crystal-facet-uml documentation

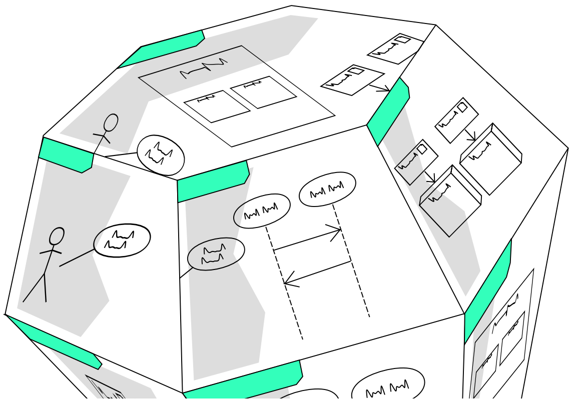
Andreas Warnke

COLLABORATORS			
	TITLE : crystal-facet-uml documentation		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Andreas Warnke	2022-10-01	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

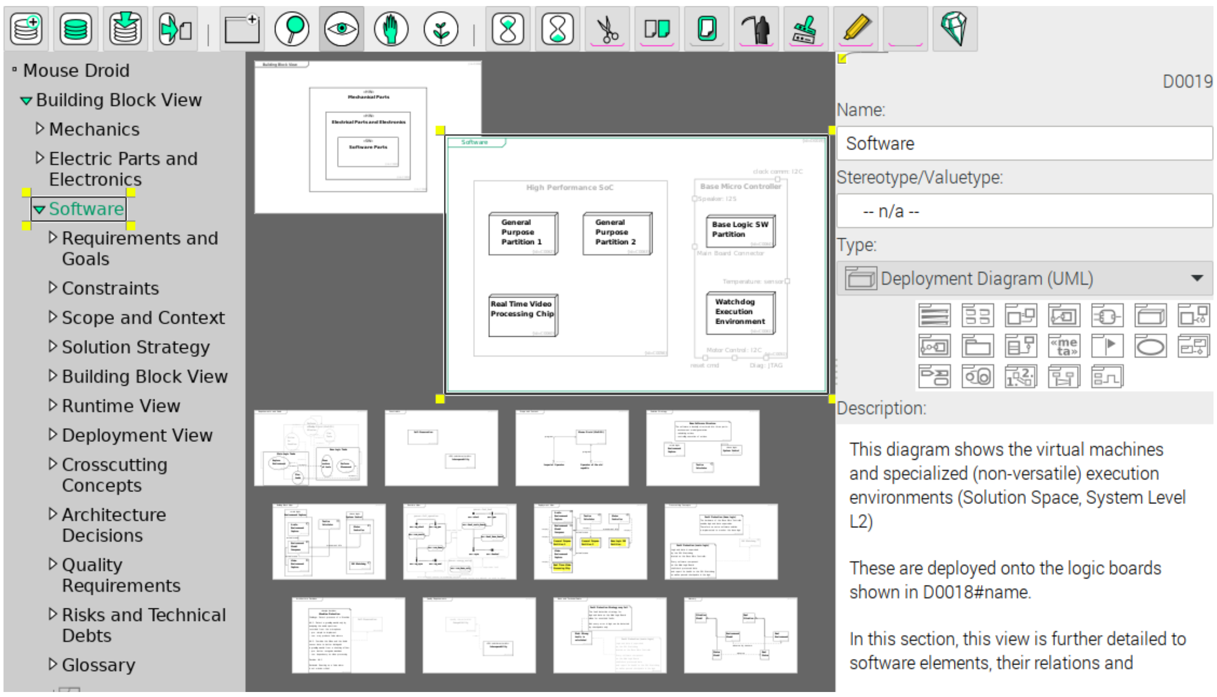
Contents



1 Introduction

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

Like a crystal shows different facets of the same thing, this application shows different views of the same system.



- Mouse Droid
  - Building Block View
    - Mechanics
    - Electric Parts and Electronics
    - Software
      - Requirements and Goals
      - Constraints
      - Scope and Context
      - Solution Strategy
      - Building Block View
      - Runtime View
      - Deployment View
      - Crosscutting Concepts
      - Architecture Decisions
      - Quality Requirements
      - Risks and Technical Debts
      - Glossary

D0019

Name: Software

Stereotype/Valuetype: -- n/a --

Type: Deployment Diagram (UML)

Description:

This diagram shows the virtual machines and specialized (non-versatile) execution environments (Solution Space, System Level L2)

These are deployed onto the logic boards shown in D0018#name.

In this section, this view is further detailed to software elements, their relations and

Software
High Performance SoC
General Purpose Partition 1
General Purpose Partition 2
Real Time Video Processing Chip
Base Micro Controller
Base Logic SW Partition
Watchdog Execution Environment

1.1 Goal

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

crystal-facet-uml keeps element names and element hierarchies consistent. It exports diagrams in svg, pdf, ps and png formats to be used in text processing systems like DocBook, html, LaTeX. crystal-facet-uml exports the model to json and xmi format; json can also be imported. This tool runs on your local PC and is based on gtk (incl. glib, gdk, cairo, pango) and sqlite.

1.2 Features

crystal-facet-uml provides a graphical user interface to

- create, modify and delete diagrams,
- create, modify and delete UML/SysML elements,
- create, modify and delete relationships,
- 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 selects 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(TM)/SysML(TM) 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 interaction diagrams: sequence, communication, timing, interaction overview.

Diagrams can be exported as

- images: pdf, ps, svg, png,
- text: utf-8, DocBook, xhtml,
- machine-readable model: json, xmi(TM).

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

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

1.3 Usage Overview

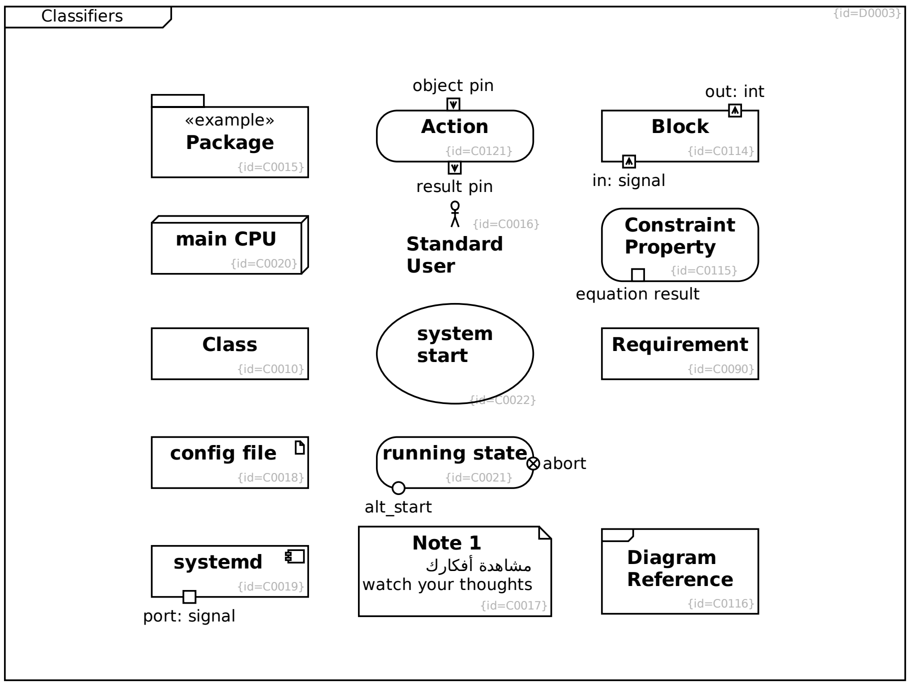
crystal-facet-uml can be started in graphical mode (see Section 3) or from command line (for help run **`crystal-facet-uml -h`**).

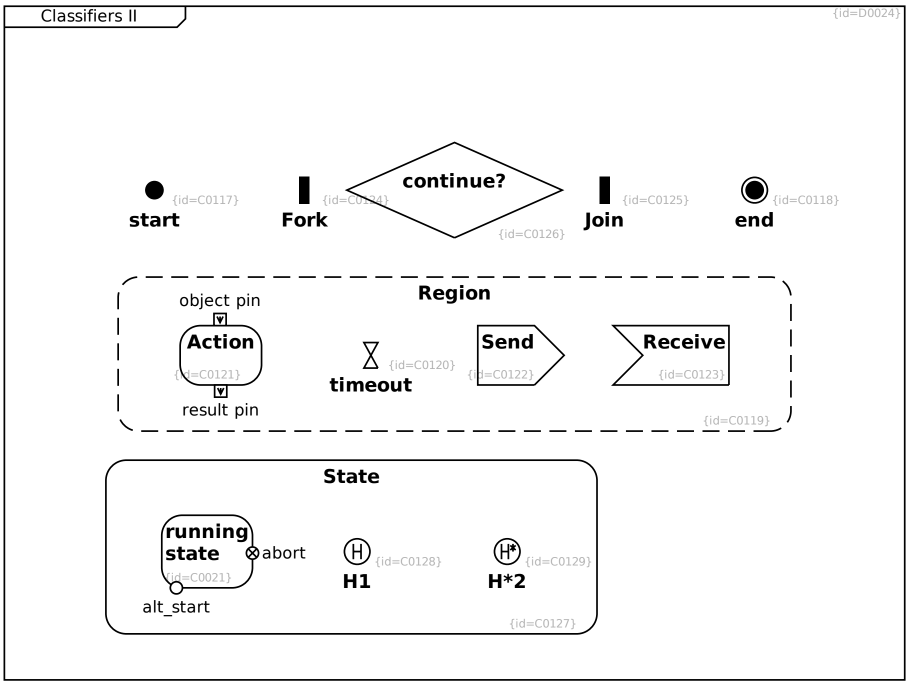
2 Example Diagrams

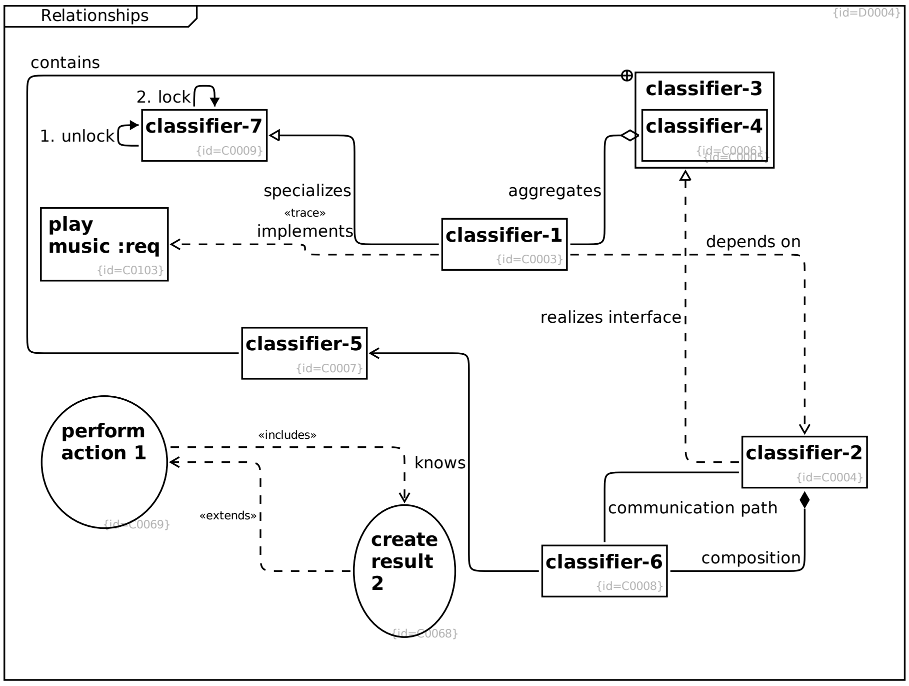
This sections presents the features of crystal-facet-uml.

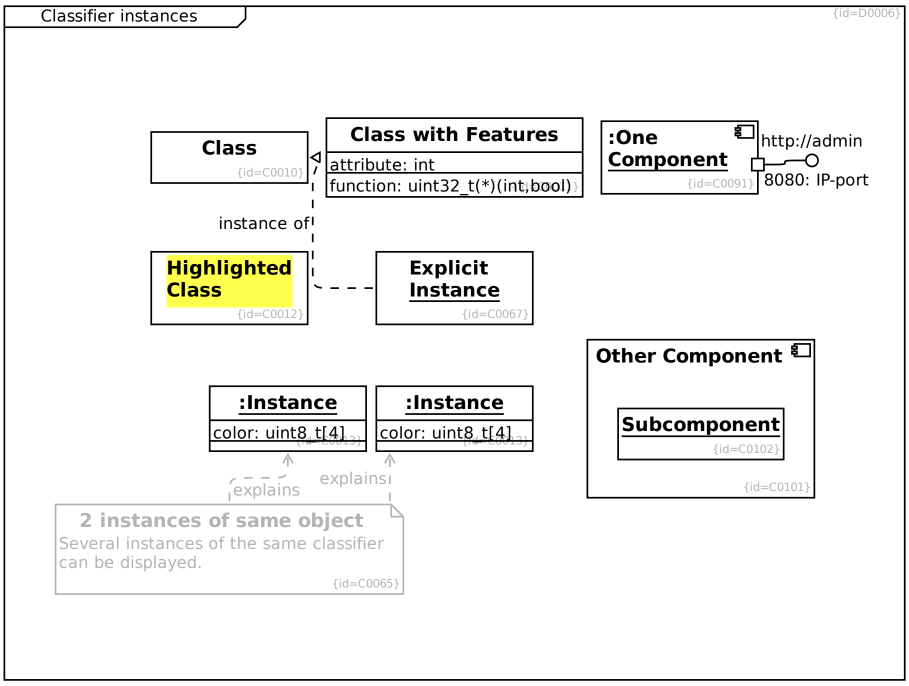
2.1 Feature List

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



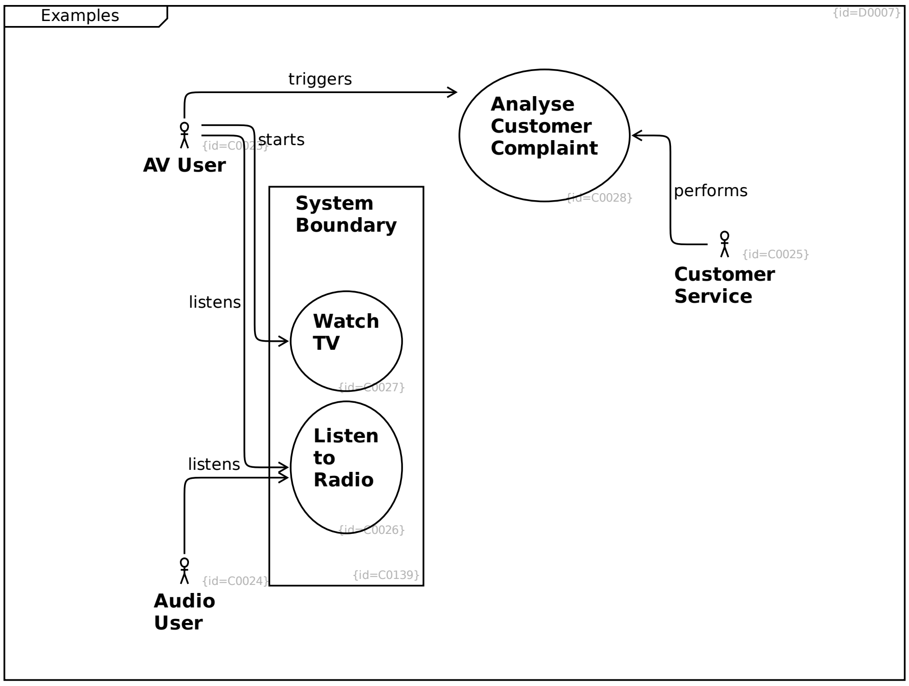

Classifiers II
{id=D0024}
start
{id=C0117}
Fork
{id=C0124}
continue?
{id=C0126}
Join
{id=C0125}
end
{id=C0118}
Region
object pin
Action
{id=C0121}
result pin
timeout
{id=C0120}
Send
{id=C0122}
Receive
{id=C0123}
{id=C0119}
State
running state
{id=C0021}
abort
alt_start
H1
{id=C0128}
H*2
{id=C0129}
{id=C0127}

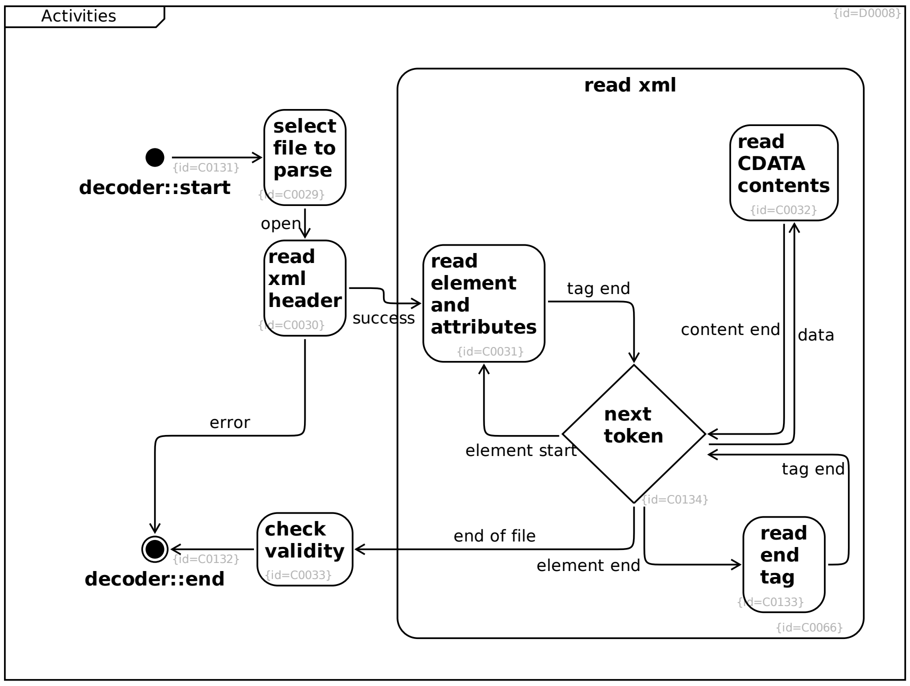

Relationships
{id=D0004}
contains
2. lock
1. unlock
classifier-7
{id=C0009}
classifier-3
classifier-4
{id=C0006}
specializes
aggregates
play music :req
{id=C0103}
«trace»
implements
classifier-1
{id=C0003}
depends on
realizes interface
classifier-5
{id=C0007}
perform action 1
{id=C0069}
«includes»
knows
«extends»
create result 2
{id=C0068}
classifier-2
{id=C0004}
communication path
classifier-6
{id=C0008}
composition

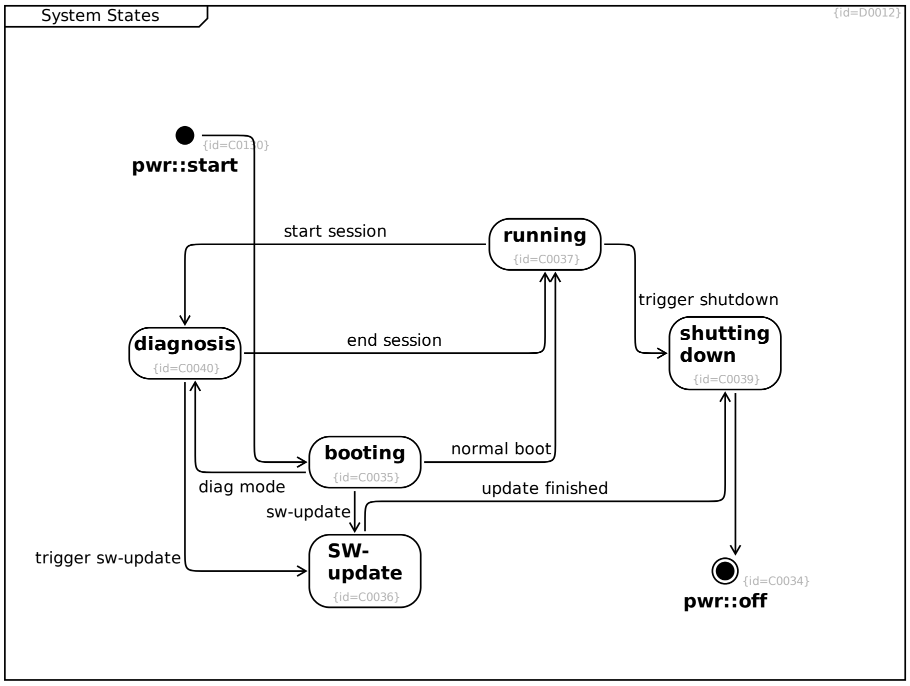


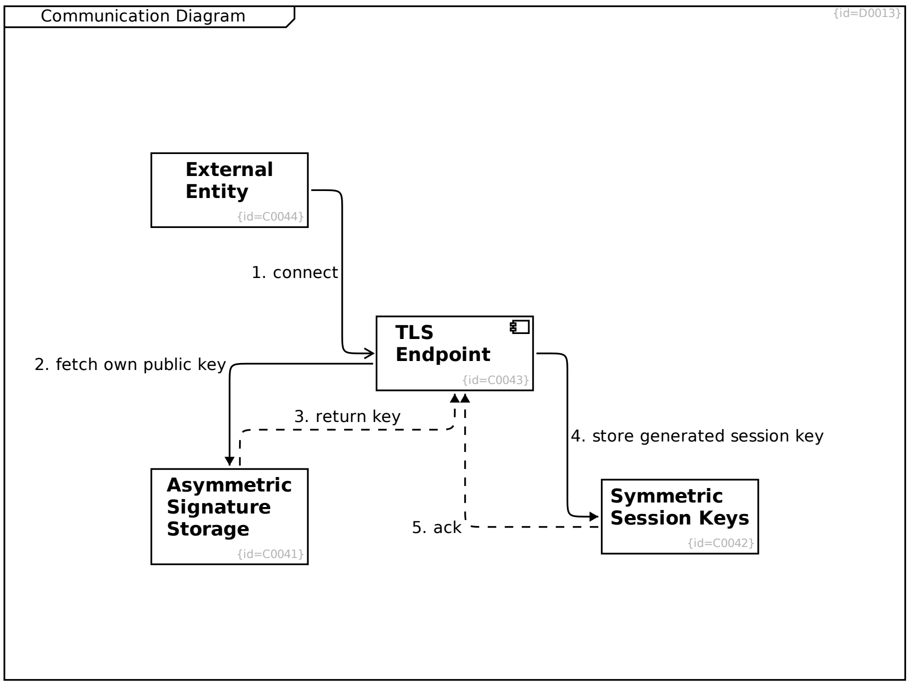
2.2 Example UML Behavioral Views

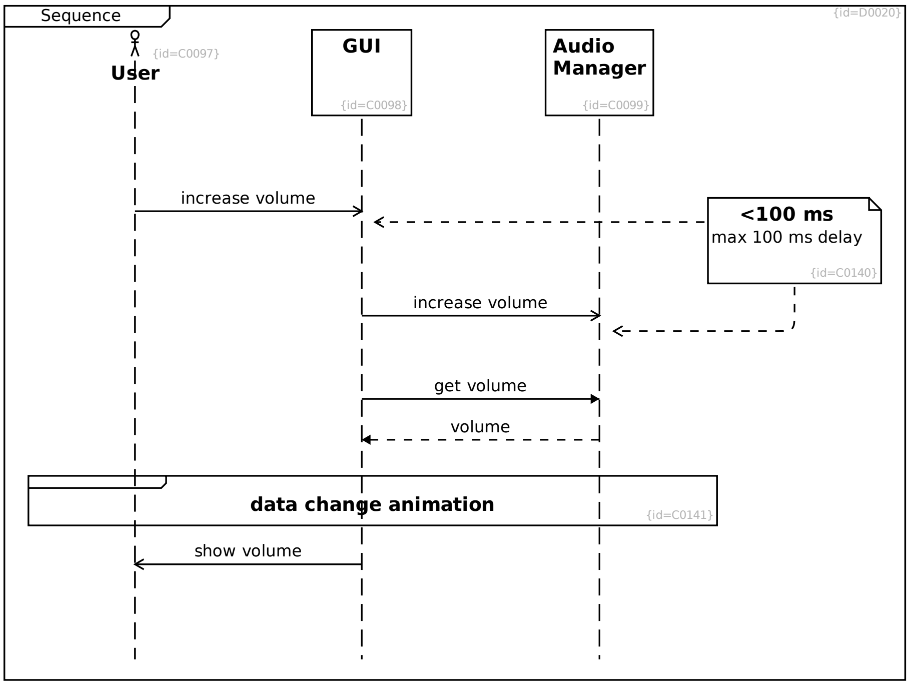
This section lists what kind of elements crystal-facet-uml can draw in diagrams.

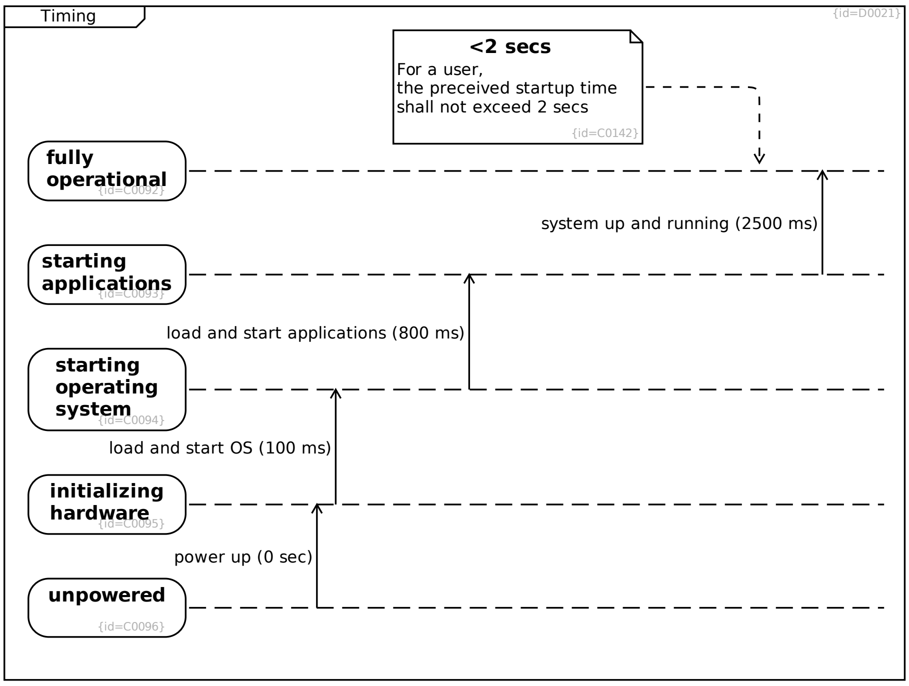


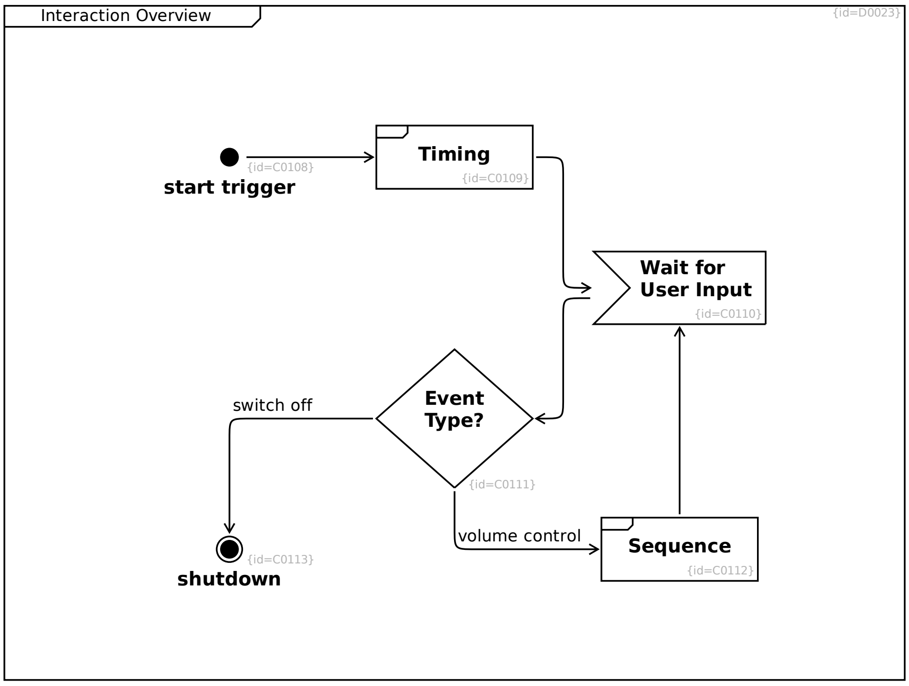

Activities
{id=D0008}
read xml
decoder::start
{id=C0131}
select file to parse
{id=C0029}
open
read xml header
{id=C0030}
success
read element and attributes
{id=C0031}
tag end
read CDATA contents
{id=C0032}
content end
data
next token
{id=C0134}
element start
tag end
error
check validity
{id=C0033}
end of file
element end
read end tag
{id=C0133}
decoder::end
{id=C0132}
{id=C0066}


System States
{id=D0012}
pwr::start
{id=C0130}
start session
running
{id=C0037}
trigger shutdown
diagnosis
{id=C0040}
end session
shutting down
{id=C0039}
booting
{id=C0035}
normal boot
diag mode
update finished
sw-update
trigger sw-update
SW-update
{id=C0036}
pwr::off
{id=C0034}


Communication Diagram
{id=D0013}
External Entity
{id=C0044}
1. connect
TLS Endpoint
{id=C0043}
2. fetch own public key
3. return key
4. store generated session key
Asymmetric Signature Storage
{id=C0041}
5. ack
Symmetric Session Keys
{id=C0042}

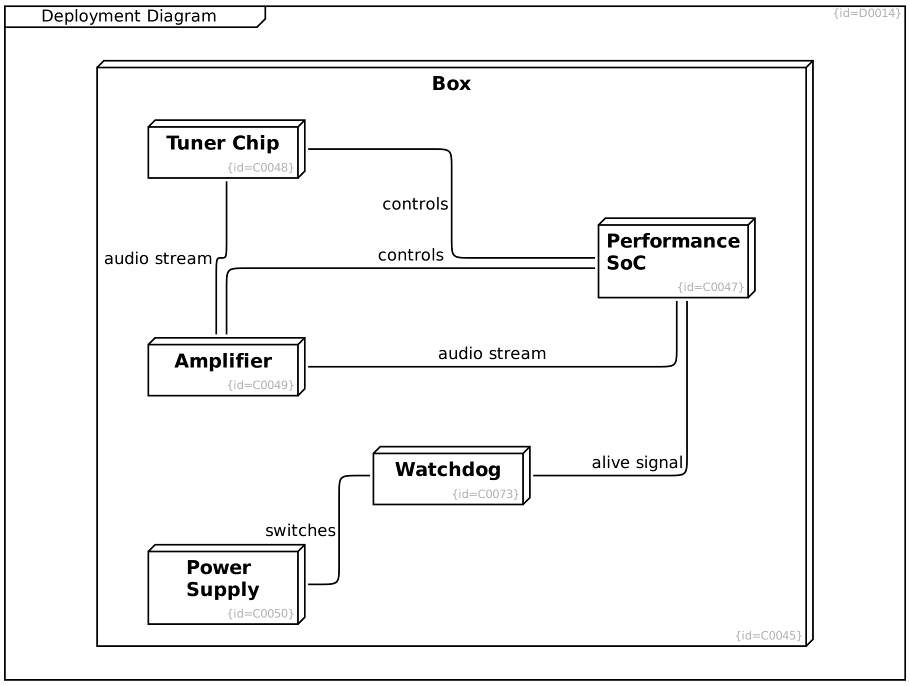

Sequence
{id=D0020}
User
{id=C0097}
GUI
{id=C0098}
Audio Manager
{id=C0099}
increase volume
<100 ms
max 100 ms delay
{id=C0140}
increase volume
get volume
volume
data change animation
{id=C0141}
show volume

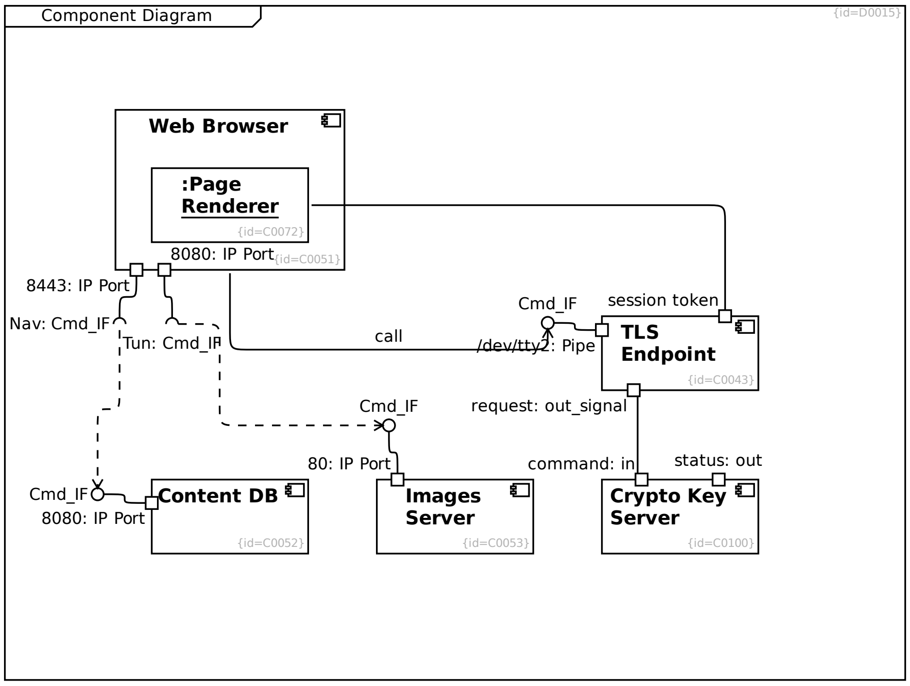


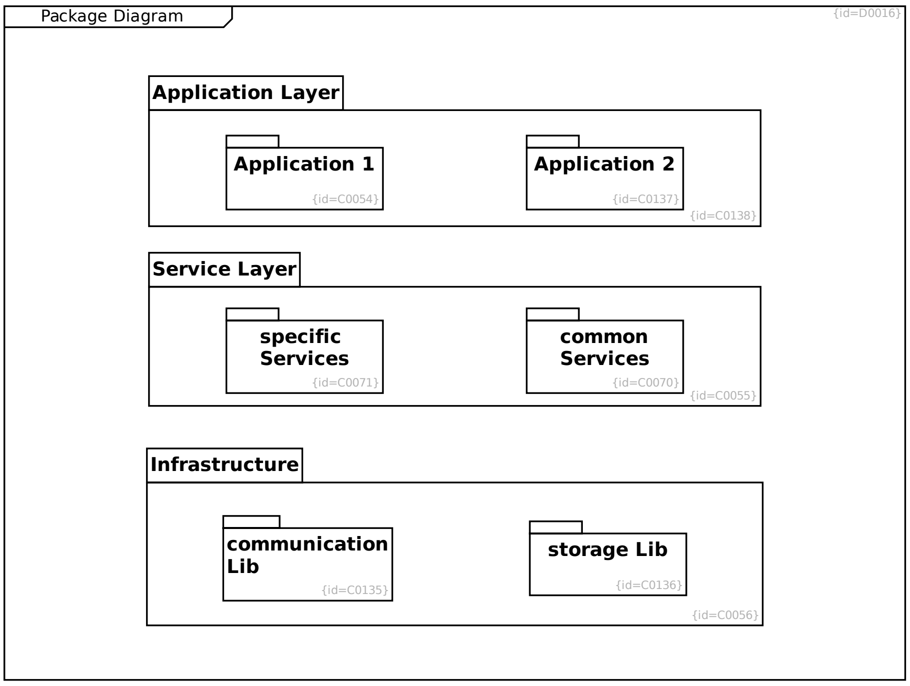


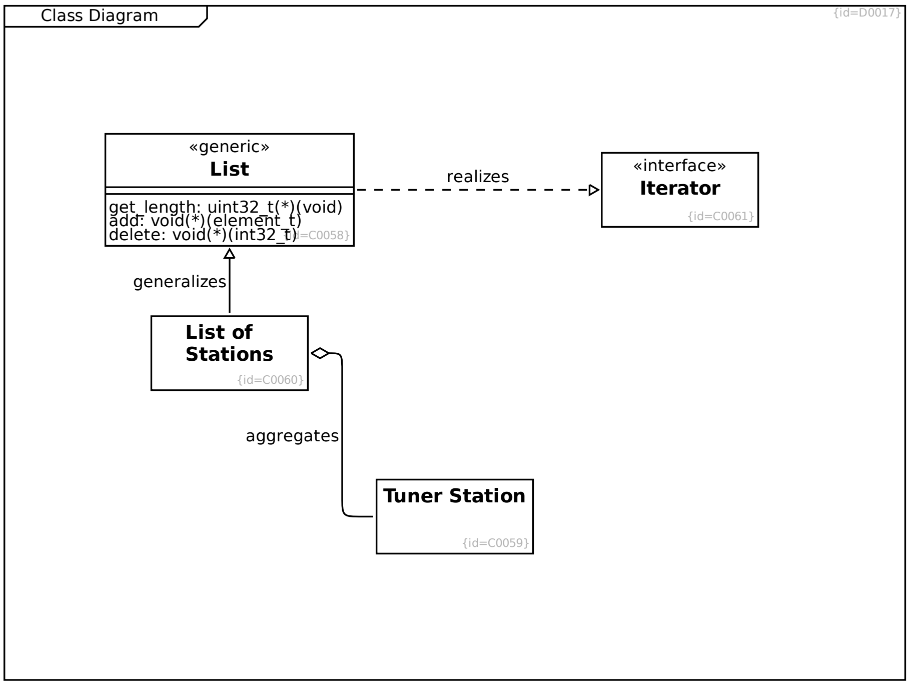
2.3 Example UML Static Views

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
controls
Performance SoC
{id=C0047}
Amplifier
{id=C0049}
audio stream
Watchdog
{id=C0073}
alive signal
switches
Power Supply
{id=C0050}
{id=C0045}

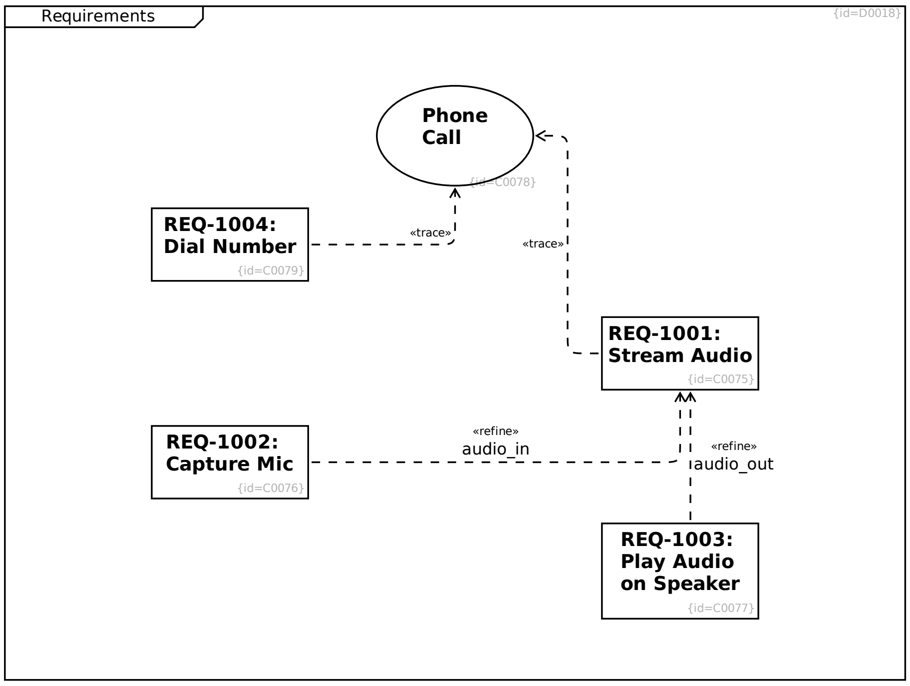

Component Diagram
{id=D0015}
Web Browser
:Page Renderer
{id=C0072}
8080: IP Port
{id=C0051}
8443: IP Port
Nav: Cmd_IF
Tun: Cmd_IF
call
Cmd_IF
session token
/dev/tty2: Pipe
TLS Endpoint
{id=C0043}
Cmd_IF
request: out_signal
80: IP Port
command: in
status: out
Cmd_IF
8080: IP Port
Content DB
{id=C0052}
Images Server
{id=C0053}
Crypto Key Server
{id=C0100}

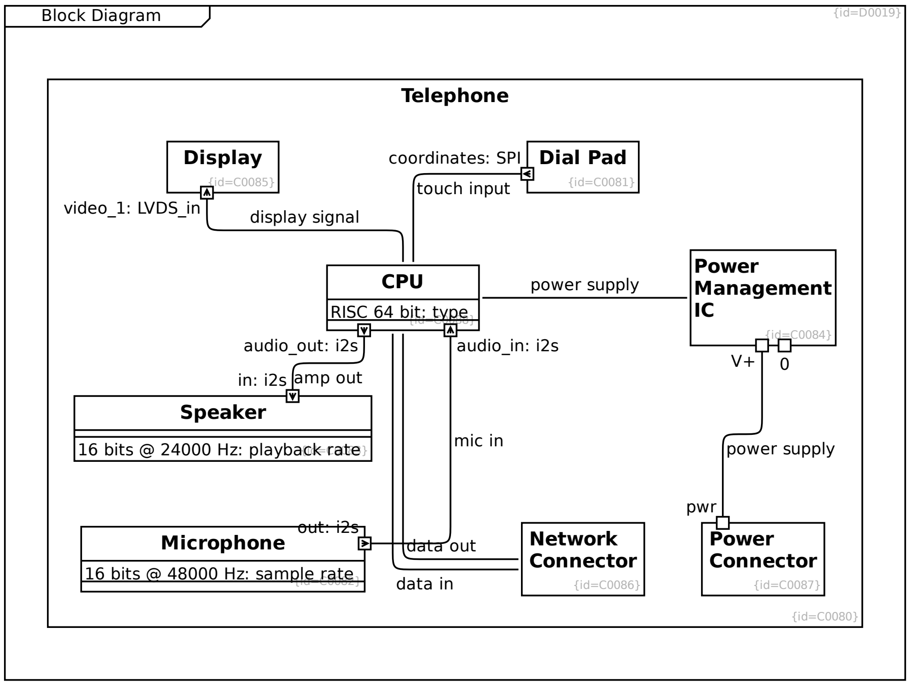


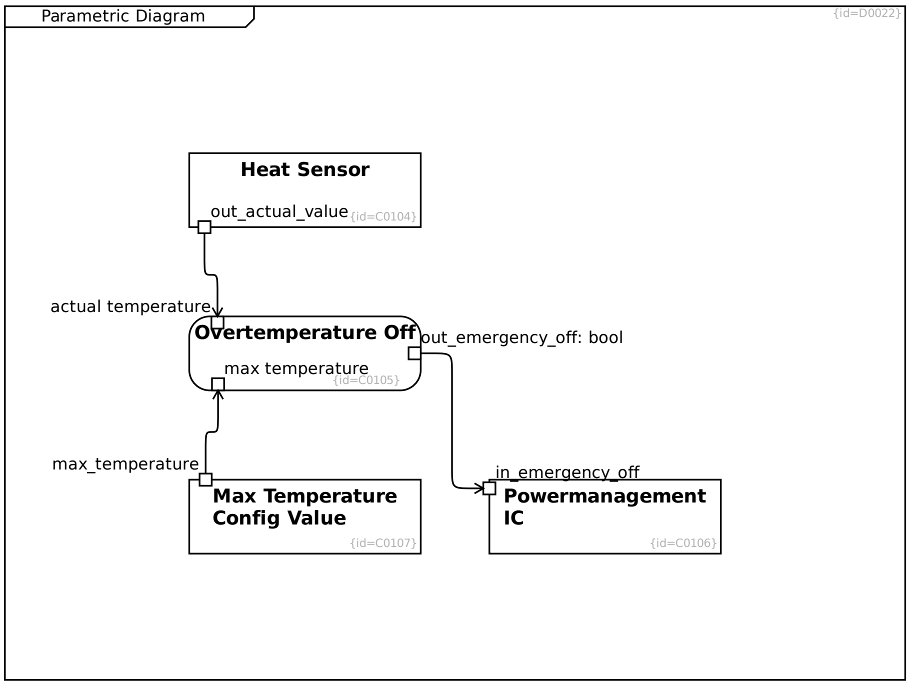


2.4 Example SysML Views

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


Requirements
{id=D0018}
Phone Call
{id=C0078}
REQ-1004: Dial Number
{id=C0079}
«trace»
«trace»
REQ-1001: Stream Audio
{id=C0075}
REQ-1002: Capture Mic
{id=C0076}
«refine»
audio_in
«refine»
audio_out
REQ-1003: Play Audio on Speaker
{id=C0077}


Block Diagram
{id=D0019}
Telephone
Display
{id=C0085}
video_1: LVDS_in
display signal
coordinates: SPI
Dial Pad
{id=C0081}
touch input
CPU
RISC 64 bit: type
power supply
Power Management IC
{id=C0084}
V+
0
audio_out: i2s
audio_in: i2s
in: i2s
amp out
Speaker
16 bits @ 24000 Hz: playback rate
mic in
power supply
pwr
Microphone
out: i2s
16 bits @ 48000 Hz: sample rate
data out
data in
Network Connector
{id=C0086}
Power Connector
{id=C0087}
{id=C0080}



2.5 More Examples

There are further examples available as html:

- mouse_droid.xhtml / mouse_droid.pdf
- self_architecture.xhtml / self_architecture.pdf
- quality.xhtml / quality.pdf

And in crystal-facet-uml binary format:

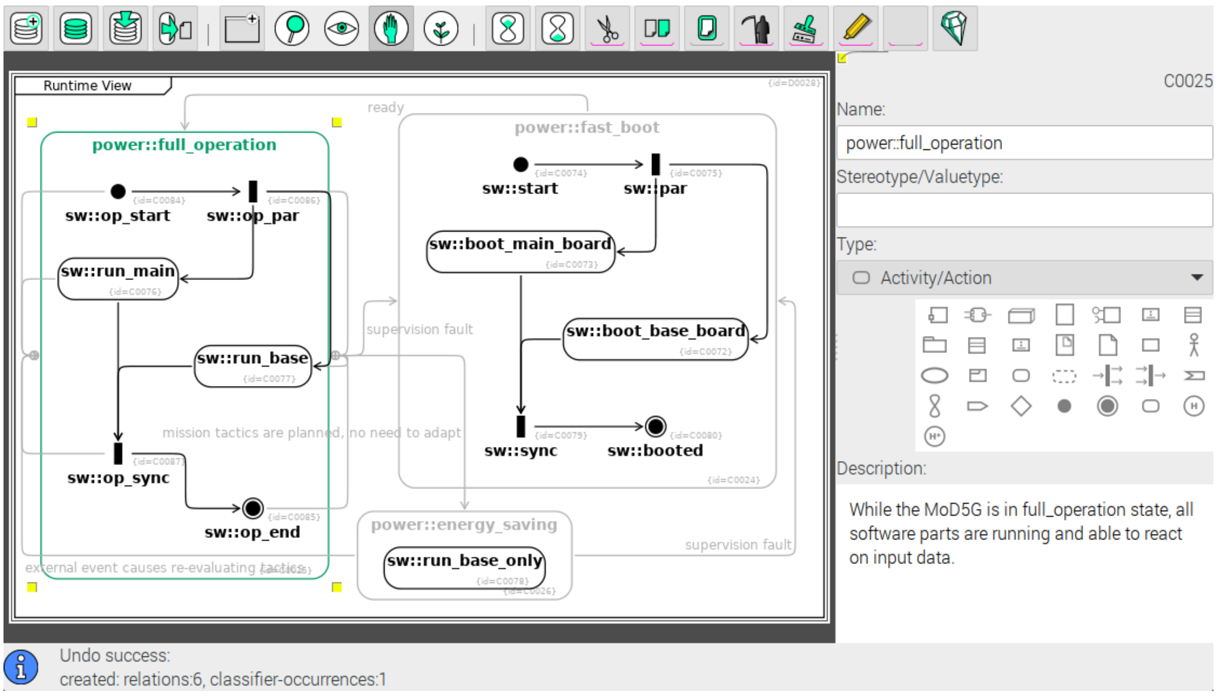
- `https://github.com/awarnke/crystal-facet-uml/tree/master/example_diagrams`
- `https://github.com/awarnke/crystal-facet-uml/tree/master/architecture/doc`

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.



C0025

Name:

power::full_operation

Stereotype/Valuetype:

Type:

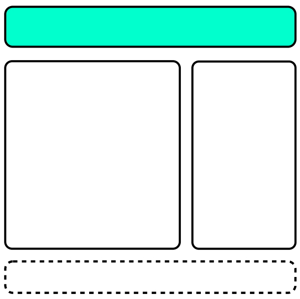
Activity/Action

Description:

While the MoD5G is in full_operation state, all software parts are running and able to react on input data.

Undo success:
created: relations:6, classifier-occurrences:1

3.2 Tool Bar



3.2.1 New DB

- Creates a new database file.
- If the file name ends with extension **.cfu1**, the binary sqlite3-based database format is stored. This is compatible to older versions of crystal-facet-uml (<1.35.0). Otherwise a json-based format is used to store your data in a git-friendly format.

3.2.2 Open DB

- Opens an existing database file.
- To open json-based formats (e.g. **.cfuJ**), 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 may open this file to continue from the latest state or the **.cfuJ** to continue at the last save action.
- If you get a warning at opening a json file, close the program and try to import the json file from command line: **crystal-facet-uml -i <NEW_DB_FILE>.cfu1 add <EXISTING_JSON_FILE>.cfuJ** On one hand you get better error messages, on the other hand there is also a repair option that may be of help.

3.2.3 Save

- Stores the latest changes to the database immediately.

3.2.4 Export

- Exports all diagrams to the selected folder. To select the export folder, navigate to the parent folder and select the target folder. Supported formats are docbook, json, pdf, png, ps, svg, txt, xhtml, xmi.

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 contains a diagram, this diagram is pasted below the current diagram. All other elements are pasted into the new diagram.
- If the clipboard does not contain diagrams, classifiers and relationships from the clipboard are copied into the current 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 Highlight

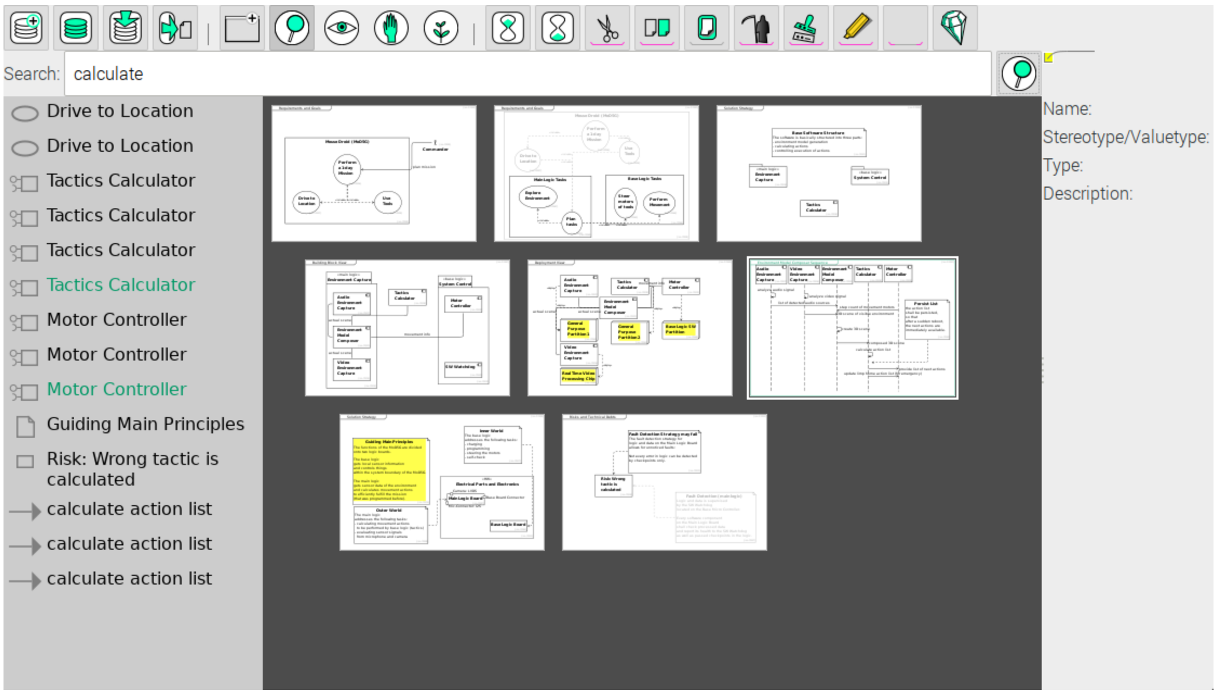
- Toggles the selected (pink-cornered) classifiers between yellow-marked, greyed-out and normal. (Does not work for relationships and features)

3.2.18 Reset Selection

- Resets the (pink-cornered) selection

3.2.19 About

- Shows version, license and copyrights

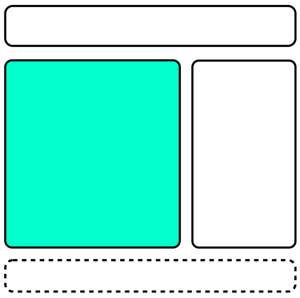


Search: calculate

Drive to Location
Drive to Location
Tactics Calculator
Tactics Calculator
Tactics Calculator
Tactics Calculator
Motor Controller
Motor Controller
Motor Controller
Guiding Main Principles
Risk: Wrong tactic is calculated
calculate action list
calculate action list
calculate action list

Name:
Stereotype/Valuetype:
Type:
Description:

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 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.

3.3.2 Navigate

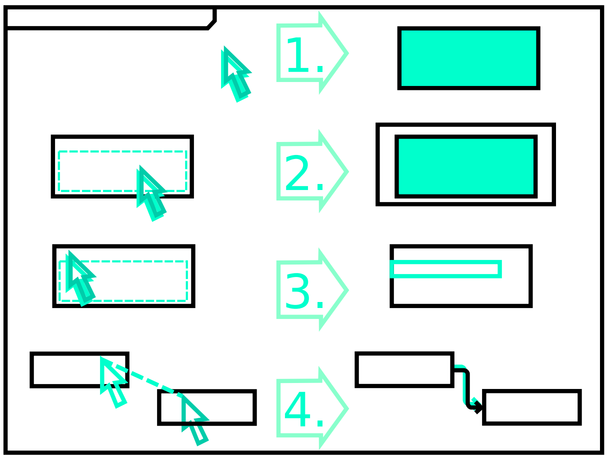
- To navigate to parent, sibling or children diagrams, click on the diagram.
- To create a new diagram, click on the icon, or the smaller icon for a new child-diagram.
- To restructure the diagram tree, drag a diagram name to the new location.

3.3.3 Edit

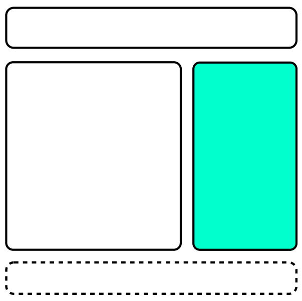
- Click on the diagram or a classifier or a feature or a relationship to edit the name, type and description of that object.
  The yellow corners indicate which object is currently focused.
- Click on an element to select or unselect an object (pink corners).
  The toolbar buttons apply to this pink-cornered 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.

3.3.4 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.



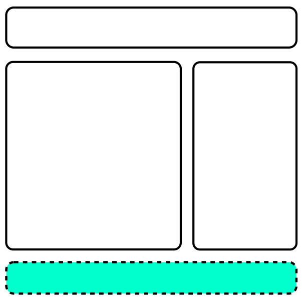
3.4 Element Configuration Area



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

- name of the focused object
- stereotype/valuetype of the focused object.
  Stereotype names may consist of characters that are valid XML tokens (Nmtoken).
  This field is deactivated for diagrams and relationships.
  For classifiers, multiple stereotypes shall be separated by comma.
- type of the focused object
- description of the focused object.
  For xhtml 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



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

3.6 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.

3.6.1 Storing data in a version control system

To store a database in a version control system (vcs) like svn or git, you have several options:

- Store the binary database format ***.cfu1** to your vcs, lock the file while modifying it.

  Advantage is a) the simple usage and b) the option to view read-only files. Disadvantage is the lack of a) diff and merge of files, b) identify who changed which line when and with which commit message, c) concurrent modifications.

- Store the json file format ***.cfuJ** to your vcs. Do not synchronize the file with your vcs while you modify it.

  Note that this feature requires at least version 1.35.0, better version 1.39.0 to work smoothly.

  On command line, there is an option to run consistency checks **-t** or repair **-r** the database:

  crystal-facet-uml -t my_database_file.cfu1 || echo "ERROR $?"

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, if they filter/hide their features and a comment stating how this is implemented in crystal-facet-uml.

	Spec	**Diagram Context and Filter**	**Comment**
Block	SysML	* / -	Limitations: Compartment Order is "properties, operations" instead of "constraints, operations, receptions, parts, (bound) references, values, properties, stereotype-tagged-values, behavior, namespace, structure" Limitations: No labeled compartments 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 / unconditional features	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	* / unconditional features	
Requirement	SysML	* / -	
Actor	UML, SysML	Use Case, Sequence / unconditional features	

	Spec	**Diagram Context and Filter**	**Comment**
Use Case	UML, SysML	Use Case / -	Limitations: No SysML extension points
Interaction Diagram Reference (Interaction Use)	UML	Interaction Overview / unconditional features	Hint: To easily find the referenced diagram, name the reference identical to the diagram. XMI-Export: For xmi export, this object may only occur in scenario/interaction diagrams.
Activity/Action	UML 2.5 (ch15.2)	Activity / -	
Interruptable Region	UML	Activity / unconditional features	XMI-Export: For xmi export, all regions belonging to the same set of activities need an outer, enclosing activity.
Fork	UML, SysML	Activity / unconditional features	XMI-Export: For xmi export, all activity-nodes belonging to the same set of activities need an outer, enclosing activity.
Join	UML, SysML	Activity / unconditional features	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 / unconditional features	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 / unconditional features	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 / unconditional features	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 / unconditional features	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.

	Spec	**Diagram Context and Filter**	**Comment**
● Initial Node	UML 2.5 (ch14.2.4), SysML	Activity, State / unconditional features	Limitations: There is no distinction in ActivityInitial and FlowInitial. Limitations: There is no separate entryPoint 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.
◉ Final Node	UML 2.5 (ch14.2.4), SysML	Activity, State / unconditional features	Limitations: There is no distinction in ActivityFinal and FlowFinal. Limitations: There is no separate exitPoint and 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. Limitations: entryPoint and exitPoint states cannot be drawn on parent state border line. 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 / unconditional features	XMI-Export: For xmi export, all states belonging to the same statemachine need an outer, enclosing state.
H* Deep History	UML 2.5 (ch14.2.4), SysML	State / unconditional features	XMI-Export: For xmi export, all states belonging to the same statemachine need an outer, enclosing state.
× 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.

	Spec	Diagram Context and Filter	Comment
× ActivityParameterNode	SysML	- / -	not supported.
× MergeNode/Junction	UML 2.5 (ch15.3), SysML	Activity, State / unconditional features	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 / unconditional features	not supported. Note: Use a parent activity instead.

LEGEND

Filter Defines which elements related to a classifier are not visible

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	Scope	Comment
Property	UML, SysML	unconditional	Limitations: no SysML Flow-Properties refinement.
Operation	UML, SysML	unconditional	
Provided Interface	UML, SysML	unconditional	
Required Interface	UML, SysML	unconditional	

	Spec	Scope	Comment
Port	UML, SysML	unconditional	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	unconditional	Exists since version 1.27.0; cannot be used in version 1.26.1 and older.
Output Port/Pin	UML, SysML	unconditional	Exists since version 1.27.0; cannot be used in version 1.26.1 and older.
State Entry	UML, SysML	unconditional	Exists since version 1.27.0; cannot be used in version 1.26.1 and older.
State Exit	UML, SysML	unconditional	Exists since version 1.27.0; cannot be used in version 1.26.1 and older.
Lifeline	UML 2.5 (ch17.2), SysML	interaction scenario, 1 per diagram	Limitations: One lifeline is visible only in one diagram. Limitations: Lifelines start and end only at diagram border. Limitations: ExecutionSpecification (ActivityBar) are not supported.

LEGEND

Scope scope is unconditional if a feature belongs to a classifier unconditionally, scenario if only applicable in 1 interaction diagram

4.3 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	Diagram Context	Comment
Dependency	UML, SysML	any	
Containment	UML, SysML	Deployment, Package, Internal Block, Composite Structure, Activity, State	

	Spec	Diagram Context	Comment
«deploy» ⇢ Deploy	UML	Deployment	
«manifest» ⇢ 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, Use Case(?)	Limitations: no Generalization-Sets supported
- - -▷ Realization	UML	Class	
«trace» ⇢ Trace	SysML	Requirement	
«refine» ⇢ Refine	SysML	Requirement	
«extend» ⇢ Extend	UML, SysML	Use Case	Limitations: no SysML-condition-notes can be attched to this relationship
«include» ⇢ 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.

	Spec	Diagram Context	Comment
→ Object Flow	UML, SysML	Activity	
→ Async. Call	UML, SysML (?)	for sequence, timing, communication and interaction overview diagrams	
➞ Sync. Call	UML, SysML (?)	for sequence, timing, communication and interaction overview diagrams	
◀- - - Return Call	UML, SysML (?)	for sequence, timing, communication and interaction overview diagrams	
× 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	not yet supported, see todo.txt.

4.4 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, if they filter/hide their features and/or relationships and a comment stating how this is implemented in crystal-facet-uml.

	Spec	Filter	Comment
List Diagram	-	any feature, any relationship	This is an overview diagram showing only classifiers without features and without relationships
Box Diagram	-	any feature, any relationship	This is an overview diagram showing only classifiers without features and without relationships
Block Definition Diagram	SysML	lifelines	
Internal Block Diagram	SysML	lifelines	

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

	Spec	Filter	Comment
Timing Diagram	UML	unconditional relationships (non-scenario), unconditional features	

LEGEND

Filter Defines which elements are not visible in the diagram

Scenario Interaction diagrams show only relationships associated with a lifeline of a visible classifier.

4.5 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,

5 Modeling Guidelines

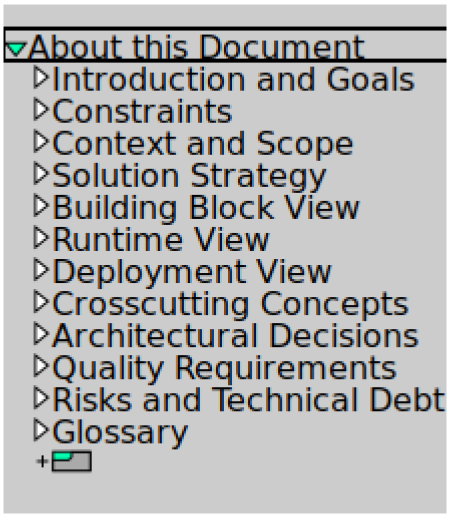
This section lists hints on efficiently using the tool crystal-facet-uml and provides some general remarks on creating a software architecture and design document.

5.1 crystal-facet-uml Hints

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

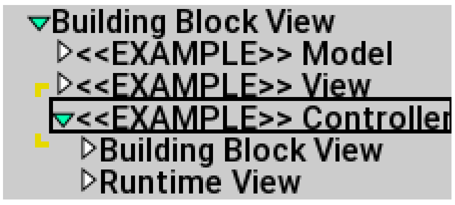
5.1.1 Tree Structure

Diagrams are organized as a tree. 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/ :



```
▽About this Document
  ▷Introduction and Goals
  ▷Constraints
  ▷Context and Scope
  ▷Solution Strategy
  ▷Building Block View
  ▷Runtime View
  ▷Deployment View
  ▷Crosscutting Concepts
  ▷Architectural Decisions
  ▷Quality Requirements
  ▷Risks and Technical Debt
  ▷Glossary
  +
```

In case you show several layers of abstraction, each building block may contain its sub-blocks, sub-blocks may again show sub-sub-blocks. In this case, structure the specification of the sub-blocks in the same way: apply the proposed folder structure recursively, omitting possibly empty or superfluous folders.



```
▽Building Block View
  ▷<<EXAMPLE>> Model
  ▷<<EXAMPLE>> View
  ▽<<EXAMPLE>> Controller
    ▷Building Block View
    ▷Runtime View
```

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.

When distributing different aspects to different diagrams, a remaining challenge may be that there is no filter on relationships. If e.g. 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).

5.1.3 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.

To achieve a more compact layout of an element, one may insert space characters into names. (In case names get long, the space allows for a linebreak).

5.1.4 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.

5.1.5 Layout

To change the positions of classifiers anf features, drag these to other locations. 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 Download and Install

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

6.1 Documentation

User documentation is available here:

- `https://andreaswarnke.de/crystal-facet-uml/crystal-facet-uml_documentation.pdf`
- `https://github.com/awarnke/crystal-facet-uml/blob/master/user_doc/crystal-facet-uml_documentation.pdf`

6.2 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

6.3 SuSE Linux

To install, type on the command line:

```
sudo zypper addrepo https://download.opensuse.org/repositories/devel:/tools/15.4  ↩
    devel_tools_15.4
# or sudo zypper addrepo https://download.opensuse.org/repositories/devel:/tools/ ↩
    openSUSE_Leap_15.3 devel_tools_15.3
sudo zypper refresh
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

6.4 Windows/Wine

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**,
- or using the wine emulation, download **libgcc_s_seh-1.dll** and unpack this to bin,
- then call **export XDG_DATA_HOME=".\\share" ; wine64 bin/crystal-facet-uml.exe**.

6.5 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.markdown**.

6.6 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

6.7 License