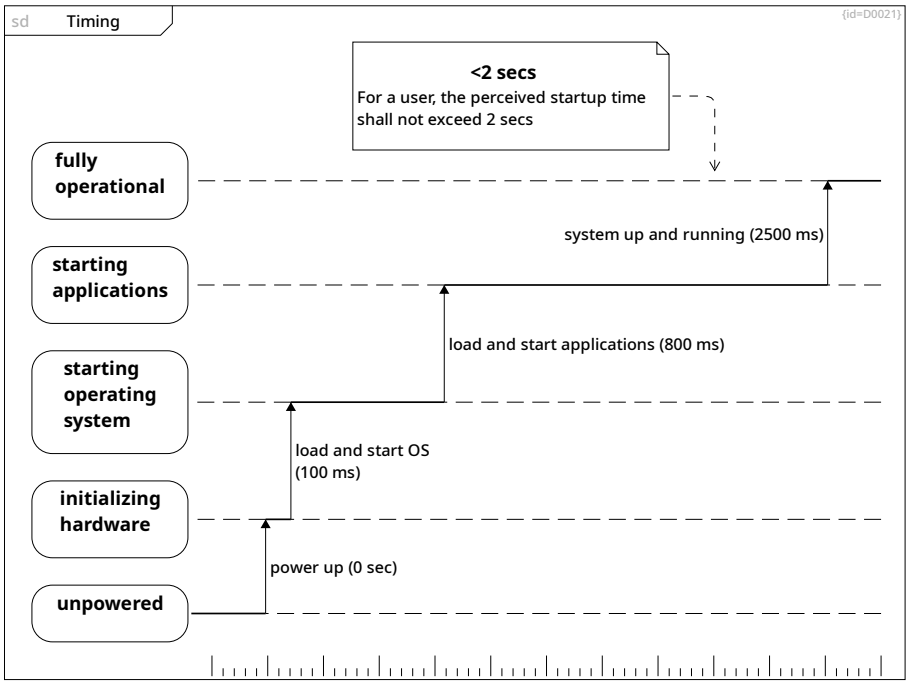


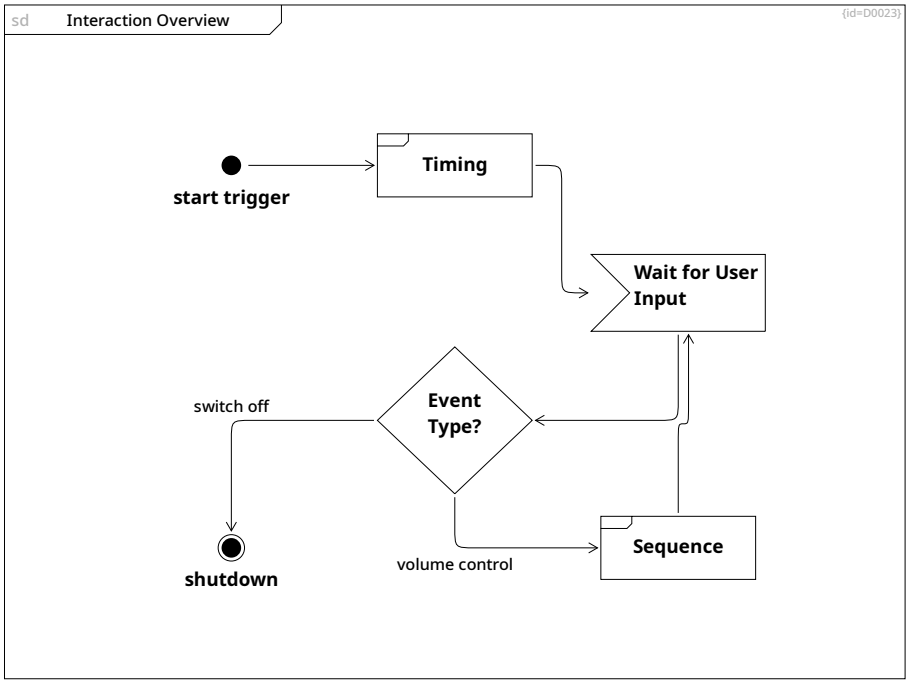

act Activities
{id=D0008}
read xml
decoder::
start
select
file to
parse
success
read
element
and
attributes
tag end
read
CDATA
contents
open
read
xml
header
content end
data
element
start
next
token
tag end
error
check
validity
element
end
read end
tag
end of file
decoder::
end


stm System States
{id=D0012}
pwr::start
booting
running
trigger shutdown
diag mode
normal
boot
start session
end session
shutting
down
sw-update
diagnosis
update finished
trigger sw-
update
SW-update
pwr::off


sd Communication Diagram
{id=D0013}
External Entity
1. connect
TLS Endpoint
2. fetch own
public key
3. return key
4. store generated session key
Asymmetric
Signature
Storage
Symmetric
Session Keys
5. ack

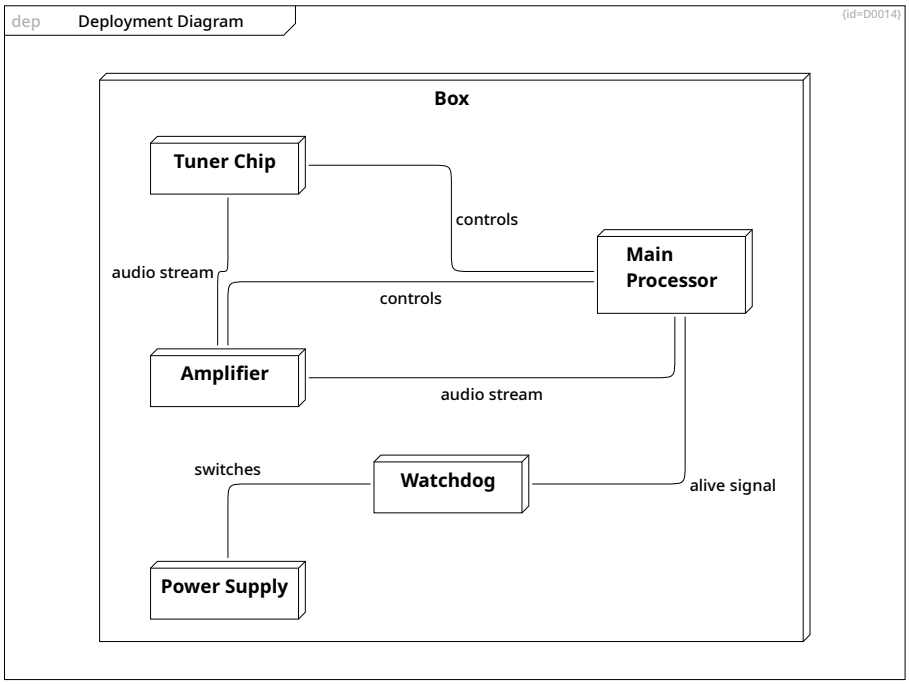

sd Sequence
{id=D0020}
User
GUI
Audio
Manager
increase volume
<100 ms
max 100 ms delay
increase volume
cyclic activation
get volume
volume
data change animation
show volume

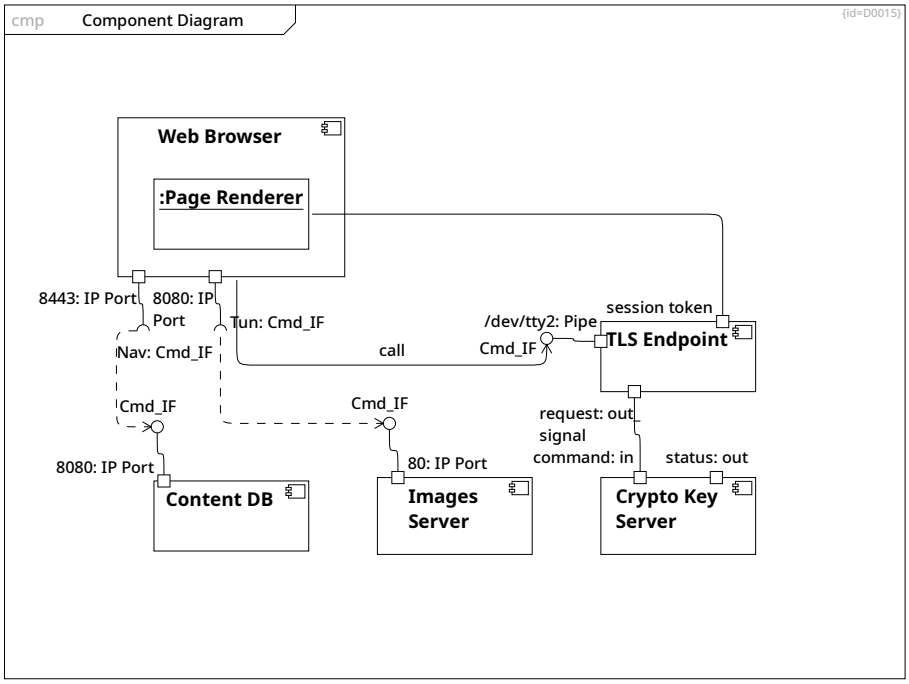


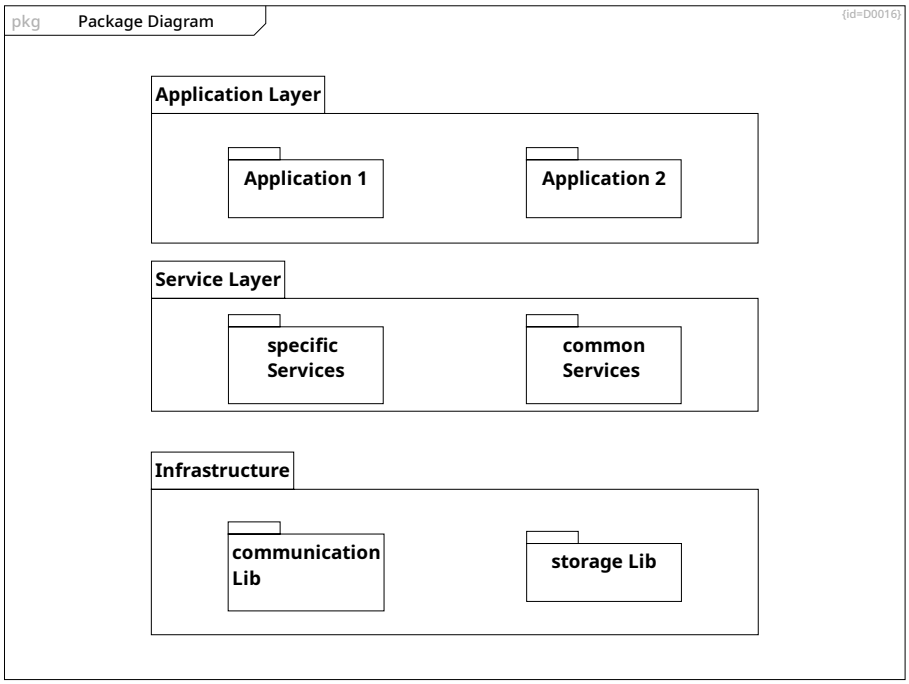


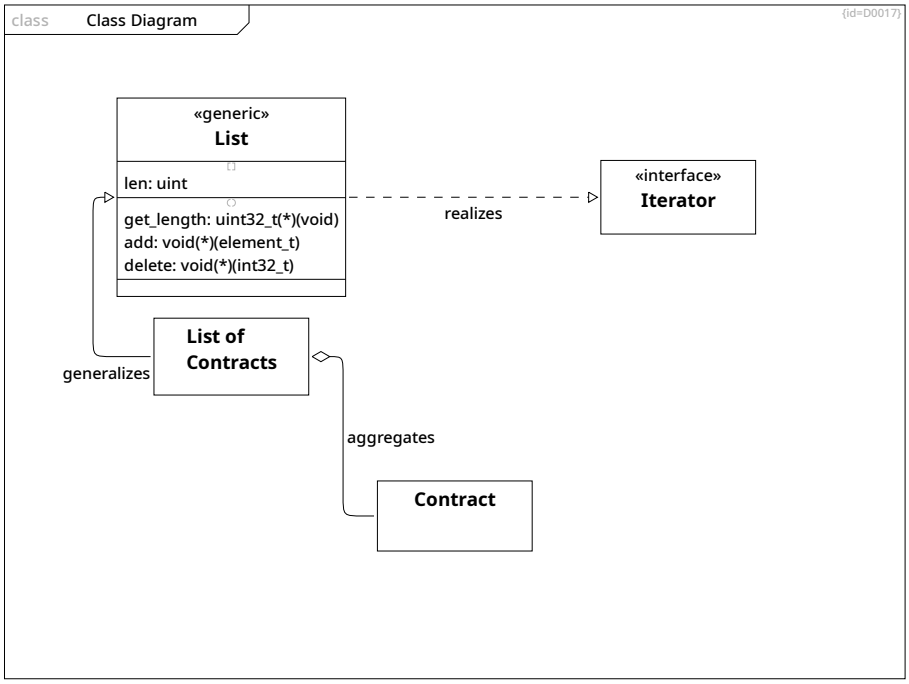
2.3 Example UML Static Views

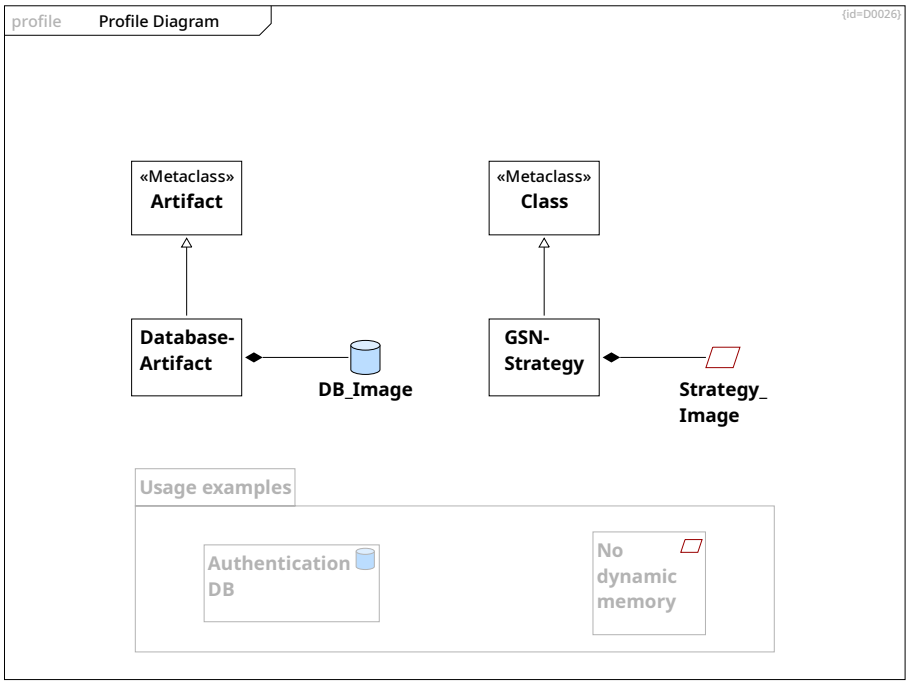
This section lists what kind of elements crystal-facet-uml can draw in diagrams.


dep
Deployment Diagram
{id=D0014}
Box
Tuner Chip
controls
audio stream
Main Processor
controls
Amplifier
audio stream
switches
Watchdog
alive signal
Power Supply


cmp
Component Diagram
{id=D0015}
Web Browser
:Page Renderer
8443: IP Port
8080: IP Port
Tun: Cmd_IF
session token
/dev/tty2: Pipe
TLS Endpoint
Nav: Cmd_IF
call
Cmd_IF
Cmd_IF
Cmd_IF
request: out
signal
command: in
status: out
8080: IP Port
80: IP Port
Content DB
Images Server
Crypto Key Server

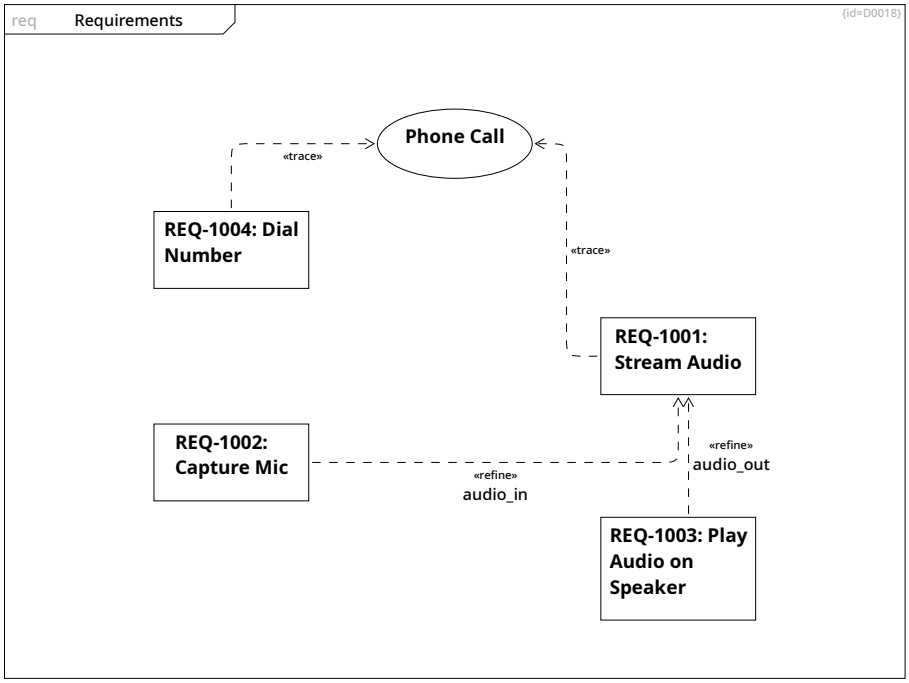

pkg
Package Diagram
{id=D0016}
Application Layer
Application 1
Application 2
Service Layer
specific Services
common Services
Infrastructure
communication Lib
storage Lib

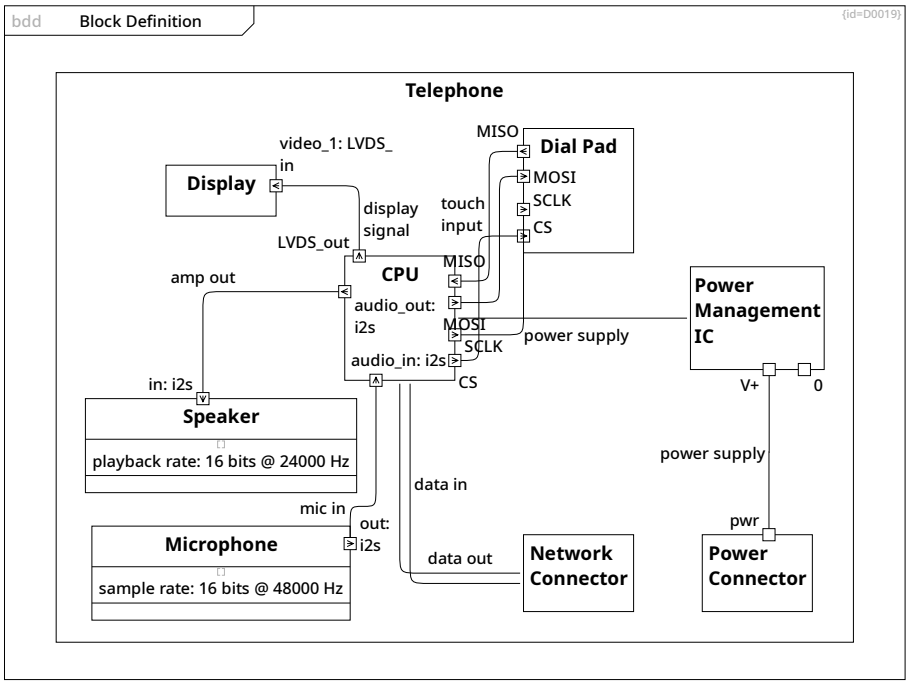

class
Class Diagram
{id=D0017}
«generic»
List
len: uint
get_length: uint32_t(*)(void)
add: void(*)(element_t)
delete: void(*)(int32_t)
realizes
«interface»
Iterator
List of Contracts
generalizes
aggregates
Contract

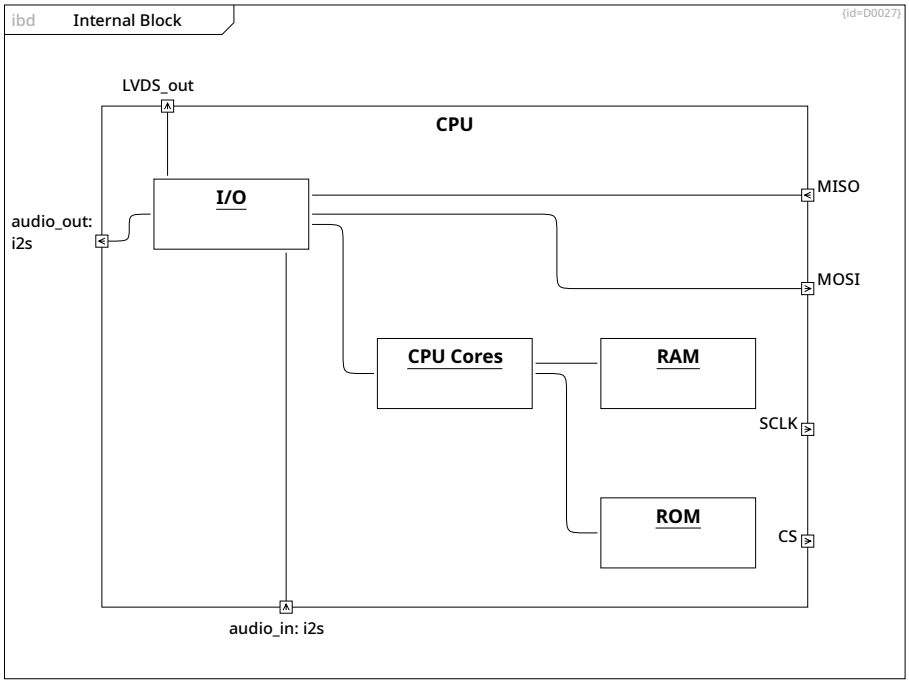


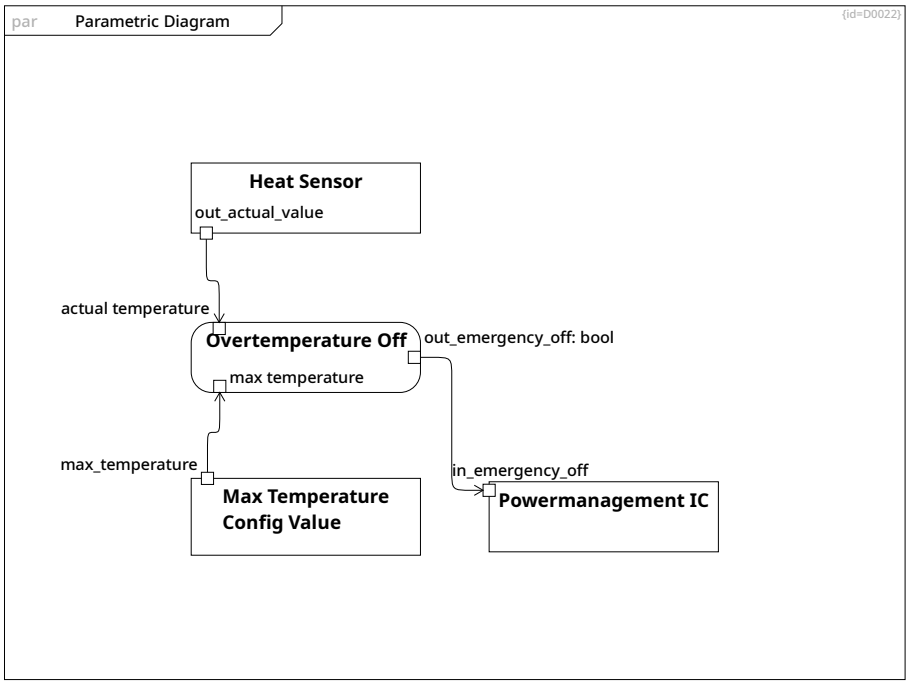
2.4 Example SysML Views

This section lists what kind of elements crystal-facet-uml can draw in diagrams.




bdd Block Definition
{id=D0019}
Telephone
video_1: LVDS_
in
Display
display
signal
LVDS_out
MISO
Dial Pad
MOSI
SCLK
touch
input
CS
CPU
MISO
amp out
audio_out:
i2s
MOSI
SCLK
audio_in: i2s
CS
power supply
Power
Management
IC
V+
0
in: i2s
Speaker
playback rate: 16 bits @ 24000 Hz
power supply
data in
mic in
out:
i2s
Microphone
sample rate: 16 bits @ 48000 Hz
data out
Network
Connector
pwr
Power
Connector


ibd Internal Block
{id=D0027}
LVDS_out
CPU
I/O
MISO
audio_out:
i2s
MOSI
CPU Cores
RAM
SCLK
ROM
CS
audio_in: i2s



2.5 More Examples

There are further examples available as html/pdf:

- mouse_droid.html / mouse_droid.pdf
- self_architecture.html / self_architecture.pdf
- quality.html / quality.pdf

And in crystal-facet-uml json format:

- https://github.com/awarnke/crystal-facet-uml/tree/master/documentation/examples
- https://github.com/awarnke/crystal-facet-uml/tree/master/documentation/architecture

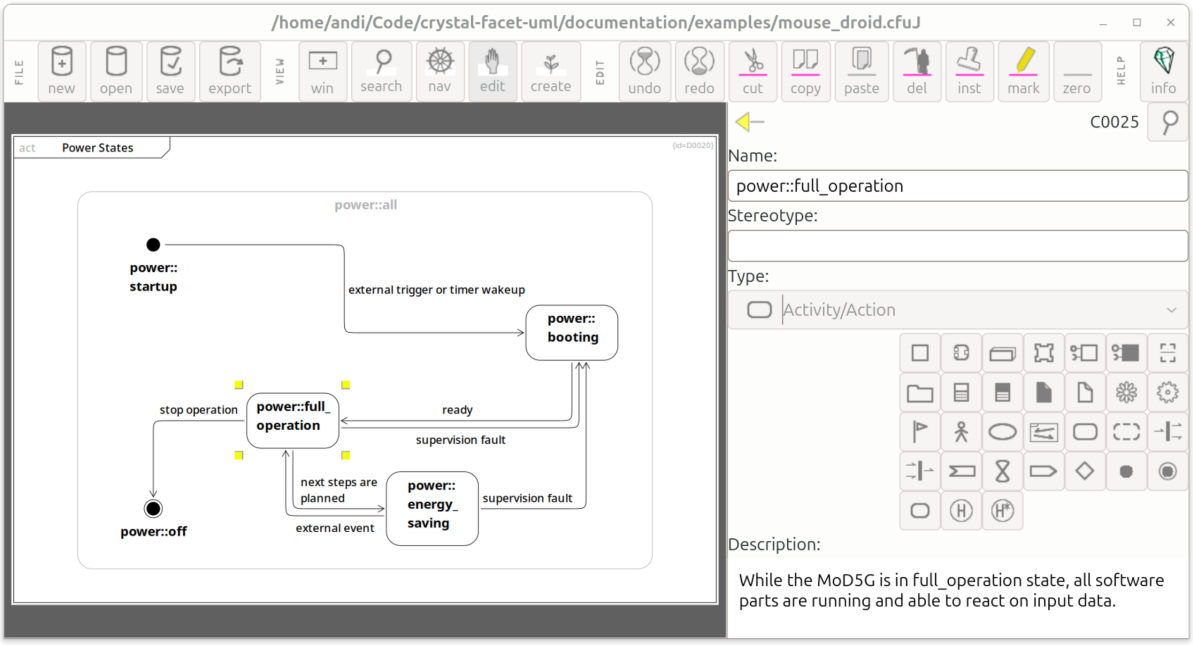
3 Graphical User Interface

Start the application by a click on the application icon.

3.1 Window Area Overview

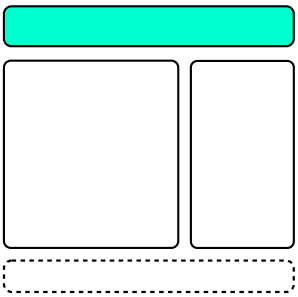
crystal-facet-uml shows a window with

- 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

The top row of a window provides buttons to select files, change the view and modify model elements.



3.2.1 New



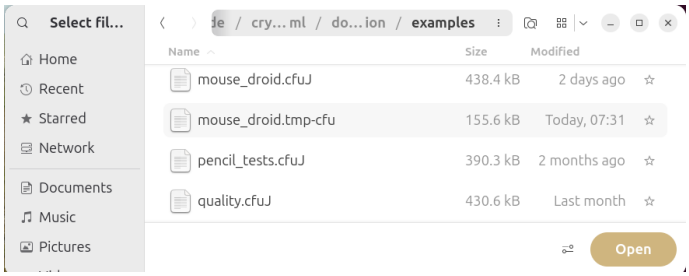
- Creates a new database file.
- Enter a filename; a json-based file structure is used to store your data in a git-friendly format.

3.2.2 Open



- Opens an existing database file.
- Note that write access to the parent folder is required.

- If you find a **.tmp-cfu** file, this indicates that the last session was possibly terminated abnormally. You should open this file to continue from the latest state.



  Alternatively, ensure that the file is not used concurrently, delete the **.tmp-cfu** file and select the **.cfu** file to continue from your last save action.

- Do not open a file twice at a time: While the file is open it cannot be shared. Close the program first.

3.2.3 Save



- Stores the latest changes to the database immediately. Note that at regular program exit, the database is stored automatically anyhow.
- The icon indicates if there are unsaved changes, it is yellow in case the window is in the background.

3.2.4 Export



- Exports all diagrams to the selected folder. Supported export formats are docbook, html, json, pdf, png, ps, svg, txt, xmi.

  Previously existing files in the selected folder are overwritten.

3.2.5 New Window



- Opens another window on the same database. This new window allows you to work reliably with multiple windows on the same database.

3.2.6 Search



- Find diagrams that contain the searched elements (see Section 3.3.1)

3.2.7 Navigate



- Navigate to parent or child diagrams
- Create a new diagram (see Section 3.3.2)

3.2.8 Edit



- Modify elements in the diagram (see Section 3.3.3)

3.2.9 Create



- Create elements in the diagram (see Section 3.3.4)

3.2.10 Undo



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

3.2.11 Redo



- Re-does the last un-done operation

3.2.12 Cut



- Cut all selected (pink-cornered) elements to the clipboard (features of classifiers are copied if the classifier is selected)

3.2.13 Copy



- Copy all selected (pink-cornered) elements to the clipboard (features of classifiers are copied if the classifier is selected)

3.2.14 Paste



- If the clipboard does not contain diagrams, classifiers and relationships from the clipboard are copied into the current diagram.
- If the clipboard contains a diagram, this diagram is pasted below the current diagram. All other elements are pasted into the new diagram.
- If a classifier is identical to an existing one (same uuid), an instance of the existing classifier is pasted to the diagram. Otherwise a new classifier is created.

3.2.15 Delete



- Deletes all selected (pink-cornered) elements.
- This operation may fail on a diagram if the selected diagram contains non-selected elements or child diagrams.

3.2.16 Instantiate



- Toggles the selected (pink-cornered) classifiers between classes, named instances and anonymous instances.
- No effect on relationships and features.

3.2.17 Mark



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

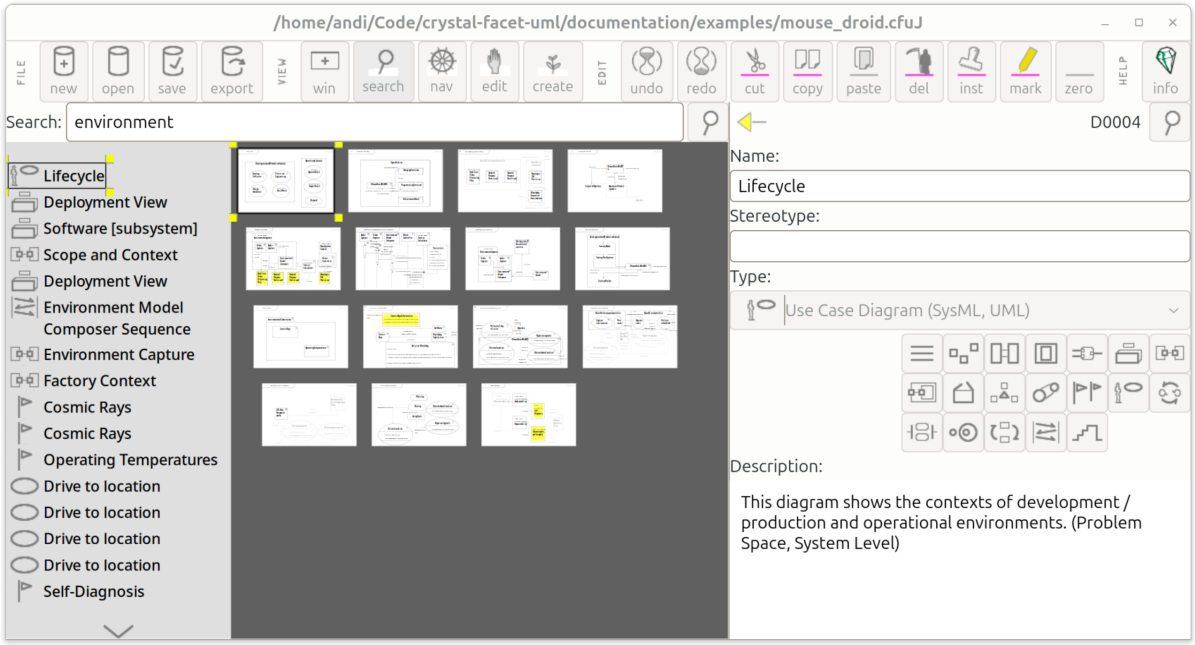
3.2.18 Reset Selection

- Resets the (pink-cornered) selection

3.2.19 About

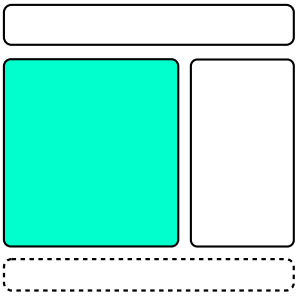


- Shows version, license and copyrights



3.3 Drawing Area

The center area of a window allows to view and change diagrams and model elements.



Diagrams are layouted automatically. You can influence the locations of classifiers only.

When adding too many classifiers or relationships, 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. For examples, see Section 5.1.3.

3.3.1 Search



- Enter the ID of an element (e.g. C0001) or a part of its name or description to find diagrams containing this element.

- Enter nothing to find diagrams containing elements without description.
-  Start a search to displays the results.

3.3.2 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.

3.3.3 Edit



- Click on a diagram or classifier or feature or relationship to edit the name, type and description of that object.

  The yellow corners indicate which object is currently focused.
- Click on an element to select or unselect an object (pink corners).

  The toolbar buttons apply to this pink-cornered set.
-  To move classifiers within the diagram, 1.) press, 2.) drag and 3.) release the mouse button.

  Note: When moving a classifier, this is moved in all diagrams where it appears. Order and locations of things stay consistent between different views.

It is not possible to change source and destination classifiers of relationships. Such structural changes would make it difficult to merge concurrent changes on multiple branches in git. Instead, delete the old and create a new relationship. Merging such changes from concurrently modified branches will not cause unexplainable results.

3.3.4 Create

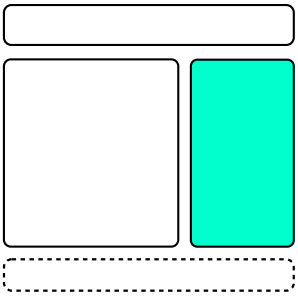


-  To create a classifier, click at an empty space in the diagram.
- To create a child classifier, click into the white space of the parent classifier. (Alternatively, create a classifier and a containment relationship.)
-  To create a feature, click onto a classifier border (not on the classifier name).
-  To create a relationship, press on the source classifier or feature and drag it to the destination classifier or feature.

To modify existing elements, switch back to edit mode: Section 3.3.3.

3.4 Element Configuration Area

The right side of a window shows the properties of the focused model element.



Edit the properties of the focused (yellow-cornered) object.

- name of the focused object
- stereotype of the focused object.

  Stereotype names shall consist of characters that are valid XML tokens (Nmtoken).

  Multiple stereotypes shall be separated by comma.

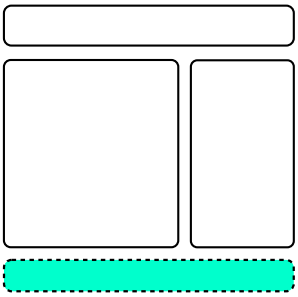
  Rendering images of stereotypes only works for single stereotypes.

  In case of properties and operations enter the type of the property or operation arguments, in case of tagged values, enter the value (instead of a stereotype).
- type of the focused object
- description of the focused object.

  For html and DocBook export, use a double linebreak to create a new paragraph, start lines with *, + or - to format a list, use D0001#id and D0001#name to create a link to the diagram D0001 (showing either the id or the name).

3.5 Notification Bar

The bottom row of a window, if shown, displays the result of the last user interaction.



The notification bar appears automatically when there is a new message.

Such a message may show statistics on performed actions like created, exported, modified, deleted for the following elements

- diagrams,
- classifiers refer to the model-nodes,
- classifier-occurrences refer to the visualization of a classifier in a diagram,
- features denote properties, operations and ports of classifiers,
- relationships are the edges between the classifiers or features or lifelines,
- lifelines refer to implicit and automatically managed objects needed to visualize temporal behavior.

3.5.1 Information



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

3.5.2 Warning



- Informs on a possible problem, e.g. a read-only database file

3.5.3 Error



- Informs on an error, e.g. invalid data pasted from clipboard

4 Diagrams and Elements Spec



This program creates diagrams that strive for compatibility to

- UML 2.5
- SysML 1.5
- MOF 1.4.1

In some cases, it deviates from these standards for several reasons:

- Reduce complexity to be able to handle such models in a small open source project
- Reduce feature-set to improve understandability of diagrams even to non-software-architects
- Reduce feature-set to enhance usability of the program

This section gives an overview on standards and implementation-status of crystal-facet-uml. It may be incomplete.

4.1 Classifiers

Classifiers are the nodes in the model-graph.

The table shows the classifier types introduced by different specifications and a comment stating how this is implemented in crystal-facet-uml.

	Spec/Context	Comment
 Block	SysML	Limitations: no Multiplicities of Block-Instances.
 Constraint Block	SysML / Parametric	Limitations: Only the rounded-rect symbol is supported.
 Node	UML / Deployment	
 Subsystem/Boundary	UML / Use Case	A subsystem is a component with stereotype subsystem
 Component	UML	
 Part	UML	
 Interface	UML	
 Package	UML, SysML	
 Class	UML	Limitations: No active classes
 Object	UML	
 Artifact	UML	
 Comment	UML, SysML	
 Image	UML, SysML	This element shows the image of its stereotype. It exists since version 1.48.0; cannot be used in version 1.47.0 and older. The description field is reserved. Use an additional comment if needed.
 Stereotype	UML, SysML	This element declares a stereotype. It exists since version 1.47.0; cannot be used in version 1.46.0 and older. The description field is reserved for icon and template data. Use an additional comment to describe the stereotype. An image can be specified by stating lines and curves according to the SVG-path spec version 2.0. **<path fill="#e0e0e0" stroke="#0088cc" d="M 2,2 10,2 8,8, 0,8 Z"/>** See examples at Section 4.6
 Requirement	SysML	
 Actor	UML, SysML / Use Case, Sequence	
 Use Case	UML, SysML / Use Case	Limitations: No SysML extension points
 Interaction Use	UML / Interaction Overview	Hint: To easily find the referenced diagram, name the interaction use identical to the diagram. XMI-Export: For xmi export, this object may only occur in interaction diagrams: interaction overview, communication, sequence and timing.
 Activity/Action	UML 2.5 (ch15.2) / Activity	

	Spec/Context	Comment
 Interruptable Region	UML / Activity	XMI-Export: For xmi export, all regions belonging to the same set of activities need an outer, enclosing activity.
 Fork	UML, SysML / Activity	XMI-Export: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
 Join	UML, SysML / Activity	XMI-Export: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
 Accept Event	UML, SysML / Activity	XMI-Export: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
 Accept Time Event	UML, SysML / Activity	XMI-Export: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
 Send Signal	UML, SysML / Activity	XMI-Export: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
 Decision/Choice	UML 2.5 (ch14.2.4,15.3), SysML / Activity, State	In activity diagrams, this is called decision, in statesmachines it is called choice. XMI-Export/State-context: For xmi export, all states belonging to the same statemachine need an outer, enclosing state. XMI-Export/Activity-context: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
 Initial Node	UML 2.5 (ch14.2.4), SysML / Activity, State	Limitations: There is no distinction in ActivityInitial and FlowInitial. XMI-Export/State-context: For xmi export, all states belonging to the same statemachine need an outer, enclosing state. XMI-Export/Activity-context: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
 Final Node	UML 2.5 (ch14.2.4), SysML / Activity, State	Limitations: There is no distinction in ActivityFinal and FlowFinal. Limitations: There is no separate terminate state-type. XMI-Export/State-context: For xmi export, all states belonging to the same statemachine need an outer, enclosing state. XMI-Export/Activity-context: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
 State	UML 2.5 (ch14.2), SysML / State, Timing	Limitations: No symbol for hidden decompositions, no regions (swimlanes) in composite states. Limitations: entry/exit/do list. XMI-Export: For xmi export, all states belonging to the same statemachine need an outer, enclosing state.
 Shallow History	UML 2.5 (ch14.2.4), SysML / State	XMI-Export: For xmi export, all states belonging to the same statemachine need an outer, enclosing state.
 Deep History	UML 2.5 (ch14.2.4), SysML / State	XMI-Export: For xmi export, all states belonging to the same statemachine need an outer, enclosing state.

	Spec/Context	Comment
× Value Type	SysML	not supported. Limitations: Compartment Order of Classifiers is "properties, operations" instead of "operations, properties, stereotype-tagged-values"
× Enumeration	UML, SysML	not supported. Note: Use a class instead.
× ActivityParameterNode	SysML	not supported.
× MergeNode/Junction	UML 2.5 (ch15.3), SysML / Activity, State	In activity diagrams, it is called merge, in state diagrams junction node. This is not supported. Note: You may directly connect the arrows to the target activity/state.
× ActivityPartition	UML, SysML / Activity	not supported. Note: Use a parent activity instead.

An InstanceSpecification (UML) denotes an instantiation of a classifier. crystal-facet-uml allows any classifier to appear in different diagrams as classifier, as anonymous InstanceSpecification or as named InstanceSpecification. (Rationale: If a classifier is an instance may depend on the context: An M1-class may be an instance if shown in an M2-meta-class diagram, an XML-parser-class may be an instance if shown in the context of stream processors.)

4.2 Features

Features are elements attached to one classifier.

The table shows the feature types introduced by different specifications, if they are visible in any diagram or just once, and a comment stating how this is implemented in crystal-facet-uml.

	Spec	Comment
Property	UML, SysML	Limitations: no SysML Flow-Properties refinement.
Operation	UML, SysML	
Provided Interface	UML, SysML	
Required Interface	UML, SysML	
Port	UML, SysML	Limitations: no SysML-compartment Notation supported. Limitations: no SysML-nested-ports, SysML-proxy-port, SysML full-ports supported. Limitations: no flow property, no compartment notation, no port-compartments. Limitations: no UML behavior ports.
Input Port/Pin	UML, SysML	
Output Port/Pin	UML, SysML	
State Entry	UML, SysML	
State Exit	UML, SysML	
Tagged Value	UML, SysML	Exists since version 1.47.0; cannot be used in version 1.46.0 and older.

4.3 Lifelines

Lifelines are attached to one classifier and they appear in exactly one interaction diagram. When a classifier appears in multiple interaction diagrams, it has the same amount of lifelines attached.

crystal-facet-uml stores lifelines in the same data structure as features.

	Spec	Comment
 Lifeline	UML 2.5 (ch17.2), SysML	Lifelines are managed automatically, one cannot create, modify or delete them. Limitations: One lifeline is visible only in one diagram. Limitations: Lifelines start and end only at diagram border.

4.4 Relationships

Relationships are the edges of the model-graph.

The table shows the relationship types introduced by different specifications, a classification in which diagram type to use them preferably, and a comment stating how this is implemented in crystal-facet-uml.

	Spec/Context	Comment
 Dependency	UML, SysML	
 Containment	UML, SysML / Deployment, Package, Internal Block, Composite Structure, Activity, State	
 Deploy	UML / Deployment	
 Manifest	UML / Deployment	
Communication Path	UML, SysML / Component, Composite Structure, Block, Internal Block	
 Association	UML, SysML / Class, Use Case	Note: SysML calls this ReferenceAssociation Limitations: no AssociationClass(SysML: ParticipantProperty) exists. Limitations: no AssociationEnd Classes exist, no multiplicities, no roles, no ownership (dot notation). Limitations: no ternary associations (only two ends supported). Limitations: no non-navigateable ends (crosses) suported yet - see todo.txt.
 Aggregation	UML, SysML / Class	Note: SysML calls this SharedAssociation
 Composition	UML, SysML / Class	Note: SysML calls this PartAssociation
 Generalization	UML, SysML / Class	Limitations: no Generalization-Sets supported
 Realization	UML / Class	

	Spec/Context	Comment
Trace	SysML / Requirement	
Refine	SysML / Requirement	
Extend	UML, SysML / Use Case	Limitations: no SysML-condition-notes can be attched to this relationship
Include	UML, SysML / Use Case	
Control Flow/Transition	UML, SysML / Activity, State	In activity diagrams, this is called control flow, in statesmachines it is called transition.
Object Flow	UML, SysML / Activity	
Async. Call	UML, SysML (?) / Sequence, Timing, Communication, Interaction overview	
Sync. Call	UML, SysML (?) / Sequence, Timing, Communication, Interaction overview	
Return Call	UML, SysML (?) / Sequence, Timing, Communication, Interaction overview	In sequence diagrams, an activation bar (execution specification) is shown above return calls
Connector	UML, SysML / Internal Block	not supported. Limitations: No Bi-directional Connectors Note: SysML calls this BindingConnector Note: Use a Communication Path instead.
Item Flow	SysML / Block Definition	not supported. Note: Use an Object Flow instead.
Exception Flow	UML 2.5 (ch15.5) / Block Definition	Use a stereotype, e.g. a flow_exception as proposed in Section 4.6

4.5 Diagrams

Diagrams are views on the model-graph. They select classifiers and may filter their features and relationships.

The table shows the diagram types introduced by different specifications and a comment stating how this is implemented in crystal-facet-uml.

	Spec	Comment
List Diagram	-	This is an overview diagram showing only classifiers as a list. `hides` features, relationships
Box Overview Diagram	-	This is an overview diagram showing only 2-dimension-layouted classifiers. `hides` features, relationships
Block Definition Diagram	SysML	
Internal Block Diagram	SysML	
Parametric Diagram	SysML	

	Spec	Comment
 Deployment Diagram	UML	
 Component Diagram	UML	
 Composite Structure Diagram	UML	
 Package Diagram	UML, SysML	
 Class Diagram	UML	
 Profile Diagram	UML	
 Requirements Diagram	SysML	
 Use Case Diagram	UML, SysML	
 Interaction Overview Diagram	UML	Limitations: There is no link from Diagram-References to referenced Diagrams Containments cannot be shown in this diagram type `hides` features (except lifelines) `hides` relationships (except messages between lifelines)
 Activity Diagram	UML 2.5 (ch15.2), SysML	Limitations: Swimlanes not supported
 State Machine Diagram	UML, SysML	
 Communication Diagram	UML	Containments cannot be shown in this diagram type `hides` features (except lifelines) `hides` relationships (except messages between lifelines)
 Sequence Diagram	UML, SysML	`hides` features (except lifelines) `hides` relationships (except messages between lifelines)
 Timing Diagram	UML	`hides` features (except lifelines) `hides` relationships (except messages between lifelines)

4.6 Example stereotype images

For use as stereotype images, this section shows some generated svg-paths, licensed under Apache-2.0 or Public Domain at your choice.

Copy these xml-fragments to the description field of a stereotype-classifier in order to show these images next to all elements implemening the respective stereotype.

STEREOTYPE IMAGES

deploy_database



<path d="m 4,5 l 0,22 c 0,2.25 5.25,4 12,4 s 12,-1.75 12,-4 l 0,-22 " /> <path d="m 4,5 c 0,-2.1875 5.375,-4 12,-4 s 12,1.8125 12,4 s -5.375,4 -12,4 s -12,-1.8125 -12,-4 " />

deploy_local



<path d="m 1,23 l 16,8 l 8,-6 l 3,-11 l -16,-7 l -2,11 z " /> <path stroke="none" fill="#cccccc" d="m 5,23 l 11.5,5.5 l 5,-3.5 l -11,-5.1875 z " /> <path stroke="none" fill="#0000dd" d="m 11.5,17 l 12.5,6 l 2,-8 l -12.8125,-6 z " />

deploy_cloud



<path d="m 3,22 c -3,-3 3,-9 6,-6 c 0,-5 8,-6 9,-1 c 7,-2 14,4 11,7 z " />

ecb_entity



<path d="M 6,18 C 6,11.375 11.375,6 18,6 S 30,11.375 30,18 S 24.625,30 18,30 S 6,24.625 6,18 " /> <path d="M 7,30.5 L 29,30.5 " />

ecb_control



<path d="M 6,18 C 6,11.375 11.375,6 18,6 S 30,11.375 30,18 S 24.625,30 18,30 S 6,24.625 6,18 " /> <path d="M 22,1 L 17,6 L 22,11 " />

ecb_boundary



<path d="M 6,18 C 6,11.375 11.375,6 18,6 S 30,11.375 30,18 S 24.625,30 18,30 S 6,24.625 6,18 " /> <path d="M 1,9 L 1,27 M 1,18 L 6,18 " />

gsn_goal



<path stroke="#000099" d="M 1,7 L 31,7 L 31,25 L 1,25 Z " />

gsn_context



<path stroke="#000099" d="M 8,25 C 0,25 0,7 8,7 L 24,7 C 32,7 32,25 24,25 Z " />

gsn_strategy



<path stroke="#000099" d="M 7,7 L 31,7 L 25,25 L 1,25 Z " />

gsn_assumption



<path stroke="#000099" d="M 1,10 C 1,5.625 7.75,2 16,2 S 31,5.625 31,10 S 24.25,18 16,18 S 1,14.375 1,10 " /> <path stroke="#000099" d="M 24,25 L 26.5,19 L 29,25 M 24.75,23.5 L 28.25,23.5 " />

gsn_justification



<path stroke="#000099" d="M 1,10 C 1,5.625 7.75,2 16,2 S 31,5.625 31,10 S 24.25,18 16,18 S 1,14.375 1,10 " /> <path stroke="#000099" d="M 25,24 C 25,26 29,26 29,24 L 29,19 L 25,19 " />

gsn_solution



<path stroke="#000099" d="M 1,16 C 1,7.75 7.75,1 16,1 S 31,7.75 31,16 S 24.25,31 16,31 S 1,24.25 1,16 " />

queue_buffer



<path d="M 1,8 L 31,8 L 31,24 L 1,24 " /> <path d="M 13,8 L 13,24 M 19,8 L 19,24 M 25,8 L 25,24 " />

queue_server



<path d="M 1,16 C 1,7.75 7.75,1 16,1 S 31,7.75 31,16 S 24.25,31 16,31 S 1,24.25 1,16 " />

queue_queue



<path d="M 1,11 L 18.5,11 L 18.5,21 L 1,21 " /> <path d="M 8,11 L 8,21 M 11.5,11 L 11.5,21 M 15,11 L 15,21 " /> <path d="M 21,16 C 21,13.25 23.25,11 26,11 S 31,13.25 31,16 S 28.75,21 26,21 S 21,18.75 21,16 " />

reason_decision



<path d="m 8,12 l -4,7 1,1 6,0 1,-1 -4,-7 l 8,-4 9,0 l -4,7 1,1 6,0 1,-1 -4,-7 " /> <path d="m 15,5 l 1,3 m 0,2 l 0,17 " />

reason_chosen



<path d="m 1,24 4,4 22,0 4,-4 " /> <path stroke="#00aa00" d="m 8,17 c 0,-4.4375 3.5625,-8 8,-8 s 8,3.5625 8,8 s -3.5625,8 -8,8 s -8,-3.5625 -8,-8 " /> <path stroke="#00aa00" d="m 11,17 l 10,0 m -5,-5 l 0,10 " />

reason_rejected



<path d="m 1,24 4,4 22,0 4,-4 " /> <path stroke="#cc0000" d="m 8,17 c 0,-4.4375 3.5625,-8 8,-8 s 8,3.5625 8,8 s -3.5625,8 -8,8 s -8,-3.5625 -8,-8 " /> <path stroke="#cc0000" d="m 11,17 l 10,0 " />

flow_object



<path d="M 32,32 M 0,0 M 5,5 27,5 27,27 5,27 Z " />

flow_control



<path d="M 1,16 l 2,0 m 3,0 l 2,0 m 3,0 l 2,0 m 3,0 l 2,0 m 3,0 l 2,0 m 3,0 l 3.5,0 M 25,13 l 5,3 l -5,3 " />

flow_exception



<path d="M 1,9 27,9 1,23 30,23 M 26,20 31,23 26,26 " />

landmark_top_left

<path stroke="none" fill="#ddccee" d="M 31,6 L 31,1 L 5,1 C 2.75,1 1,2.75 1,5 L 1,31 L 6,31 L 6,6 Z " /> <path stroke="none" fill="#ddccee" d="M 31,11 L 31,7 L 7,7 L 7,31 L 11,31 L 11,15 C 9,13.5 8.5,11.5 10,10 C 11.5,8.5 13.5,9 15,11 Z " />

landmark_top_right

<path stroke="none" fill="#ddccee" d="M 26,31 L 31,31 L 31,5 C 31,2.75 29.25,1 27,1 L 1,1 L 1,6 L 26,6 Z " /> <path stroke="none" fill="#ddccee" d="M 21,31 L 25,31 L 25,7 L 1,7 L 1,11 L 17,11 C 18.5,9 20.5,8.5 22,10 C 23.5,11.5 23,13.5 21,15 Z " />

landmark_bottom_left

<path stroke="none" fill="#ddccee" d="M 31,26 L 31,31 L 5,31 C 2.75,31 1,29.25 1,27 L 1,1 L 6,1 L 6,26 Z " /> <path stroke="none" fill="#ddccee" d="M 31,21 L 31,25 L 7,25 L 7,1 L 11,1 L 11,17 C 9,18.5 8.5,20.5 10,22 C 11.5,23.5 13.5,23 15,21 Z " />

landmark_bottom_right

<path stroke="none" fill="#ddccee" d="M 26,1 L 31,1 L 31,27 C 31,29.25 29.25,31 27,31 L 1,31 L 1,26 L 26,26 Z " /> <path stroke="none" fill="#ddccee" d="M 21,1 L 25,1 L 25,25 L 1,25 L 1,21 L 17,21 C 18.5,23 20.5,23.5 22,22 C 23.5,20.5 23,18.5 21,17 Z " />

4.7 Maximum stringlengths

All strings (names, descriptions, stereotypes) have a maximum length.

Ascii characters require one, most other characters two or three 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_STEREOTYPE_LENGTH = 47,

- DATA_RELATIONSHIP_MAX_DESCRIPTION_LENGTH = 1023,

Diagrams:

- DATA_DIAGRAM_MAX_NAME_LENGTH = 47,
- DATA_DIAGRAM_MAX_STEREOTYPE_LENGTH = 47,
- DATA_DIAGRAM_MAX_DESCRIPTION_LENGTH = 8191,

In case the text your entered exceeds the string limit, you are warned that the string is truncated. Consider attaching a comment or a requirement element and move parts of the text there. Also attaching a tagged value and commenting this may allow to store a long description.

5 Modeling Guidelines

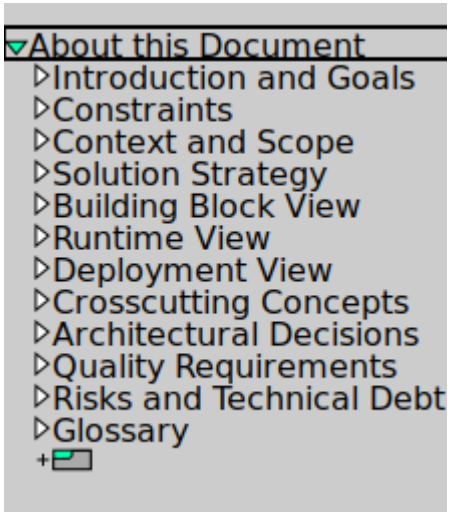
This section lists hints on efficiently using the tool crystal-facet-uml and provides some general remarks on creating a software architecture and design document.

5.1 crystal-facet-uml Hints

Modelling aspects that are special to the tool crystal-facet-uml are describes in this section.

5.1.1 Tree Structure

Diagrams are organized as a tree. The root diagram is the title page. This can be used to state the project goal or to explain 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/ :



5.1.2 Focus

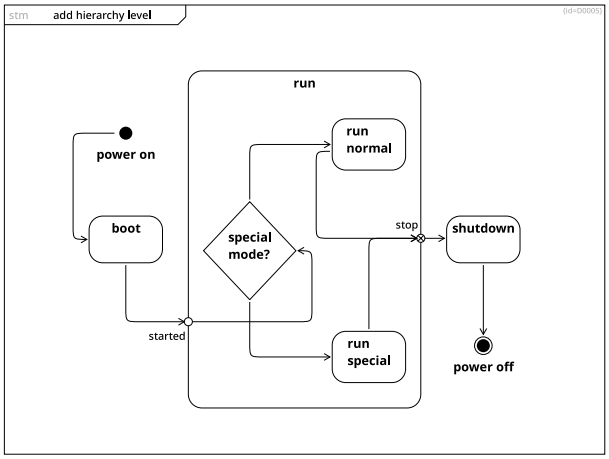
Put only few elements into each diagram. This increases understandability of the main purpuse of the diagram. Put further aspects of a topic into a separate diagram. Do not hesitate to copy an element from one diagram to the next. This is what crystal-facet-uml is good at: it keeps the model in sync.

5.1.3 Use Abstractions and Hierarchies

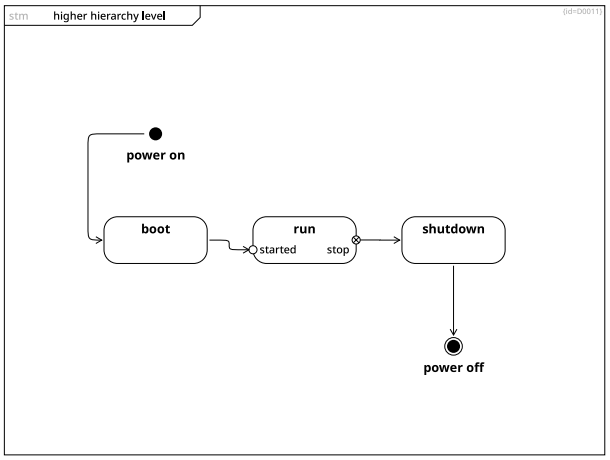
When distributing different aspects to different diagrams, a remaining challenge may be that there is no filter on features and relationships. If for example two classes are connected via a generalization arrow and an aggregation arrow, each diagram will show both arrows even if only one is of interest for the shown aspect (except for interaction diagrams).

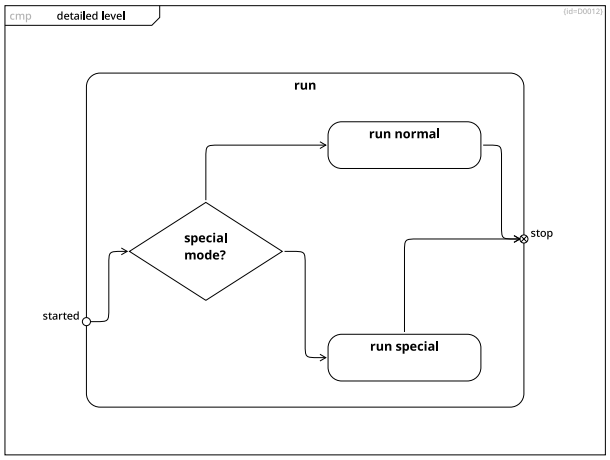
Solutions may be:

- Split by hierarchy level:

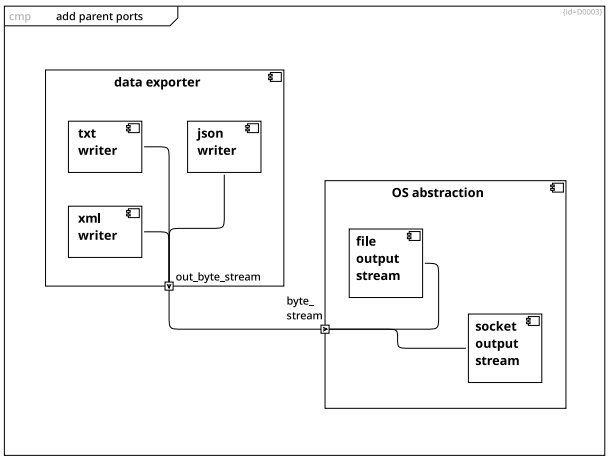


Add an hierarchy level, show only the outer elements in one diagram, show only the contained elements in another diagram.

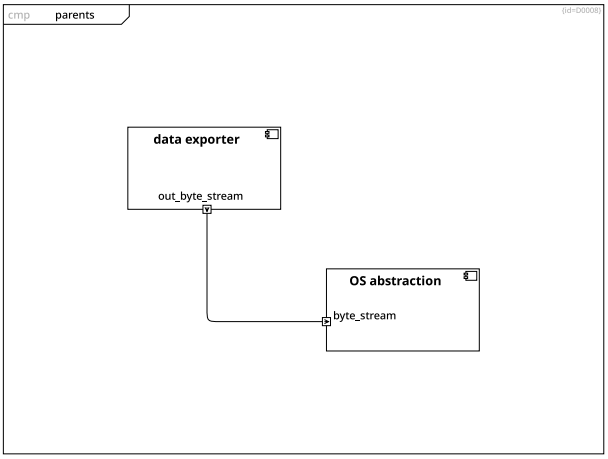


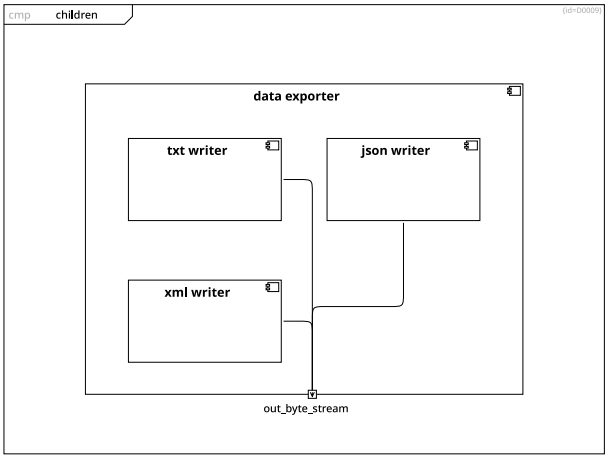


- Bundle communication paths:

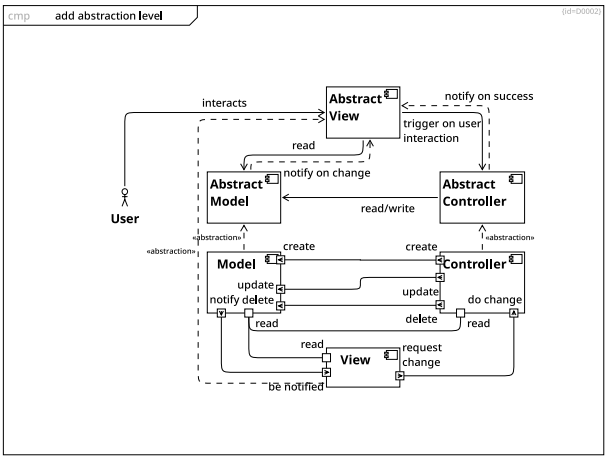


Define a port at a parent to bundle multiple communication paths of children.

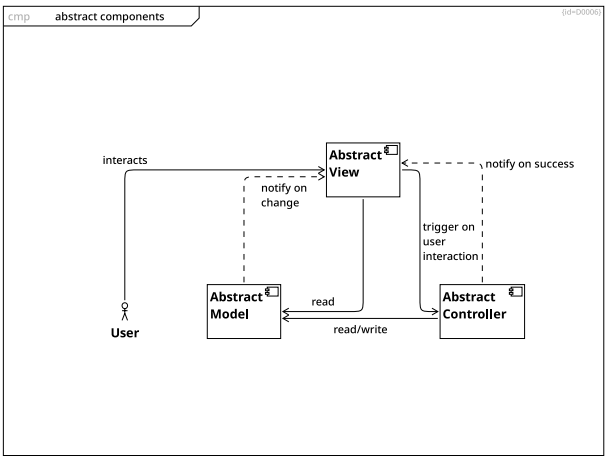


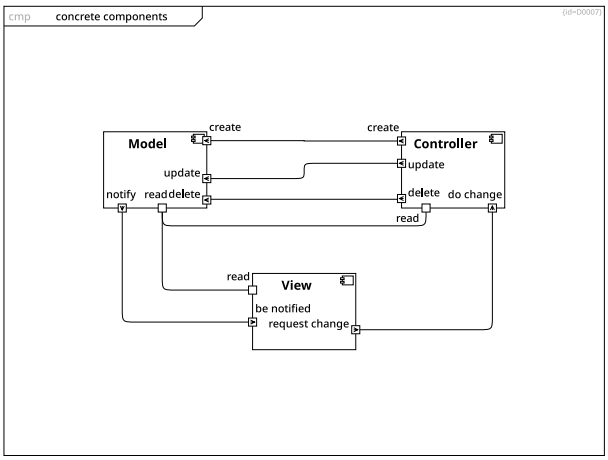


- Separate abstraction levels:

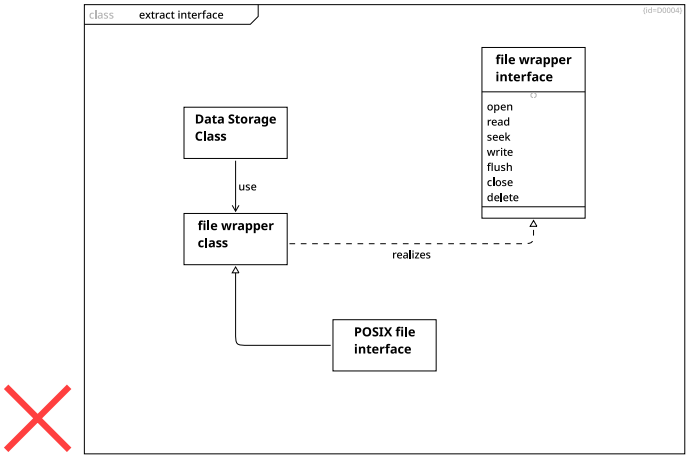


Separate abstract classes/components/blocks and their specializations to hide details when showing the abstract concepts and vice versa.

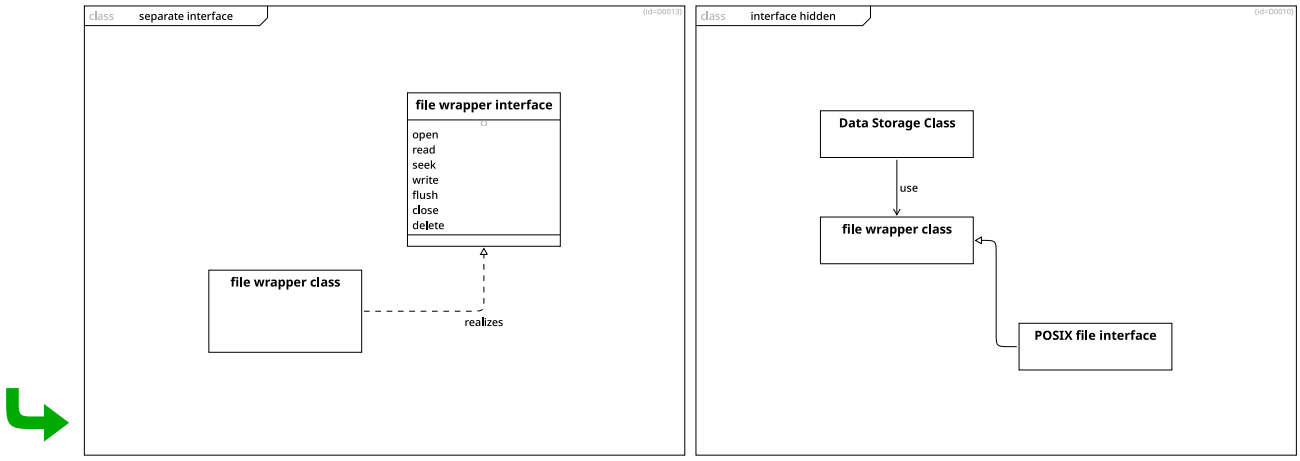




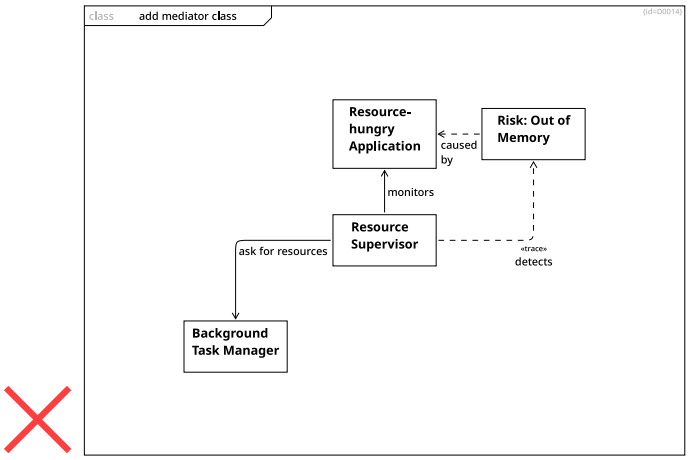
- Hide details:



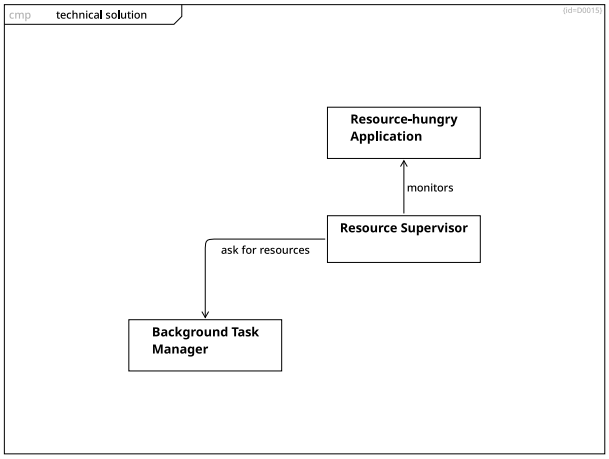
Extract methods to an interface to hide details when using the class.

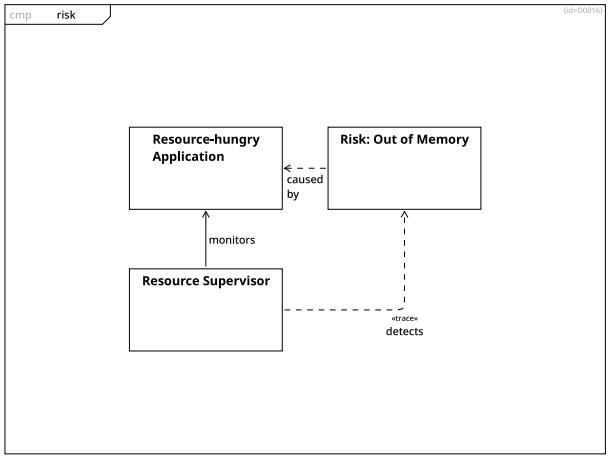


- Use relationships via a mediator:

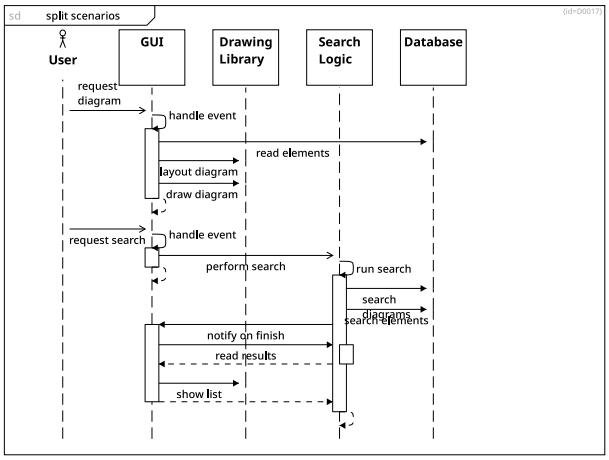


Add a topic-specific mediator class (e.g. a risk) that is only shown in selected diagrams.

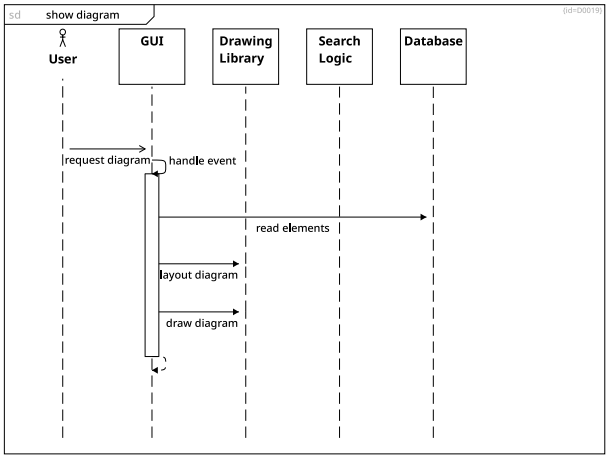


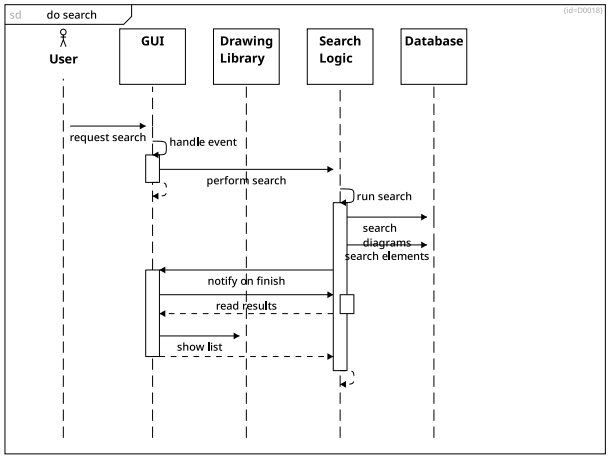


- Use interaction diagram:



If applicable, use an interaction diagram: Interaction-Overview, Communication, Sequence or Timing. These hide features and relationships except the ones defined at the local diagram, therefore they allow to focus on a single aspect of the interaction in each diagram.





5.1.4 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.

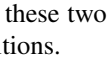
5.1.5 Attic/Storage room

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. Note that copy, in contrast to a cut, keeps all relationships.

5.1.6 Layout of Classifiers and Features

To change the positions of classifiers and features, drag these to the target location. Note that the relative position of classifiers towards each other affects the layout of other diagrams also.

5.1.7 Layout of Relationships

Relationships can only be dragged in sequence and timing diagrams. Relationships in other diagrams are auto-layouted. crystal-facet-uml prevents to cross/overlay these two types of relationships:  . If the layouting result is still inappropriate, move classifiers and features to other positions.

5.2 General Hints on Architecture Documentation

This section povides some general remarks on creating a software architecture and detailed design document.

5.2.1 Problem vs. Solution

Distinguish things that are

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

5.2.2 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.

5.2.3 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.

5.2.4 Precise sentences

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

5.2.5 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).

6 Command Line Interface

6.1 Command Line Usage

If starting crystal-facet-uml from command line, there are a couple of options, call **crystal-facet-uml -h** for a list.

To run consistency checks, use the **-t** option:

crystal-facet-uml -t my_database_file.cfu || echo "ERROR $?"

To repair the database, use the **-r** option.

To merge two model files, use the **-i** option.

crystal-facet-uml -i my_database_file_1.cfu add my_database_file_2.cfu"

To export a model, use the **-e** option.

crystal-facet-uml -e my_database_file.cfu html output_directory"

6.2 Storing Data in a Version Control System

The json file can be stored in a version control system (vcs) like svn or git.

- Close the crystal-facet-uml application before synchronizing a file with your vcs. Do not synchronize the file with your vcs while you modify it at the same time.
- In case of merge conflicts in the json file, note that uuid strings do uniquely identify all json objects. Relations between objects are defined by these uuid strings. In contrast to uuids, integer-id can be changed as long as they are unique among all objects of same type within the file. Also the names of classifiers (nodes) must be unique.
- To see the changes between versions of the json file, consider to use the patience option: **git diff --patience** The result may better explain the changes.

6.3 Command Line Usage of related Tools

Convert docbook to pdf using dblatex:

dblatex --type=pdf mouse_droid.xml

Convert docbook to pdf using pandoc:

pandoc -f docbook -t pdf --top-level-division=chapter -o mouse_droid.pdf mouse_droid.xml

Check if json/cfu files are valid:

check-jsonschema --schemafile myModel.schema myModel.json

7 Download and Install

This appendix shows where to get further documentation and how to install the software.

7.1 Documentation

User documentation is available here:

- https://andreaswarnke.de/crystal-facet-uml/crystal-facet-uml_documentation.pdf
- https://andreaswarnke.de/crystal-facet-uml/html/index.html

7.2 Debian/Ubuntu Linux

You may install crystal-facet-uml by the following command:

```
sudo apt install crystal-facet-uml
```

If you instead manually download the .deb archive, you may e.g. invoke **sudo dpkg --install <filename.deb>** . Find the latest executable version of crystal-facet-uml at:

- https://tracker.debian.org/pkg/crystal-facet-uml

7.3 SuSE Linux

To install, type on the command line:

```
sudo zypper addrepo https://download.opensuse.org/repositories/devel:/tools/ ↩
    openSUSE_Tumbleweed devel_tools_tumbleweed
# or sudo zypper addrepo https://download.opensuse.org/repositories/devel:/tools/16.0  ↩
    devel_tools_16.0
# for more repositories see https://download.opensuse.org/repositories/devel:/tools/
sudo zypper install crystal-facet-uml
```

If you instead manually download the .rpm archive, you may e.g. invoke **sudo zypper install <filename.rpm>** . Check for rpm packages at:

- https://build.opensuse.org/package/show/devel:tools/crystal_facet_uml

7.4 Windows

Find the latest executable version of crystal-facet-uml at one of the following addresses:

- https://www.heise.de/download/product/crystal-facet-uml/
- https://sourceforge.net/projects/crystal-facet-uml/

Unpack the zip archive.

- On windows, doubleclick on **crystal-facet-uml.bat**,

7.5 Build from Source

Find the latest source version of crystal-facet-uml at one of the following addresses:

- https://sourceforge.net/projects/crystal-facet-uml/
- https://github.com/awarnke/crystal-facet-uml

Follow the instructions in **/README.md**.

7.6 Further Links

Static code analysis results are available here:

- https://scan.coverity.com/projects/awarnke-crystal_facet_uml

Test coverage report is available here:

- https://andreaswarnke.de/crystal-facet-uml/test_coverage/index.html

Validate your XMI exports at:

- http://validator.omg.org/se-interop/tools/validator

7.7 License

License of crystal-facet-uml is Apache-2.0. Copyright 2016-2026 Andreas Warnke; Email-contact: cfu-at-andreaswarnke-dot-de