



Building Reliable AV applications with Compute Graph Framework (CGF)

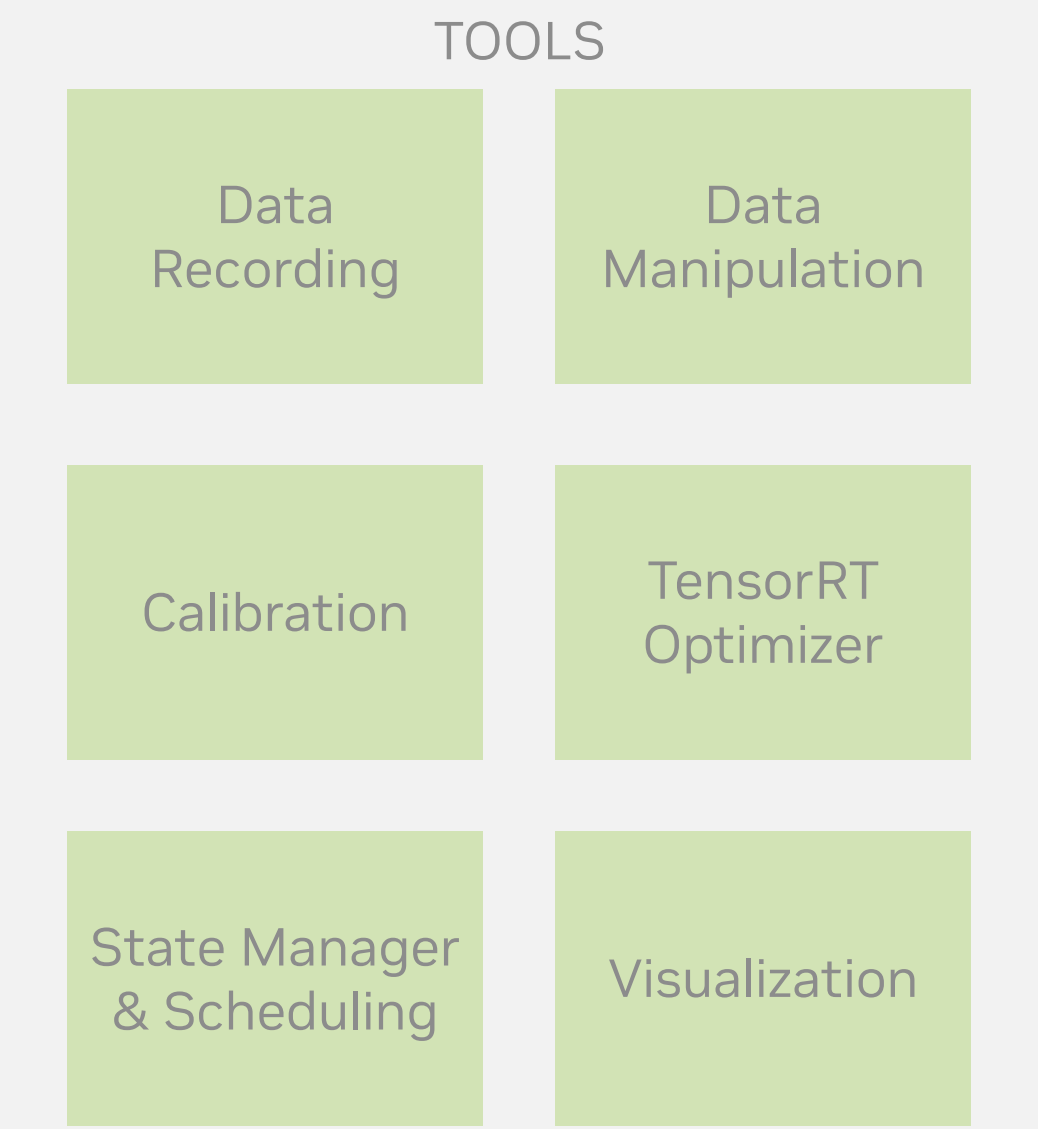
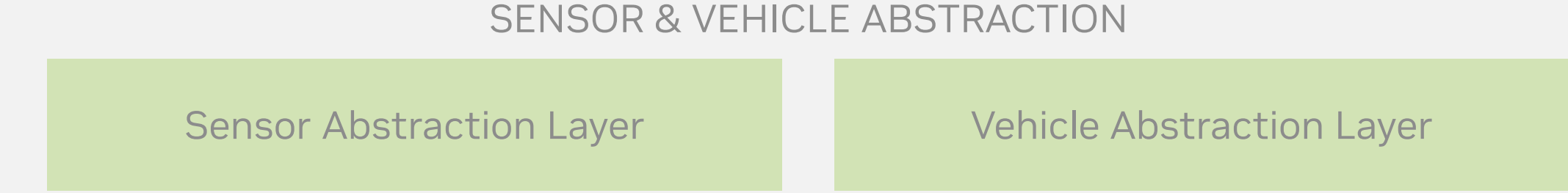
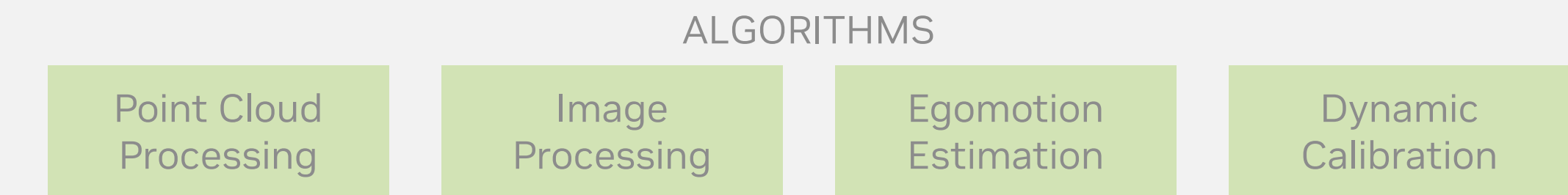
August 2023

DRIVE SDK Supercharges AV Development

One architecture
DRIVE OS & DriveWorks
DRIVE AGX Orin

DriveWorks
Middleware

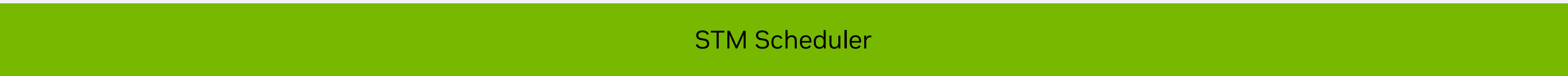
Application Building Blocks



Application Framework



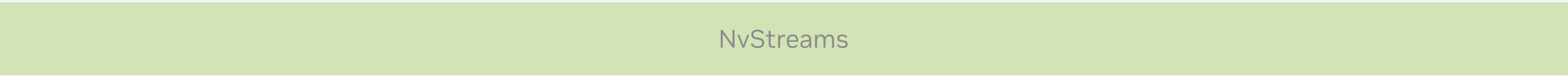
Deterministic Scheduler



Application APIs



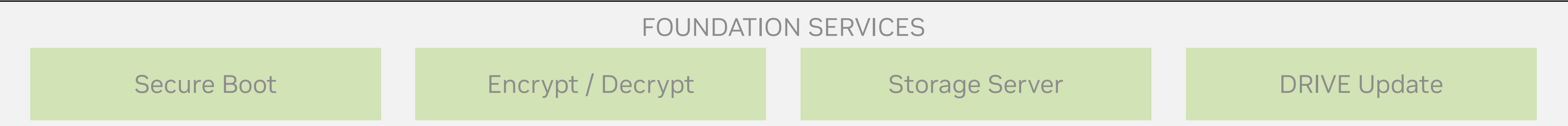
Hardware Synchronization



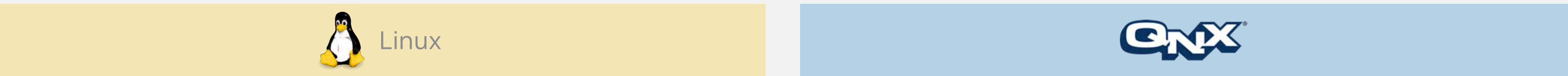
Safety Framework
(QNX Only)



DRIVE OS
Linux / QNX



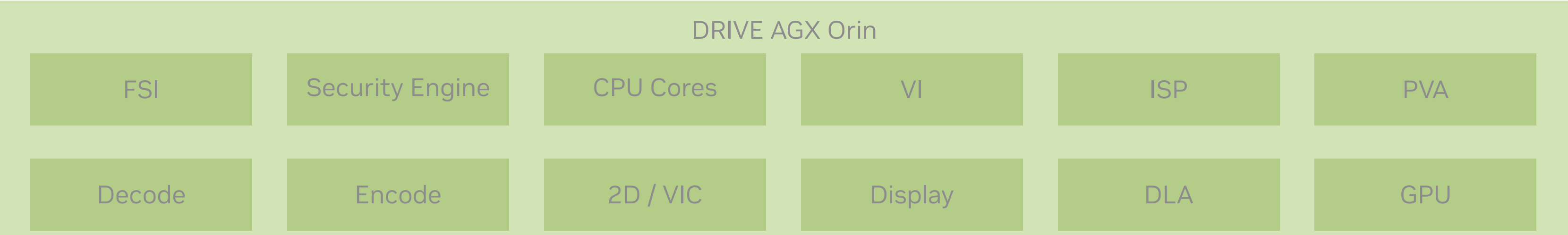
Operating System



Resource Isolation &
Freedom of Interference



Hardware

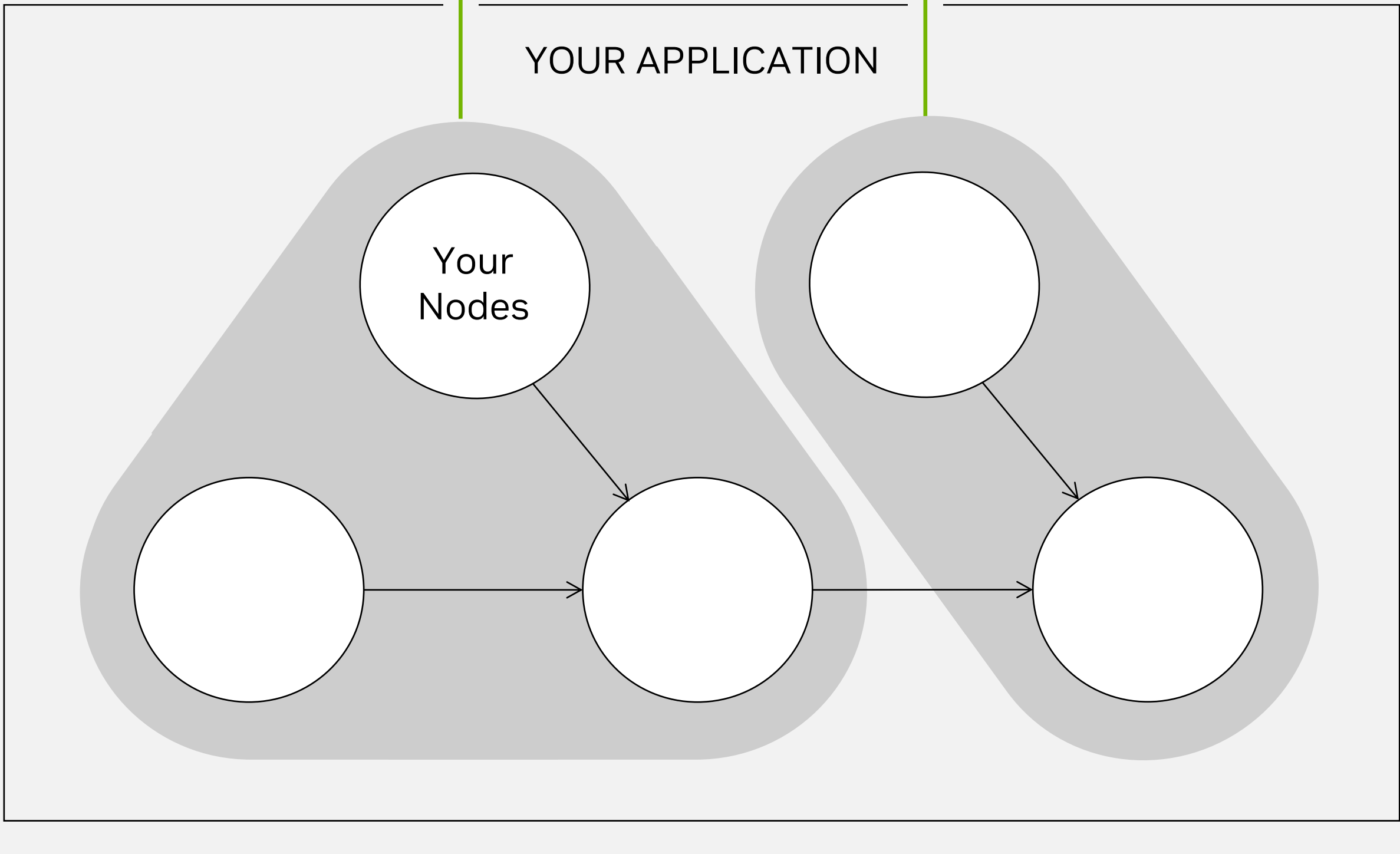


DriveWorks — Comprehensive Middleware Solution

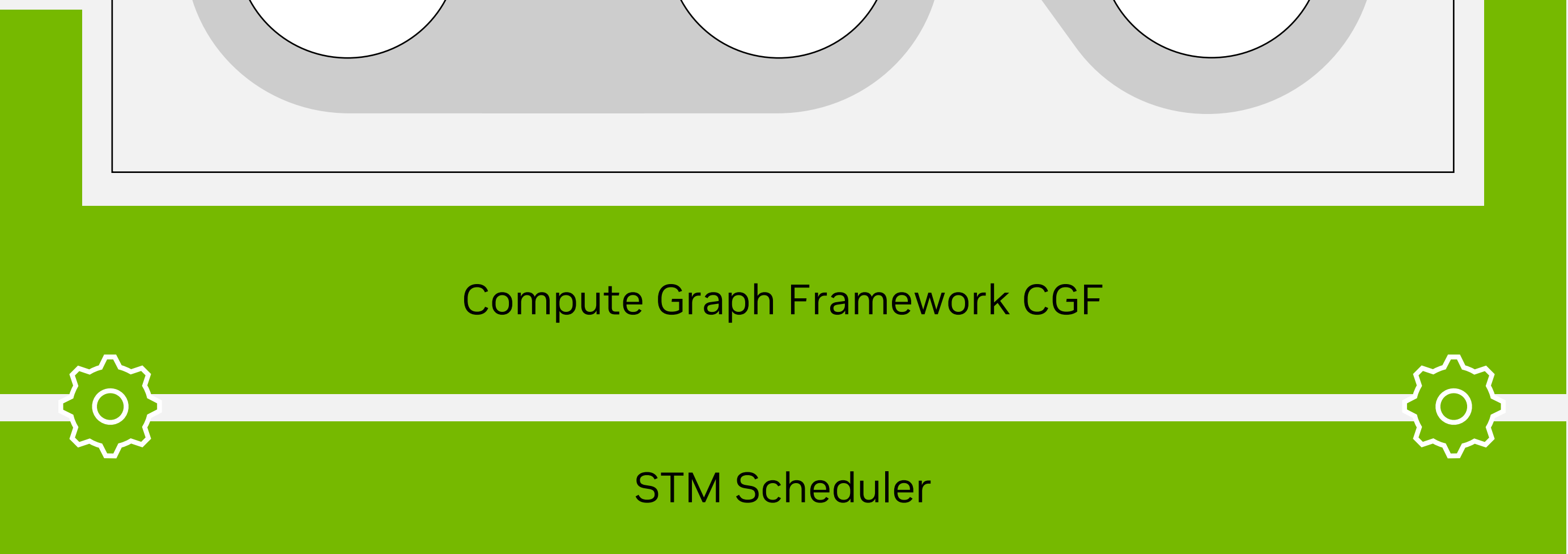
Rich Library of Algorithms and Tools
to accelerate your applications

DRIVEWORKS ACCELERATION LIBRARIES & TOOLS			
Calibration	Visualization	Data Manipulation	TensorRT Optimizer
Vehicle Abstraction Layer	Egomotion Estimation	Data Recording	High-Speed Transport Framework
Sensor Abstraction Layer	Image Processing	Point Cloud Processing	Logging & Diagnostics

Compute Graph Framework
to simplify complex computational pipelines



System Task Manager
to deterministically schedule application tasks

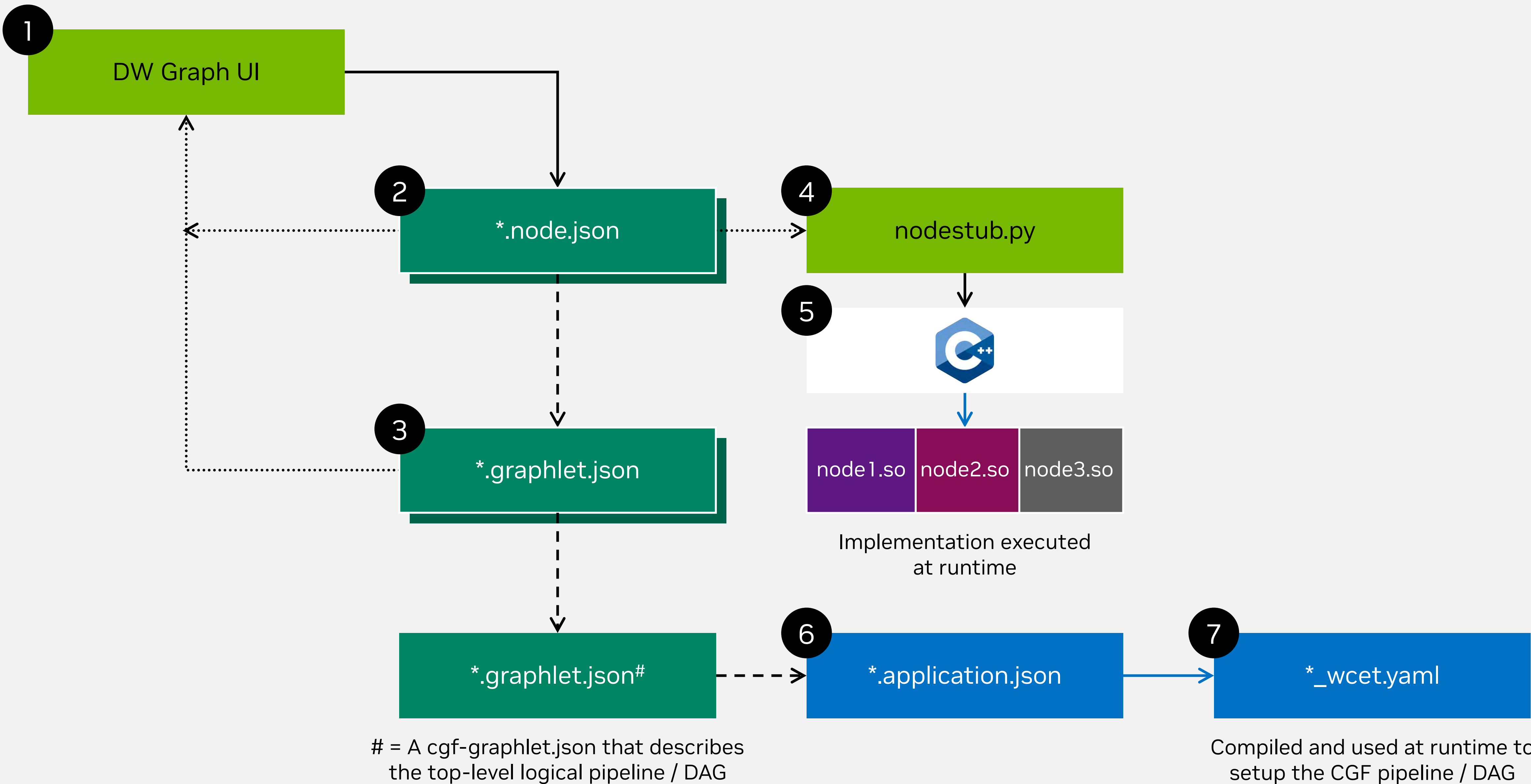




Agenda

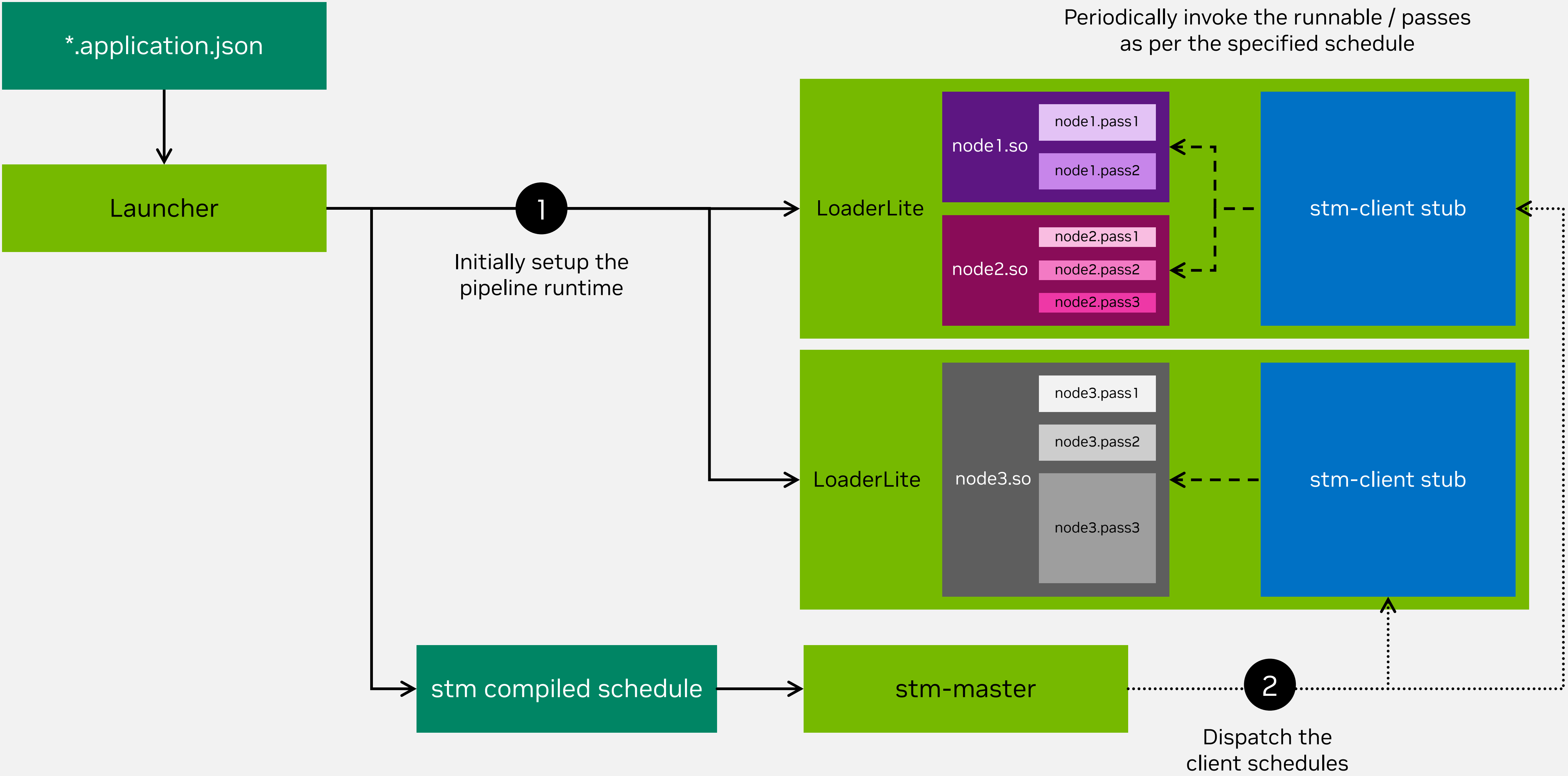
- CGF Workflow at Design and Development Time
- CGF Execution Flow at Runtime
- What is a node
- Typical pipeline design and development workflow
- DW Graph UI — Introduction to Nodes and Graphlets
- Nodestub — Autogenerating Code for Custom Nodes
- Adding Custom Logic Within Autogenerated Code
- Introduction to an Application (CGF Demo)
- stmvizschedule — Static Visualization
- Profiling — NSight Systems + NVTX
- Overall Safety Concept

CGF Workflow at Design / Development Time



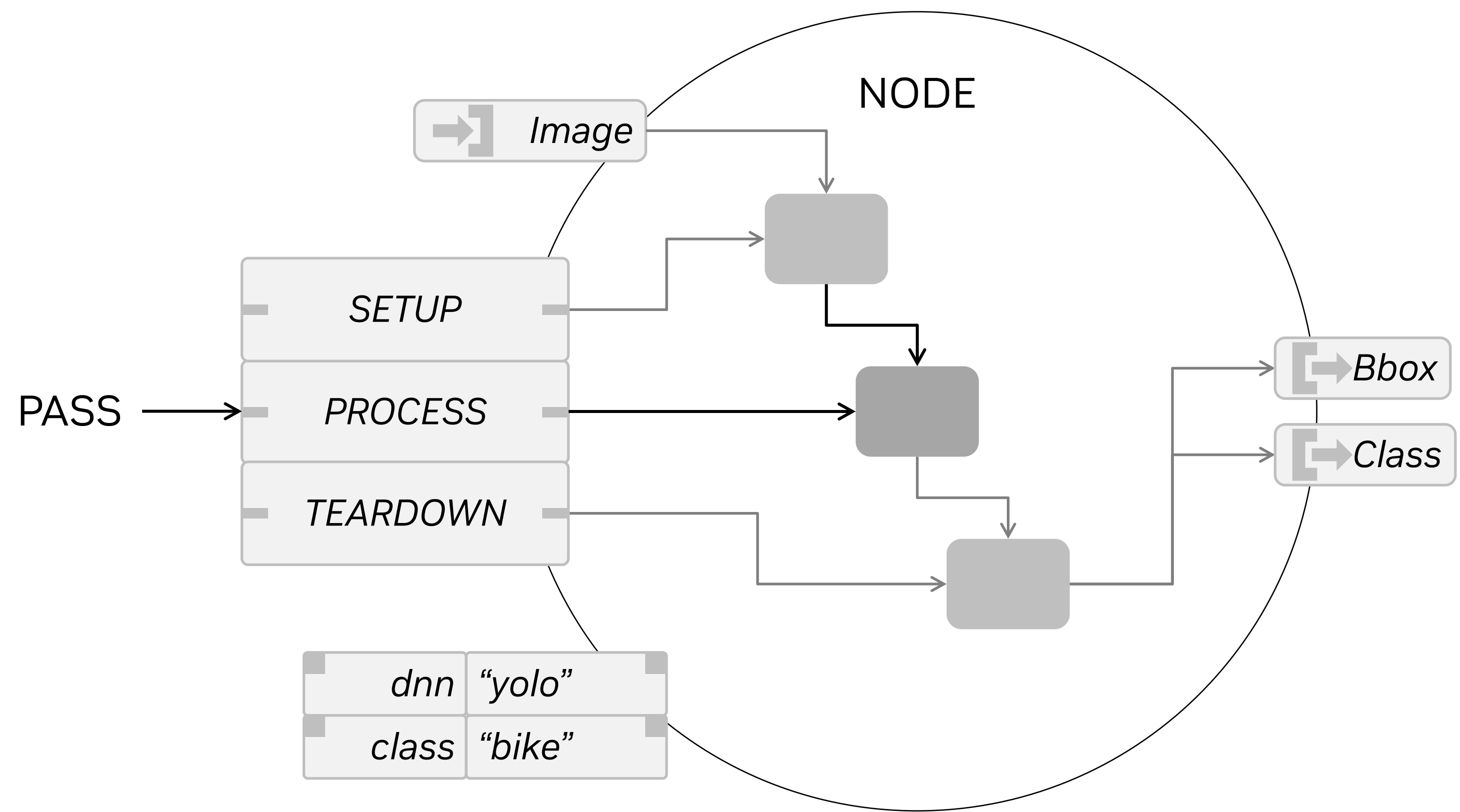
CGF Execution Flow at Runtime

All application logic required to be executed deterministically as a pipeline



What is a CGF NODE

Key components and implementation



○ **Node**
Inheriting from base Node
Constructor receives parameters
Create function registers node

➡ **Ports**
Inputs/Output ports specifying data type and unique port name

➡➡ **Passes**
Passes to be executed in order, on specified compute engine

a	x
b	y

Parameters
Parameters for the constructor

➡ **Ports**
Initialize ports, prepare output data containers

➡➡ **Passes**
Register methods used for passes defined in public interface
Implementation of the passes

Public Node Interface **MyNode.hpp**

```
class MyNode : public dw::framework::ExceptionSafeProcessNode
{
    MyNode(const MyNodeParams& params, ...);

    dw::framework::create<Node>(ParameterProvider& provider);

    static constexpr auto describeInputPorts() {...};
    static constexpr auto describeOutputPorts() {...};

    static constexpr auto describePasses() {
        return dw::framework::describePassCollection(
            dw::framework::describePass("SETUP", DW_PROCESSOR_TYPE_CPU), ...);
    }

    static constexpr auto describeParameters() {...};
}
```

Implementation **MyNode.cpp**

```
void MyNodeImpl::initPorts() {
    NODE_INIT_INPUT_PORT("IMAGE"_sv);
    ...
}

void MyNodeImpl::initPasses() {
    NODE_REGISTER_PASS("PROCESS"_sv, [this]() { return process(); }); ...
}

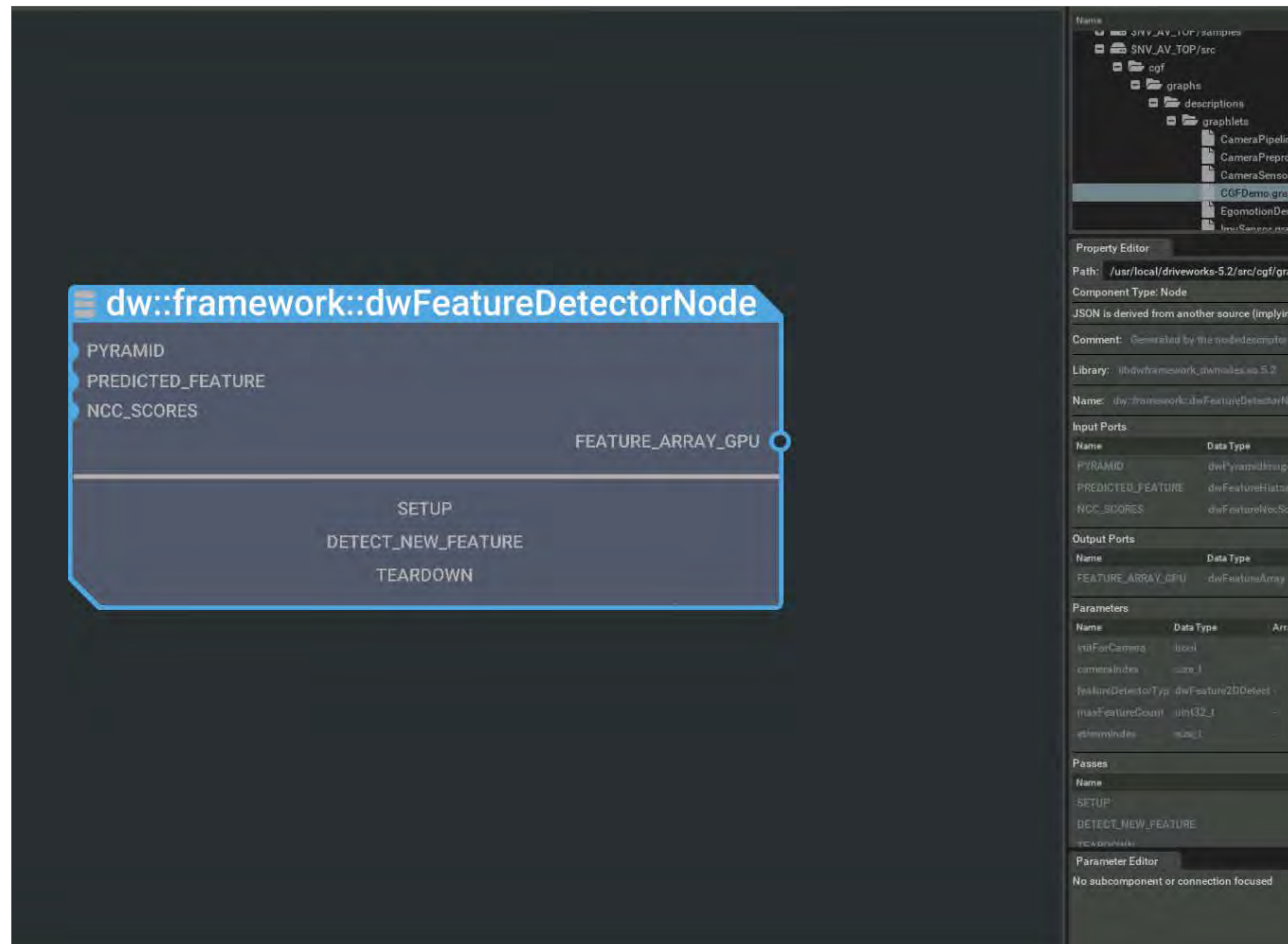
dwStatus MyNodeImpl::process() {...}
```

Typical Pipeline Design and Development Workflow

1. Create logical nodes
2. Create graphlets using the nodes
 - Specifying the data dependencies
3. **Generate stub / template code**
4. **Add the actual logic at appropriate places within the stub code**
5. **Compile and generate executable library**
6. Specify an application
 - a. Process layout
 - b. Distribution across execution engines (CPU, GPU, ...)
 - c. Timing Constraints (WCET)
7. Compile a schedule that satisfies the above specified constraints (data-dependency and timing)
8. Execute the application logic on the target
9. Profile. Tune. Repeat.

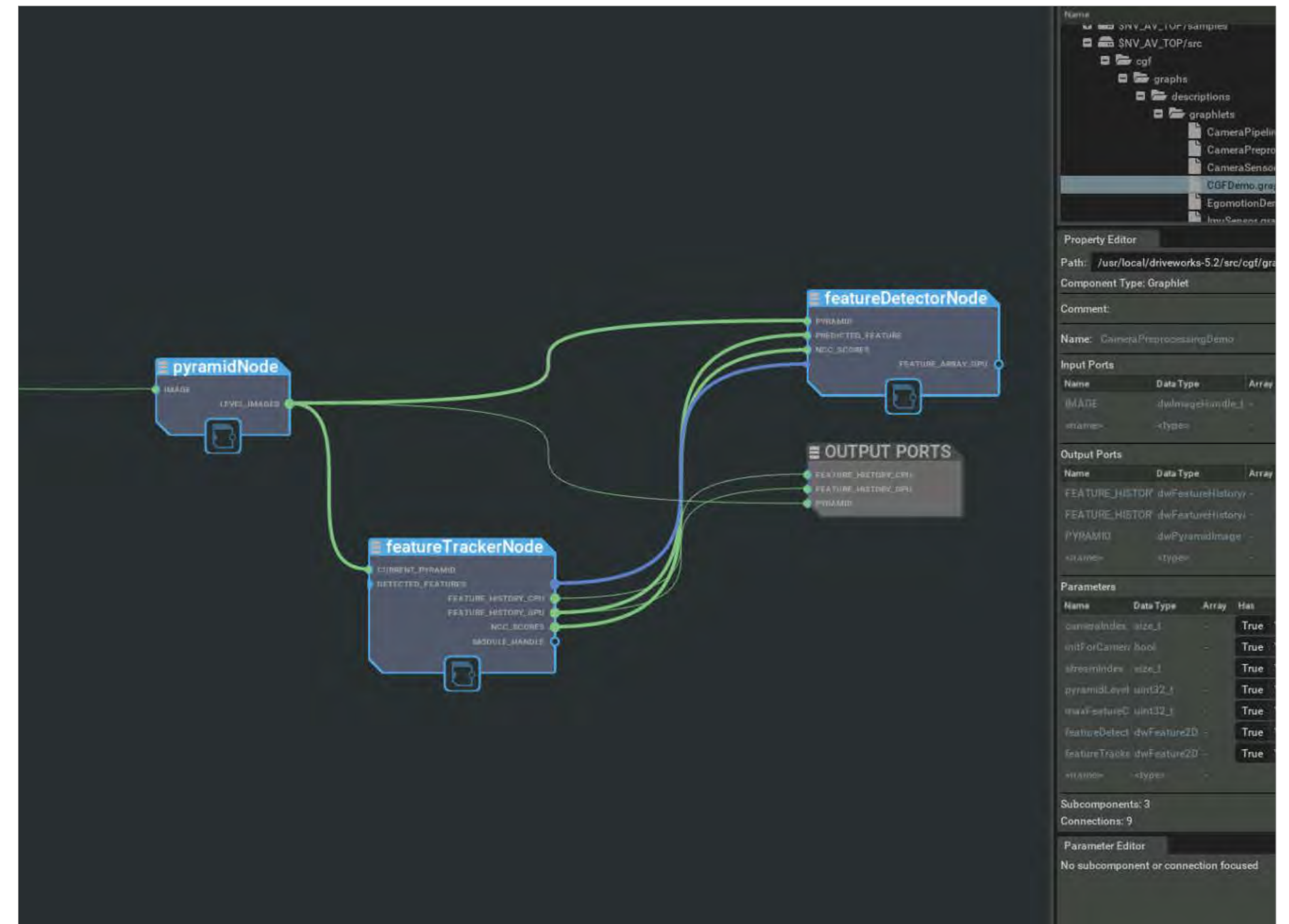
NOTE: Even without **writing/compiling any code** one can specify nodes, graphlets, schedule constraints, WCET constraints and generate a schedule, visualize it, and draw conclusions about expected performance.

DW Graph UI — Introduction to Nodes and Graphlets



Please Refer:

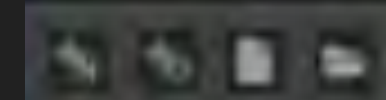
developer.nvidia.com/docs/drive/drive-os/6.0.6/public/driveworks-nvcgf/cgf_details_node.html



Please Refer:

developer.nvidia.com/docs/drive/drive-os/6.0.6/public/driveworks-nvcgf/cgf_details_graphlet.html

File Help



File Browser

Name

File Browser

- SNV_AV_TOP/apps
- SNV_AV_TOP/private
- SNV_AV_TOP/samples
- SNV_AV_TOP/src
 - cgt
 - graphs
 - descriptions
 - graphlets
 - CameraPipeline.graphlet.json
 - CameraPreprocessingDemo.graphlet.json
 - CameraSensor.graphlet.json
 - CDPDemo.graphlet.json
 - EgomotionDemo.graphlet.json
 - InertialSensor.graphlet.json
 - RadarDopplerMotion.graphlet.json
 - RadarSensor.graphlet.json
 - Render.graphlet.json
 - RenderDemo.graphlet.json
 - SelfCalibrationDemo.graphlet.json
 - SensorServiceCameraSensor.graphlet.json
 - SensorServiceGpsSensor.graphlet.json
 - SensorServices.graphlet.json
 - SensorSync.graphlet.json

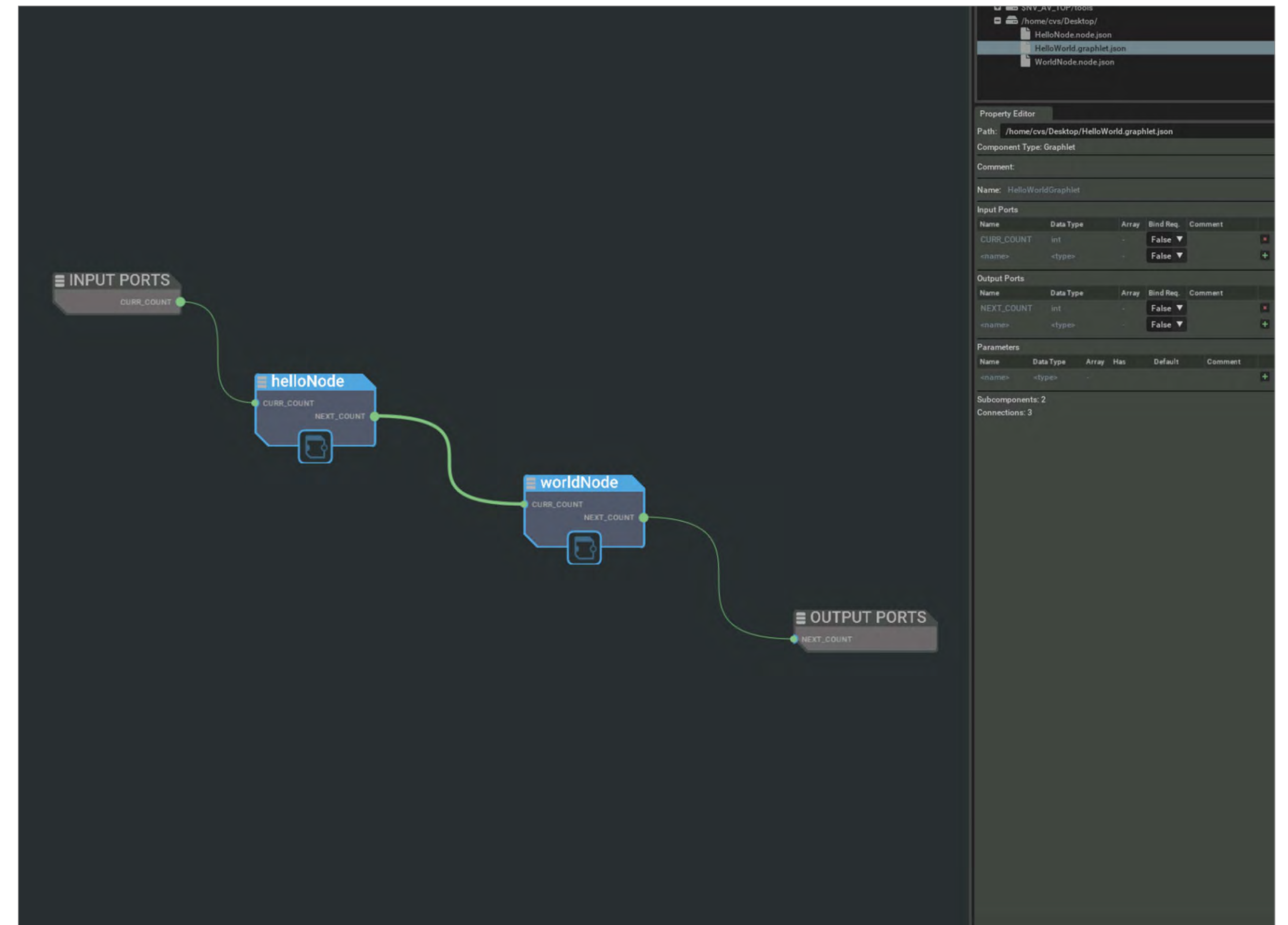
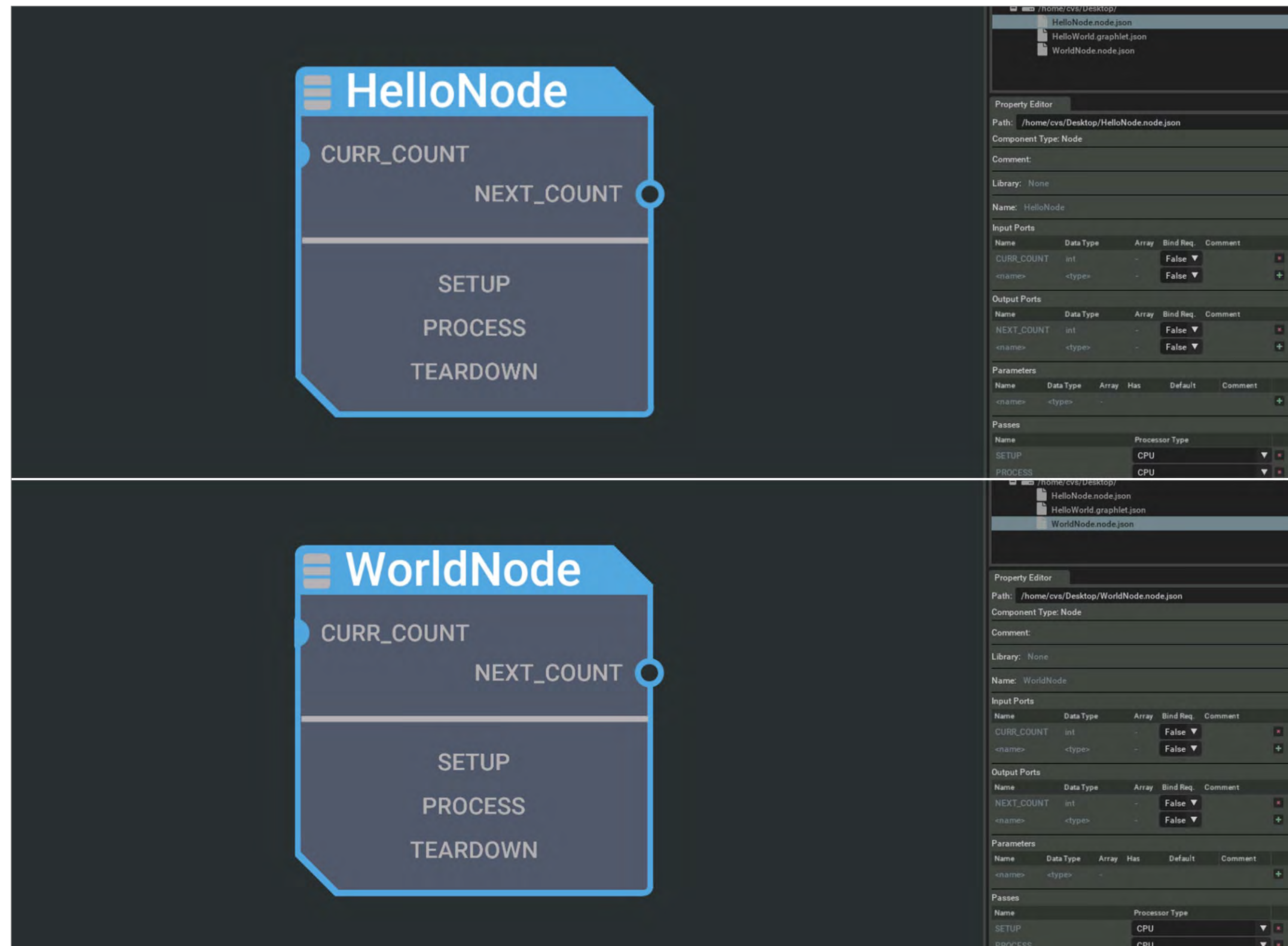
Property Editor

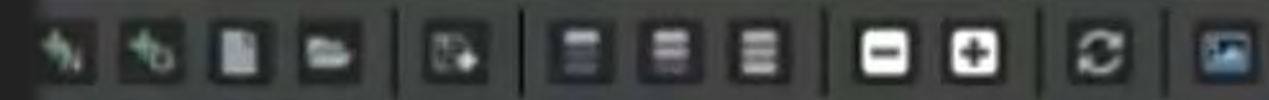
No component focused

Parameter Editor


No subcomponent or connection focused

DW Graph UI — Introduction to Nodes and Graphlets

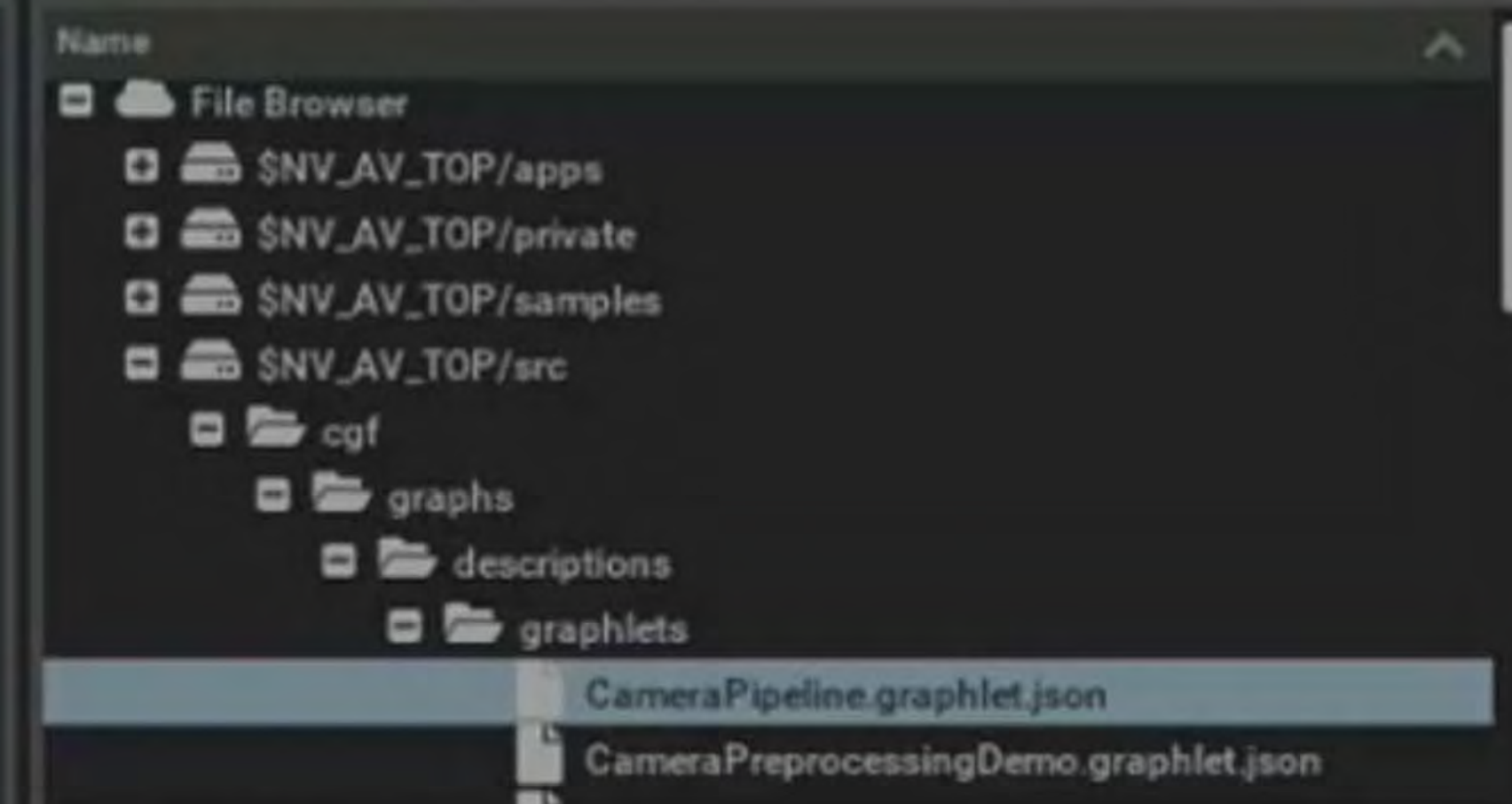




[G] CGFDemo (/usr/local/driveworks-5.2/s... [G] CameraPipeline (/usr/local/driveworks... [G] CameraPreprocessingDemo (/usr/local... [N] dw::framework:dwFeatureTrackerNod... [N] dw::framework:dwFeatureDetectorNo... [N] Unnamed1

 **Unnamed1**

File Browser



Property Editor

Path: None

Component Type: Node

Comment:

Library: None

Name: Unnamed1

Input Ports

Name	Data Type	Array	Bind Req.	Comment	
<name>	<type>	-	False ▼		+

Output Ports

Name	Data Type	Array	Bind Req.	Comment	
<name>	<type>	-	False ▼		+

Parameters

Name	Data Type	Array	Has	Default	Comment	
<name>	<type>	-				+

Passes

Name	Processor Type	
<name>	CPU ▼	+

Parameter Editor

No subcomponent or connection focused

Nodestub — Autogenerating Code for Custom Nodes

```
/usr/local/driveworks/tools/nodestub/nodestub.py \  
--output-path ./hello-world-cgf/ ./HelloNode.node.json dw::framework::ExceptionSafeProcessNode  
Generating files in: hello-world-cgf  
* /usr/local/driveworks/tools/nodestub/Node.thpp -> hello-world-cgf/HelloNode.hpp  
Unknown data type 'int', additional #include directives might be needed in the node header  
* /usr/local/driveworks/tools/nodestub/Node.tcpp -> hello-world-cgf/HelloNode.cpp  
* /usr/local/driveworks/tools/nodestub/NodeImpl.thpp -> hello-world-cgf/HelloNodeImpl.hpp  
* /usr/local/driveworks/tools/nodestub/NodeImpl.tcpp -> hello-world-cgf/HelloNodeImpl.cpp
```

```
/usr/local/driveworks/tools/nodestub/nodestub.py \  
--output-path ./hello-world-cgf/ ./WorldNode.node.json dw::framework::ExceptionSafeProcessNode  
Generating files in: hello-world-cgf  
* /usr/local/driveworks/tools/nodestub/Node.thpp -> hello-world-cgf/WorldNode.hpp  
Unknown data type 'int', additional #include directives might be needed in the node header  
* /usr/local/driveworks/tools/nodestub/Node.tcpp -> hello-world-cgf/WorldNode.cpp  
* /usr/local/driveworks/tools/nodestub/NodeImpl.thpp -> hello-world-cgf/WorldNodeImpl.hpp  
* /usr/local/driveworks/tools/nodestub/NodeImpl.tcpp -> hello-world-cgf/WorldNodeImpl.cpp
```


cvs@cvs-dt:~/Desktop\$ /usr/local/driveworks/tools/nodestub/nodestub.py --output-path ./hello-world-cgf/ ./WorldNode.node.json dw::framework::ExceptionSafeProcessNode

I




```
1 ///////////////////////////////////////////////////////////////////
2 //
3 // Notice
4 // ALL NVIDIA DESIGN SPECIFICATIONS AND CODE ("MATERIALS") ARE PROVIDED "AS IS" NVIDIA MAKES
5 // NO REPRESENTATIONS, WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO
6 // THE MATERIALS, AND EXPRESSLY DISCLAIMS ANY IMPLIED WARRANTIES OF NONINFRINGEMENT,
7 // MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8 //
9 // NVIDIA CORPORATION & AFFILIATES assumes no responsibility for the consequences of use of such
10 // information or for any infringement of patents or other rights of third parties that may
11 // result from its use. No license is granted by implication or otherwise under any patent
12 // or patent rights of NVIDIA CORPORATION & AFFILIATES. No third party distribution is allowed unless
13 // expressly authorized by NVIDIA. Details are subject to change without notice.
14 // This code supersedes and replaces all information previously supplied.
15 // NVIDIA CORPORATION & AFFILIATES products are not authorized for use as critical
16 // components in life support devices or systems without express written approval of
17 // NVIDIA CORPORATION & AFFILIATES.
18 //
19 // SPDX-FileCopyrightText: Copyright (c) 2022 NVIDIA CORPORATION & AFFILIATES. All rights reserved.
20 // SPDX-License-Identifier: LicenseRef-NvidiaProprietary
21 //
22 // NVIDIA CORPORATION, its affiliates and licensors retain all intellectual
23 // property and proprietary rights in and to this material, related
24 // documentation and any modifications thereto. Any use, reproduction,
25 // disclosure or distribution of this material and related documentation
26 // without an express license agreement from NVIDIA CORPORATION or
27 // its affiliates is strictly prohibited.
28 //
29 ///////////////////////////////////////////////////////////////////
30
31 #include "HelloNodeImpl.hpp"
32
33
34 constexpr char HelloNodeImpl::LOG_TAG[];
35
36 HelloNodeImpl::HelloNodeImpl(const dwContextHandle_t ctx)
37 {
38     initInputPorts();
39     initOutputPorts();
40     registerPasses();
41 }
42
43 HelloNodeImpl::~HelloNodeImpl()
44 {
45 }
46
47 void HelloNodeImpl::initInputPorts()
48 {
49     using namespace dw::core; // for operator"" _sv
50     NODE_INIT_INPUT_PORT("CURR_COUNT"_sv);
51 }
52
53 void HelloNodeImpl::initOutputPorts()
54 {
55     using namespace dw::core; // for operator"" _sv
56     {
57         dw::framework::parameter_traits<int>::SpecimenT ref{};
58         NODE_INIT_OUTPUT_PORT("NEXT_COUNT"_sv, ref);
59     }
60 }
```


Adding Custom Logic Within Autogenerated Code

```
70
71 dwStatus WorldNodeImpl::processPass()
72 {
73     |
74     return DW_SUCCESS;
75 }
76
```

Add logic in the auto-generated stub code

- `HelloNodeImpl.cpp`
- `WorldNodeImpl.cpp`

Please Refer:

developer.nvidia.com/docs/drive/drive-os/6.0.6/public/driveworks-nvcgf/cgf_tutorials_node.html

Especially section “Impl Source”

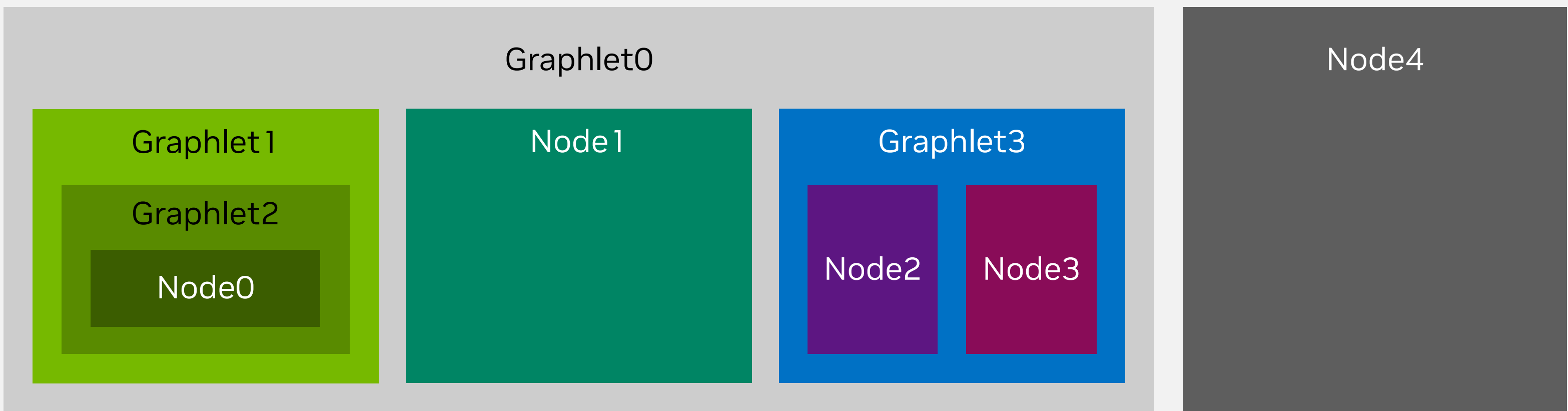
- [`NODE_GET_INPUT_PORT\(\)`](#)
- [`NODE_GET_OUTPUT_PORT\(\)`](#)

Compiling Custom Logic Within Custom Nodes

```
cv$@cvs-dt:~/Downloads/demo-6.0.2.0-linux/driveworks_workspace/build-linux-aarch64$ make clean
cv$@cvs-dt:~/Downloads/demo-6.0.2.0-linux/driveworks_workspace/build-linux-aarch64$ make cgf_custom_nodes -j8
[ 3%] Generating '_data' symbolic link
[ 7%] Building CXX object 3rdparty/src/lodepng/CMakeFiles/lodepng-src.dir/src/lodepng.cpp.o
[ 11%] Building CXX object src/framework/CMakeFiles/samples_allocator.dir/Allocator.cpp.o
[ 14%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/monitor.c.o
[ 14%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/init.c.o
[ 18%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/vulkan.c.o
[ 22%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/context.c.o
[ 22%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/input.c.o
[ 22%] Built target create-data-symlink
[ 22%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/window.c.o
[ 25%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/x11_init.c.o
[ 25%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/x11_monitor.c.o
[ 29%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/x11_window.c.o
[ 33%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/xkb_unicode.c.o
[ 33%] Linking CXX static library libsamples_allocator.a
[ 33%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/linux_joystick.c.o
[ 37%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/posix_time.c.o
[ 37%] Built target samples_allocator
[ 37%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/posix_tls.c.o
[ 40%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/glx_context.c.o
[ 40%] Building C object 3rdparty/src/glfw/CMakeFiles/glfw-x11.dir/3.2-screen/src/glx_context.c.o
    * * *
[ 81%] Building CXX object src/framework/CMakeFiles/samples_framework.dir/screenshooter.cpp.o
[ 85%] Building CXX object src/framework/CMakeFiles/samples_framework.dir/RenderUtils.cpp.o
[ 88%] Building CXX object src/framework/CMakeFiles/samples_framework.dir/SamplesDataPath.cpp.o
[ 88%] Building CXX object src/framework/CMakeFiles/samples_framework.dir/WindowEGL.cpp.o
[ 92%] Building CXX object src/framework/CMakeFiles/samples_framework.dir/WindowLinuxEGL.cpp.o
[ 92%] Linking CXX static library libsamples_framework.a
[ 92%] Built target samples_framework
[ 96%] Building CXX object src/cgf_nodes/CMakeFiles/cgf_custom_nodes.dir/HelloWorldNodeImpl.cpp.o
[ 96%] Building CXX object src/cgf_nodes/CMakeFiles/cgf_custom_nodes.dir/HelloWorldNode.cpp.o
[ 96%] Building CXX object src/cgf_nodes/CMakeFiles/cgf_custom_nodes.dir/SumNodeImpl.cpp.o
[100%] Building CXX object src/cgf_nodes/CMakeFiles/cgf_custom_nodes.dir/SumNode.cpp.o
[100%] Linking CXX shared library libcgf_custom_nodes.so
[100%] Built target cgf_custom_nodes
cv$@cvs-dt:~/Downloads/demo-6.0.2.0-linux/driveworks_workspace/build-linux-aarch64$ ll ./src/cgf_nodes/libcgf_custom_nodes.so
-rwxrwxr-x 1 cvs cvs 412400 Jul 28 12:19 ./src/cgf_nodes/libcgf_custom_nodes.so
```


Introduction to an Application

Application

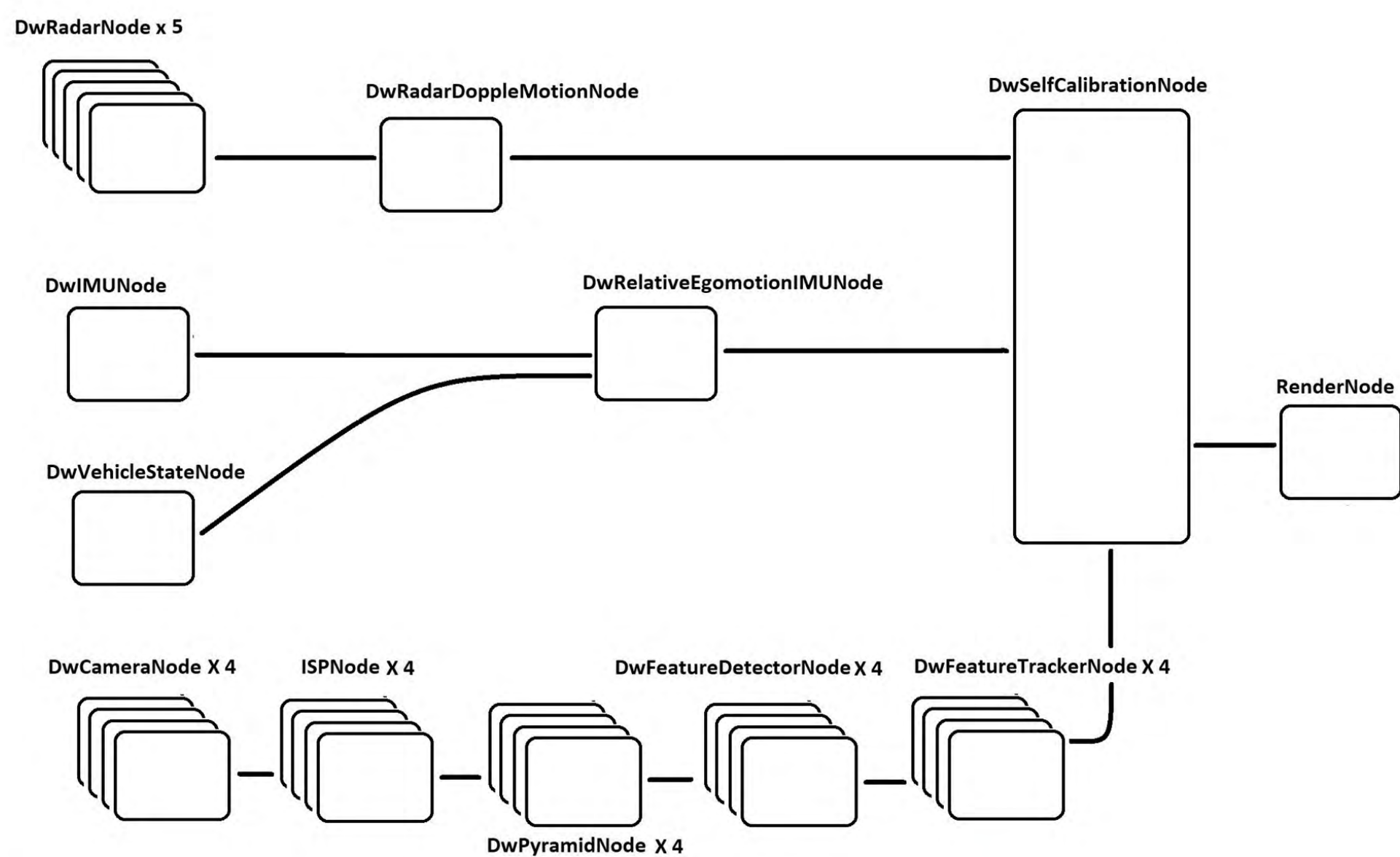


An application is composed of..

- a **data pipeline** described by one or multiple graphlets defining a DAG
- an **execution model** described by the scheduling and the process layout
- various **configuration parameters** eg. configuring logging

```
1 {
2   "version": "2.0",
3   "name": "CGFDemo",
4   "logSpec": "file://c4d4:((logpath))/((appName)).log",
5   "parameters": {},
6   "graphlet": "CGFDemo",
7   "graphlet": ".../graphlets/CGFDemo.graphlet.json",
8   "schedule": ".../CGFDemo.schedule.json",
9   "apps": {
10    "sen": {
11      "binfile": "senillbase",
12      "logSpec": "file://(logpath))/((appName)).log",
13    },
14    "sensor_sync_server": {
15      "binfile": "sensor_sync_server",
16      "logSpec": "file://(logpath))/((appName)).log",
17      "resources": {
18        "cpu": "0",
19        "mem": "1000000",
20        "radarGroup": "0",
21        "imgpsCardGroup": "0"
22      }
23    },
24    "stm_master": {
25      "binfile": "stm_master",
26      "logSpec": "file://(logpath))/((appName)).log",
27      "resources": {
28        "cpu": "0",
29        "mem": "1000000",
30        "log": "stm_report",
31        "cpu": "0"
32      }
33    },
34    "stm_slave": {
35      "binfile": "stm_slave",
36      "logSpec": "file://(logpath))/((appName)).log",
37      "resources": {
38        "cpu": "0",
39        "mem": "1000000",
40        "log": "stm_report",
41        "cpu": "0"
42      }
43    }
44  },
45  "resources": {
46    "cpu": "0",
47    "mem": "1000000",
48    "radarGroup": "0",
49    "imgpsCardGroup": "0"
50  },
51  "schedule": ".../CGFDemo.schedule.json",
52  "apps": {
53    "sen": {
54      "binfile": "senillbase",
55      "logSpec": "file://(logpath))/((appName)).log",
56    },
57    "sensor_sync_server": {
58      "binfile": "sensor_sync_server",
59      "logSpec": "file://(logpath))/((appName)).log",
60      "resources": {
61        "cpu": "0",
62        "mem": "1000000",
63        "radarGroup": "0",
64        "imgpsCardGroup": "0"
65      }
66    },
67    "stm_master": {
68      "binfile": "stm_master",
69      "logSpec": "file://(logpath))/((appName)).log",
70      "resources": {
71        "cpu": "0",
72        "mem": "1000000",
73        "log": "stm_report",
74        "cpu": "0"
75      }
76    },
77    "stm_slave": {
78      "binfile": "stm_slave",
79      "logSpec": "file://(logpath))/((appName)).log",
80      "resources": {
81        "cpu": "0",
82        "mem": "1000000",
83        "log": "stm_report",
84        "cpu": "0"
85      }
86    }
87  },
88  "resources": {
89    "cpu": "0",
90    "mem": "1000000",
91    "radarGroup": "0",
92    "imgpsCardGroup": "0"
93  },
94  "schedule": ".../CGFDemo.schedule.json",
95  "apps": {
96    "sen": {
97      "binfile": "senillbase",
98      "logSpec": "file://(logpath))/((appName)).log",
99    },
100   "sensor_sync_server": {
101     "binfile": "sensor_sync_server",
102     "logSpec": "file://(logpath))/((appName)).log",
103     "resources": {
104       "cpu": "0",
105       "mem": "1000000",
106       "radarGroup": "0",
107       "imgpsCardGroup": "0"
108     }
109   },
110   "stm_master": {
111     "binfile": "stm_master",
112     "logSpec": "file://(logpath))/((appName)).log",
113     "resources": {
114       "cpu": "0",
115       "mem": "1000000",
116       "log": "stm_report",
117       "cpu": "0"
118     }
119   },
120   "stm_slave": {
121     "binfile": "stm_slave",
122     "logSpec": "file://(logpath))/((appName)).log",
123     "resources": {
124       "cpu": "0",
125       "mem": "1000000",
126       "log": "stm_report",
127       "cpu": "0"
128     }
129   }
130 },
131 "resources": {
132   "cpu": "0",
133   "mem": "1000000",
134   "radarGroup": "0",
135   "imgpsCardGroup": "0"
136 },
137 "schedule": ".../CGFDemo.schedule.json",
138 "apps": {
139   "sen": {
140     "binfile": "senillbase",
141     "logSpec": "file://(logpath))/((appName)).log",
142   },
143   "sensor_sync_server": {
144     "binfile": "sensor_sync_server",
145     "logSpec": "file://(logpath))/((appName)).log",
146     "resources": {
147       "cpu": "0",
148       "mem": "1000000",
149       "radarGroup": "0",
150       "imgpsCardGroup": "0"
151     }
152   },
153   "stm_master": {
154     "binfile": "stm_master",
155     "logSpec": "file://(logpath))/((appName)).log",
156     "resources": {
157       "cpu": "0",
158       "mem": "1000000",
159       "log": "stm_report",
160       "cpu": "0"
161     }
162   },
163   "stm_slave": {
164     "binfile": "stm_slave",
165     "logSpec": "file://(logpath))/((appName)).log",
166     "resources": {
167       "cpu": "0",
168       "mem": "1000000",
169       "log": "stm_report",
170       "cpu": "0"
171     }
172   }
173 },
174 "resources": {
175   "cpu": "0",
176   "mem": "1000000",
177   "radarGroup": "0",
178   "imgpsCardGroup": "0"
179 },
180 "schedule": ".../CGFDemo.schedule.json",
181 "apps": {
182   "sen": {
183     "binfile": "senillbase",
184     "logSpec": "file://(logpath))/((appName)).log",
185   },
186   "sensor_sync_server": {
187     "binfile": "sensor_sync_server",
188     "logSpec": "file://(logpath))/((appName)).log",
189     "resources": {
190       "cpu": "0",
191       "mem": "1000000",
192       "radarGroup": "0",
193       "imgpsCardGroup": "0"
194     }
195   },
196   "stm_master": {
197     "binfile": "stm_master",
198     "logSpec": "file://(logpath))/((appName)).log",
199     "resources": {
200       "cpu": "0",
201       "mem": "1000000",
202       "log": "stm_report",
203       "cpu": "0"
204     }
205   },
206   "stm_slave": {
207     "binfile": "stm_slave",
208     "logSpec": "file://(logpath))/((appName)).log",
209     "resources": {
210       "cpu": "0",
211       "mem": "1000000",
212       "log": "stm_report",
213       "cpu": "0"
214     }
215   }
216 },
217 "resources": {
218   "cpu": "0",
219   "mem": "1000000",
220   "radarGroup": "0",
221   "imgpsCardGroup": "0"
222 },
223 "schedule": ".../CGFDemo.schedule.json",
224 "apps": {
225   "sen": {
226     "binfile": "senillbase",
227     "logSpec": "file://(logpath))/((appName)).log",
228   },
229   "sensor_sync_server": {
230     "binfile": "sensor_sync_server",
231     "logSpec": "file://(logpath))/((appName)).log",
232     "resources": {
233       "cpu": "0",
234       "mem": "1000000",
235       "radarGroup": "0",
236       "imgpsCardGroup": "0"
237     }
238   },
239   "stm_master": {
240     "binfile": "stm_master",
241     "logSpec": "file://(logpath))/((appName)).log",
242     "resources": {
243       "cpu": "0",
244       "mem": "1000000",
245       "log": "stm_report",
246       "cpu": "0"
247     }
248   },
249   "stm_slave": {
250     "binfile": "stm_slave",
251     "logSpec": "file://(logpath))/((appName)).log",
252     "resources": {
253       "cpu": "0",
254       "mem": "1000000",
255       "log": "stm_report",
256       "cpu": "0"
257     }
258   }
259 },
260 "resources": {
261   "cpu": "0",
262   "mem": "1000000",
263   "radarGroup": "0",
264   "imgpsCardGroup": "0"
265 },
266 "schedule": ".../CGFDemo.schedule.json",
267 "apps": {
268   "sen": {
269     "binfile": "senillbase",
270     "logSpec": "file://(logpath))/((appName)).log",
271   },
272   "sensor_sync_server": {
273     "binfile": "sensor_sync_server",
274     "logSpec": "file://(logpath))/((appName)).log",
275     "resources": {
276       "cpu": "0",
277       "mem": "1000000",
278       "radarGroup": "0",
279       "imgpsCardGroup": "0"
280     }
281   },
282   "stm_master": {
283     "binfile": "stm_master",
284     "logSpec": "file://(logpath))/((appName)).log",
285     "resources": {
286       "cpu": "0",
287       "mem": "1000000",
288       "log": "stm_report",
289       "cpu": "0"
290     }
291   },
292   "stm_slave": {
293     "binfile": "stm_slave",
294     "logSpec": "file://(logpath))/((appName)).log",
295     "resources": {
296       "cpu": "0",
297       "mem": "1000000",
298       "log": "stm_report",
299       "cpu": "0"
300     }
301   }
302 },
303 "resources": {
304   "cpu": "0",
305   "mem": "1000000",
306   "radarGroup": "0",
307   "imgpsCardGroup": "0"
308 },
309 "schedule": ".../CGFDemo.schedule.json",
310 "apps": {
311   "sen": {
312     "binfile": "senillbase",
313     "logSpec": "file://(logpath))/((appName)).log",
314   },
315   "sensor_sync_server": {
316     "binfile": "sensor_sync_server",
317     "logSpec": "file://(logpath))/((appName)).log",
318     "resources": {
319       "cpu": "0",
320       "mem": "1000000",
321       "radarGroup": "0",
322       "imgpsCardGroup": "0"
323     }
324   },
325   "stm_master": {
326     "binfile": "stm_master",
327     "logSpec": "file://(logpath))/((appName)).log",
328     "resources": {
329       "cpu": "0",
330       "mem": "1000000",
331       "log": "stm_report",
332       "cpu": "0"
333     }
334   },
335   "stm_slave": {
336     "binfile": "stm_slave",
337     "logSpec": "file://(logpath))/((appName)).log",
338     "resources": {
339       "cpu": "0",
340       "mem": "1000000",
341       "log": "stm_report",
342       "cpu": "0"
343     }
344   }
345 },
346 "resources": {
347   "cpu": "0",
348   "mem": "1000000",
349   "radarGroup": "0",
350   "imgpsCardGroup": "0"
351 },
352 "schedule": ".../CGFDemo.schedule.json",
353 "apps": {
354   "sen": {
355     "binfile": "senillbase",
356     "logSpec": "file://(logpath))/((appName)).log",
357   },
358   "sensor_sync_server": {
359     "binfile": "sensor_sync_server",
360     "logSpec": "file://(logpath))/((appName)).log",
361     "resources": {
362       "cpu": "0",
363       "mem": "1000000",
364       "radarGroup": "0",
365       "imgpsCardGroup": "0"
366     }
367   },
368   "stm_master": {
369     "binfile": "stm_master",
370     "logSpec": "file://(logpath))/((appName)).log",
371     "resources": {
372       "cpu": "0",
373       "mem": "1000000",
374       "log": "stm_report",
375       "cpu": "0"
376     }
377   },
378   "stm_slave": {
379     "binfile": "stm_slave",
380     "logSpec": "file://(logpath))/((appName)).log",
381     "resources": {
382       "cpu": "0",
383       "mem": "1000000",
384       "log": "stm_report",
385       "cpu": "0"
386     }
387   }
388 },
389 "resources": {
390   "cpu": "0",
391   "mem": "1000000",
392   "radarGroup": "0",
393   "imgpsCardGroup": "0"
394 },
395 "schedule": ".../CGFDemo.schedule.json",
396 "apps": {
397   "sen": {
398     "binfile": "senillbase",
399     "logSpec": "file://(logpath))/((appName)).log",
400   },
401   "sensor_sync_server": {
402     "binfile": "sensor_sync_server",
403     "logSpec": "file://(logpath))/((appName)).log",
404     "resources": {
405       "cpu": "0",
406       "mem": "1000000",
407       "radarGroup": "0",
408       "imgpsCardGroup": "0"
409     }
410   },
411   "stm_master": {
412     "binfile": "stm_master",
413     "logSpec": "file://(logpath))/((appName)).log",
414     "resources": {
415       "cpu": "0",
416       "mem": "1000000",
417       "log": "stm_report",
418       "cpu": "0"
419     }
420   },
421   "stm_slave": {
422     "binfile": "stm_slave",
423     "logSpec": "file://(logpath))/((appName)).log",
424     "resources": {
425       "cpu": "0",
426       "mem": "1000000",
427       "log": "stm_report",
428       "cpu": "0"
429     }
430   }
431 },
432 "resources": {
433   "cpu": "0",
434   "mem": "1000000",
435   "radarGroup": "0",
436   "imgpsCardGroup": "0"
437 },
438 "schedule": ".../CGFDemo.schedule.json",
439 "apps": {
440   "sen": {
441     "binfile": "senillbase",
442     "logSpec": "file://(logpath))/((appName)).log",
443   },
444   "sensor_sync_server": {
445     "binfile": "sensor_sync_server",
446     "logSpec": "file://(logpath))/((appName)).log",
447     "resources": {
448       "cpu": "0",
449       "mem": "1000000",
450       "radarGroup": "0",
451       "imgpsCardGroup": "0"
452     }
453   },
454   "stm_master": {
455     "binfile": "stm_master",
456     "logSpec": "file://(logpath))/((appName)).log",
457     "resources": {
458       "cpu": "0",
459       "mem": "1000000",
460       "log": "stm_report",
461       "cpu": "0"
462     }
463   },
464   "stm_slave": {
465     "binfile": "stm_slave",
466     "logSpec": "file://(logpath))/((appName)).log",
467     "resources": {
468       "cpu": "0",
469       "mem": "1000000",
470       "log": "stm_report",
471       "cpu": "0"
472     }
473   }
474 },
475 "resources": {
476   "cpu": "0",
477   "mem": "1000000",
478   "radarGroup": "0",
479   "imgpsCardGroup": "0"
480 },
481 "schedule": ".../CGFDemo.schedule.json",
482 "apps": {
483   "sen": {
484     "binfile": "senillbase",
485     "logSpec": "file://(logpath))/((appName)).log",
486   },
487   "sensor_sync_server": {
488     "binfile": "sensor_sync_server",
489     "logSpec": "file://(logpath))/((appName)).log",
490     "resources": {
491       "cpu": "0",
492       "mem": "1000000",
493       "radarGroup": "0",
494       "imgpsCardGroup": "0"
495     }
496   },
497   "stm_master": {
498     "binfile": "stm_master",
499     "logSpec": "file://(logpath))/((appName)).log",
500     "resources": {
501       "cpu": "0",
502       "mem": "1000000",
503       "log": "stm_report",
504       "cpu": "0"
505     }
506   },
507   "stm_slave": {
508     "binfile": "stm_slave",
509     "logSpec": "file://(logpath))/((appName)).log",
510     "resources": {
511       "cpu": "0",
512       "mem": "1000000",
513       "log": "stm_report",
514       "cpu": "0"
515     }
516   }
517 },
518 "resources": {
519   "cpu": "0",
520   "mem": "1000000",
521   "radarGroup": "0",
522   "imgpsCardGroup": "0"
523 },
524 "schedule": ".../CGFDemo.schedule.json",
525 "apps": {
526   "sen": {
527     "binfile": "senillbase",
528     "logSpec": "file://(logpath))/((appName)).log",
529   },
530   "sensor_sync_server": {
531     "binfile": "sensor_sync_server",
532     "logSpec": "file://(logpath))/((appName)).log",
533     "resources": {
534       "cpu": "0",
535       "mem": "1000000",
536       "radarGroup": "0",
537       "imgpsCardGroup": "0"
538     }
539   },
540   "stm_master": {
541     "binfile": "stm_master",
542     "logSpec": "file://(logpath))/((appName)).log",
543     "resources": {
544       "cpu": "0",
545       "mem": "1000000",
546       "log": "stm_report",
547       "cpu": "0"
548     }
549   },
550   "stm_slave": {
551     "binfile": "stm_slave",
552     "logSpec": "file://(logpath))/((appName)).log",
553     "resources": {
554       "cpu": "0",
555       "mem": "1000000",
556       "log": "stm_report",
557       "cpu": "0"
558     }
559   }
560 },
561 "resources": {
562   "cpu": "0",
563   "mem": "1000000",
564   "radarGroup": "0",
565   "imgpsCardGroup": "0"
566 },
567 "schedule": ".../CGFDemo.schedule.json",
568 "apps": {
569   "sen": {
570     "binfile": "senillbase",
571     "logSpec": "file://(logpath))/((appName)).log",
572   },
573   "sensor_sync_server": {
574     "binfile": "sensor_sync_server",
575     "logSpec": "file://(logpath))/((appName)).log",
576     "resources": {
577       "cpu": "0",
578       "mem": "1000000",
579       "radarGroup": "0",
580       "imgpsCardGroup": "0"
581     }
582   },
583   "stm_master": {
584     "binfile": "stm_master",
585     "logSpec": "file://(logpath))/((appName)).log",
586     "resources": {
587       "cpu": "0",
588       "mem": "1000000",
589       "log": "stm_report",
590       "cpu": "0"
591     }
592   },
593   "stm_slave": {
594     "binfile": "stm_slave",
595     "logSpec": "file://(logpath))/((appName)).log",
596     "resources": {
597       "cpu": "0",
598       "mem": "1000000",
599       "log": "stm_report",
600       "cpu": "0"
601     }
602   }
603 },
604 "resources": {
605   "cpu": "0",
606   "mem": "1000000",
607   "radarGroup": "0",
608   "imgpsCardGroup": "0"
609 },
610 "schedule": ".../CGFDemo.schedule.json",
611 "apps": {
612   "sen": {
613     "binfile": "senillbase",
614     "logSpec": "file://(logpath))/((appName)).log",
615   },
616   "sensor_sync_server": {
617     "binfile": "sensor_sync_server",
618     "logSpec": "file://(logpath))/((appName)).log",
619     "resources": {
620       "cpu": "0",
621       "mem": "1000000",
622       "radarGroup": "0",
623       "imgpsCardGroup": "0"
624     }
625   },
626   "stm_master": {
627     "binfile": "stm_master",
628     "logSpec": "file://(logpath))/((appName)).log",
629     "resources": {
630       "cpu": "0",
631       "mem": "1000000",
632       "log": "stm_report",
633       "cpu": "0"
634     }
635   },
636   "stm_slave": {
637     "binfile": "stm_slave",
638     "logSpec": "file://(logpath))/((appName)).log",
639     "resources": {
640       "cpu": "0",
641       "mem": "1000000",
642       "log": "stm_report",
643       "cpu": "0"
644     }
645   }
646 },
647 "resources": {
648   "cpu": "0",
649   "mem": "1000000",
650   "radarGroup": "0",
651   "imgpsCardGroup": "0"
652 },
653 "schedule": ".../CGFDemo.schedule.json",
654 "apps": {
655   "sen": {
656     "binfile": "senillbase",
657     "logSpec": "file://(logpath))/((appName)).log",
658   },
659   "sensor_sync_server": {
660     "binfile": "sensor_sync_server",
661     "logSpec": "file://(logpath))/((appName)).log",
662     "resources": {
663       "cpu": "0",
664       "mem": "1000000",
665       "radarGroup": "0",
666       "imgpsCardGroup": "0"
667     }
668   },
669   "stm_master": {
670     "binfile": "stm_master",
671     "logSpec": "file://(logpath))/((appName)).log",
672     "resources": {
673       "cpu": "0",
674       "mem": "1000000",
675       "log": "stm_report",
676       "cpu": "0"
677     }
678   },
679   "stm_slave": {
680     "binfile": "stm_slave",
681     "logSpec": "file://(logpath))/((appName)).log",
682     "resources": {
683       "cpu": "0",
684       "mem": "1000000",
685       "log": "stm_report",
686       "cpu": "0"
687     }
688   }
689 },
690 "resources": {
691   "cpu": "0",
692   "mem": "1000000",
693   "radarGroup": "0",
694   "imgpsCardGroup": "0"
695 },
696 "schedule": ".../CGFDemo.schedule.json",
697 "apps": {
698   "sen": {
699     "binfile": "senillbase",
700     "logSpec": "file://(logpath))/((appName)).log",
701   },
702   "sensor_sync_server": {
703     "binfile": "sensor_sync_server",
704     "logSpec": "file://(logpath))/((appName)).log",
705     "resources": {
706       "cpu": "0",
707       "mem": "1000000",
708       "radarGroup": "0",
709       "imgpsCardGroup": "0"
710     }
711   },
712   "stm_master": {
713     "binfile": "stm_master",
714     "logSpec": "file://(logpath))/((appName)).log",
715     "resources": {
716       "cpu": "0",
717       "mem": "1000000",
718       "log": "stm_report",
719       "cpu": "0"
720     }
721   },
722   "stm_slave": {
723     "binfile": "stm_slave",
724     "logSpec": "file://(logpath))/((appName)).log",
725     "resources": {
726       "cpu": "0",
727       "mem": "1000000",
728       "log": "stm_report",
729       "cpu": "0"
730     }
731   }
732 },
733 "resources": {
734   "cpu": "0",
735   "mem": "1000000",
736   "radarGroup": "0",
737   "imgpsCardGroup": "0"
738 },
739 "schedule": ".../CGFDemo.schedule.json",
740 "apps": {
741   "sen": {
742     "binfile": "senillbase",
743     "logSpec": "file://(logpath))/((appName)).log",
744   },
745   "sensor_sync_server": {
746     "binfile": "sensor_sync_server",
747     "logSpec": "file://(logpath))/((appName)).log",
748     "resources": {
749       "cpu": "0",
750       "mem": "1000000",
751       "radarGroup": "0",
752       "imgpsCardGroup": "0"
753     }
754   },
755   "stm_master": {
756     "binfile": "stm_master",
757     "logSpec": "file://(logpath))/((appName)).log",
758     "resources": {
759       "cpu": "0",
760       "mem": "1000000",
761       "log": "stm_report",
762       "cpu": "0"
763     }
764   },
765   "stm_slave": {
766     "binfile": "stm_slave",
767     "logSpec": "file://(logpath))/((appName)).log",
768     "resources": {
769       "cpu": "0",
770       "mem": "1000000",
771       "log": "stm_report",
772       "cpu": "0"
773     }
774   }
775 },
776 "resources": {
777   "cpu": "0",
778   "mem": "1000000",
779   "radarGroup": "0",
780   "imgpsCardGroup": "0"
781 },
782 "schedule": ".../CGFDemo.schedule.json",
783 "apps": {
784   "sen": {
785     "binfile": "senillbase",
786     "logSpec": "file://(logpath))/((appName)).log",
787   },
788   "sensor_sync_server": {
789     "binfile": "sensor_sync_server",
790     "logSpec": "file://(logpath))/((appName)).log",
791     "resources": {
792       "cpu": "0",
793       "mem": "1000000",
794       "radarGroup": "0",
795       "imgpsCardGroup": "0"
796     }
797   },
798   "stm_master": {
799     "binfile": "stm_master",
800     "logSpec": "file://(logpath))/((appName)).log",
801     "resources": {
802       "cpu": "0",
803       "mem": "1000000",
804       "log": "stm_report",
805       "cpu": "0"
806     }
807   },
808   "stm_slave": {
809     "binfile": "stm_slave",
810     "logSpec": "file://(logpath))/((appName)).log",
811     "resources": {
812       "cpu": "0",
813       "mem": "1000000",
814       "log": "stm_report",
815       "cpu": "0"
816     }
817   }
818 },
819 "resources": {
820   "cpu": "0",
821   "mem": "1000000",
822   "radarGroup": "0",
823   "imgpsCardGroup": "0"
824 },
825 "schedule": ".../CGFDemo.schedule.json",
826 "apps": {
827   "sen": {
828     "binfile": "senillbase",
829     "logSpec": "file://(logpath))/((appName)).log",
830   },
831   "sensor_sync_server": {
832     "binfile": "sensor_sync_server",
833     "logSpec": "file://(logpath))/((appName)).log",
834     "resources": {
835       "cpu": "0",
836       "mem": "1000000",
837       "radarGroup": "0",
838       "imgpsCardGroup": "0"
839     }
840   },
841   "stm_master": {
842     "binfile": "stm_master",
843     "logSpec": "file://(logpath))/((appName)).log",
844     "resources": {
845       "cpu": "0",
846       "mem": "1000000",
847       "log": "stm_report",
848       "cpu": "0"
849     }
850   },
851   "stm_slave": {
852     "binfile": "stm_slave",
853     "logSpec": "file://(logpath))/((appName)).log",
854     "resources": {
855       "cpu": "0",
856       "mem": "1000000",
857       "log": "stm_report",
858       "cpu": "0"
859     }
860   }
861 },
862 "resources": {
863   "cpu": "0",
864   "mem": "1000000",
865   "radarGroup": "0",
866   "imgpsCardGroup": "0"
867 },
868 "schedule": ".../CGFDemo.schedule.json",
869 "apps": {
870   "sen": {
871     "binfile": "senillbase",
872     "logSpec": "file://(logpath))/((appName)).log",
873   },
874   "sensor_sync_server": {
875     "binfile": "sensor_sync_server",
876     "logSpec": "file://(logpath))/((appName)).log",
877     "resources": {
878       "cpu": "0",
879       "mem": "1000000",
880       "radarGroup": "0",
881       "imgpsCardGroup": "0"
882     }
883   },
884   "stm_master": {
885     "binfile": "stm_master",
886     "logSpec": "file://(logpath))/((appName)).log",
887     "resources": {
888       "cpu": "0",
889       "mem": "1000000",
890       "log": "stm_report",
891       "cpu": "0"
892     }
893   },
894   "stm_slave": {
895     "binfile": "stm_slave",
896     "logSpec": "file://(logpath))/((appName)).log",
897     "resources": {
898       "cpu": "0",
899       "mem": "1000000",
900       "log": "stm_report",
901       "cpu": "0"
902     }
903   }
904 },
905 "resources": {
906   "cpu": "0",
907   "mem": "1000000",
908   "radarGroup": "0",
909   "imgpsCardGroup": "0"
910 },
911 "schedule": ".../CGFDemo.schedule.json",
912 "apps": {
913   "sen": {
914     "binfile": "senillbase",
915     "logSpec": "file://(logpath))/((appName)).log",
916   },
917   "sensor_sync_server": {
918     "binfile": "sensor_sync_server",
919     "logSpec": "file://(logpath))/((appName)).log",
920     "resources": {
921       "cpu": "0",
922       "mem": "1000000",
923       "radarGroup": "0",
924       "imgpsCardGroup": "0"
925     }
926   },
927   "stm_master": {
928     "binfile": "stm_master",
929     "logSpec": "file://(logpath))/((appName)).log",
930     "resources": {
931       "cpu": "0",
932       "mem": "1000000",
933       "log": "stm_report",
934       "cpu": "0"
935     }
936   },
937   "stm_slave": {
938     "binfile": "stm_slave",
939     "logSpec": "file://(logpath))/((appName)).log",
940     "resources": {
941       "cpu": "0",
942       "mem": "1000000",
943       "log": "stm_report",
944       "cpu": "0"
945     }
946   }
947 },
948 "resources": {
949   "cpu": "0",
950   "mem": "1000000",
951   "radarGroup": "0",
952   "imgpsCardGroup": "0"
953 },
954 "schedule": ".../CGFDemo.schedule.json",
955 "apps": {
956   "sen": {
957     "binfile": "senillbase",
958     "logSpec": "file://(logpath))/((appName)).log",
959   },
960   "sensor_sync_server": {
961     "binfile": "sensor_sync_server",
962     "logSpec": "file://(logpath))/((appName)).log",
963     "resources": {
964       "cpu": "0",
965       "mem": "1000000",
966       "radarGroup": "0",
967       "imgpsCardGroup": "0"
968     }
969   },
970   "stm_master": {
971     "binfile": "stm_master",
972     "logSpec": "file://(logpath))/((appName)).log",
973     "resources": {
974       "cpu": "0",
975       "mem": "1000000",
976       "log": "stm_report",
977       "cpu": "0"
978     }
979   },
980   "stm_slave": {
981     "binfile": "stm_slave",
982     "logSpec": "file://(logpath))/((appName)).log",
983     "resources": {
984       "cpu": "0",
985       "mem": "1000000",
986       "log": "stm_report",
987       "cpu": "0"
988     }
989   }
990 },
991 "resources": {
992   "cpu": "0",
993   "mem": "1000000",
994   "radarGroup": "0",
995   "imgpsCardGroup": "0"
996 },
997 "schedule": ".../CGFDemo.schedule.json",
998 "apps": {
999   "sen": {
1000     "binfile": "senillbase",
1001     "logSpec": "file://(logpath))/((appName)).log",
1002   },
1003   "sensor_sync_server": {
1004     "binfile": "sensor_sync_server",
1005     "logSpec": "file://(logpath))/((appName)).log",
1006     "resources": {
1007       "cpu": "0",
1008       "mem": "1000000",
1009       "radarGroup": "0",
1010       "imgpsCardGroup": "0"
1011     }
1012   },
1013   "stm_master": {
1014     "binfile": "stm_master",
1015     "logSpec": "file://(logpath))/((appName)).log",
1016     "resources": {
1017       "cpu": "0",
1018       "mem": "1000000",
1019       "log": "stm_report",
1020       "cpu": "0"
1021     }
1022   },
1023   "stm_slave": {
1024     "binfile": "stm_slave",
1025     "logSpec": "file://
```


CGFDemo Application



Nodes in CGFDemo pipeline

- **RenderingCGFDemoNode**: Rendering node for the demo pipeline. The node is described by the `RenderingCGFDemoNode.node.json` file
- **dwRadarNode**: 5 instantiations of this node to create 5 radar inputs. Each node is described by the `dwRadarNode.node.json` file
- **dwIMUNode**: 1 instantiation of this node to create 1 IMU input. This node is described by the `dwIMUNode.node.json` file
- **dwVehicleStateNode**: 1 instantiation of this node, described by the `dwVehicleStateNode.node.json` file
- **dwCameraNode**: 4 instantiations of this node to create four camera inputs. The node is described by the `dwCameraNode.node.json` file
- **SensorSyncNode**: this node is responsible for sensor inputs synchronization. This node is described by the `SensorSyncNode.node.json` file
- **dwRelativeEgomotionIMUNode**: this node is responsible to generate egomotion, described by the `dwRelativeEgomotionIMUNode.node.json` file
- **dwRadarDopplerMotionNode**: this node post processes radar sensor outputs for self-calibration, described by the `dwRadarDopplerMotionNode.node.json` file
- **dwPyramidNode**: this node prepares the images in pyramid representation for feature detection purposes, described in the `dwPyramidNode.node.json` file
- **dwFeatureDetectionNode**: this node is responsible for feature detection, described by the `dwFeatureDetectionNode.node.json` file
- **dwFeatureTrackerNode**: this node is responsible for feature tracking, described by the `dwFeatureTrackerNode.node.json` file
- **dwSelfCalibrationNode**: this node is responsible to generate self-calibration results, described by the `dwSelfCalibrationNode.node.json` file

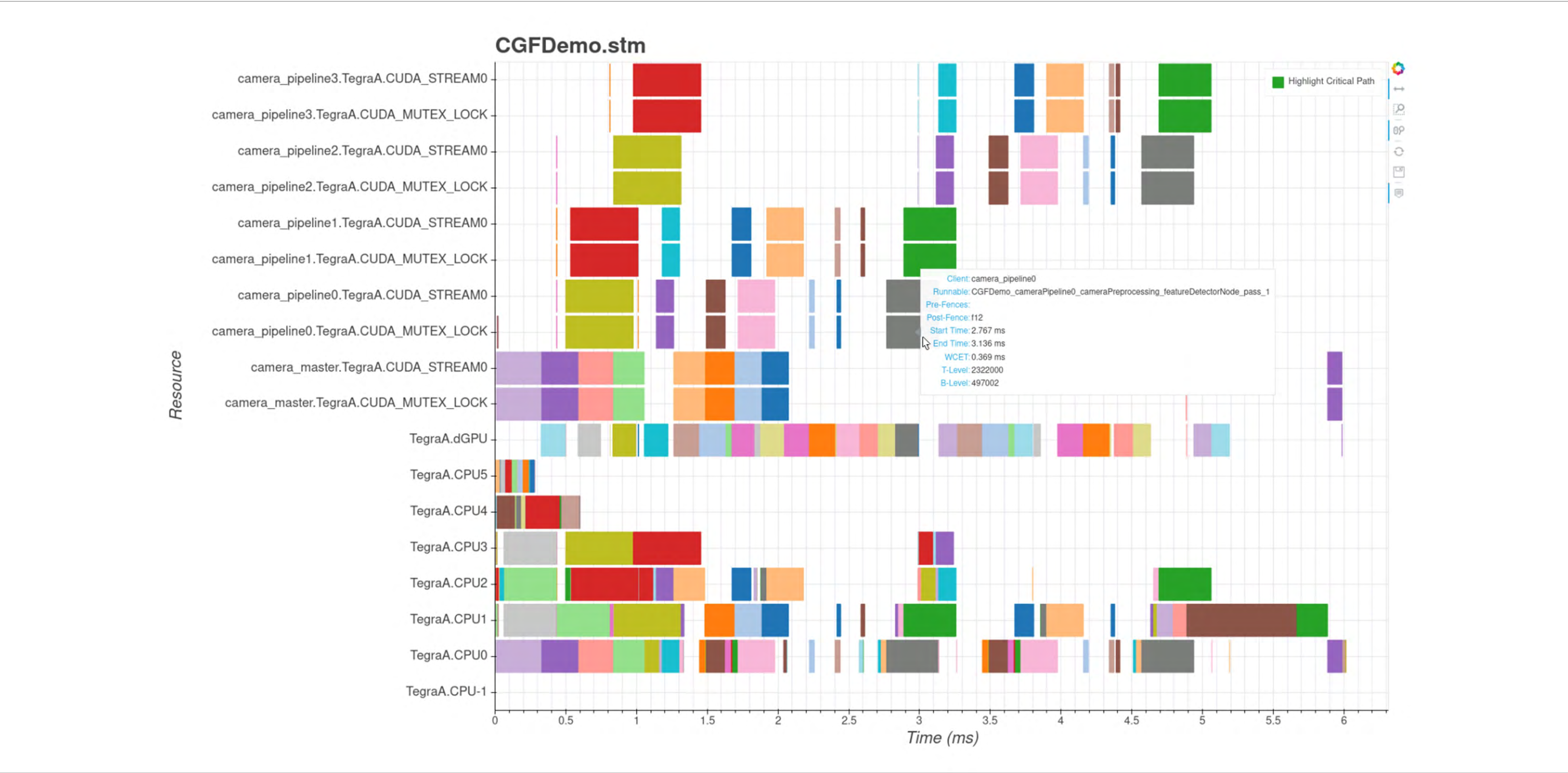
Graphlets in CGFDemo pipeline

- **RenderDemo graphlet**: this graphlet contains `RenderingCGFDemoNode` node
- **RadarSensor graphlet**: this graphlet contains `dwRadarnode` node
- **ImuSensor graphlet**: this graphlet contains `dwIMUNode` node
- **VehicleStateConsumer graphlet**: this graphlet contains `dwVehicleStatenode` node
- **CameraSensor graphlet**: this graphlet contains `dwCameraNode` and `ISPNode` nodes. Camera frames are fed through `dwCameraNode` and `ISPNode` to provide demosaiced image outputs to post processing in later pipeline
- **SensorSync graphlet**: this graphlet contains `SensorSyncNode` node
- **EgomotionDemo graphlet**: this graphlet contains two `dwRelativeEgomotionIMUNode` nodes. One of the egomotion is responsible for outputting egomotion state as odometry whereas the other egomotion node generates egomotion state that later feeds into self calibration graphlet
- **RadarDopplerMotion graphlet**: this graphlet contains eight instantiation of `dwRadarDopplerMotionNodes`
- **CameraPreprocessingDemo graphlet**: this graphlet contains `dwPyramidNode`, `dwFeatureDetectionNode`, and `dwFeatureTrackerNode`. The data feeds through these three nodes to produce feature tracking that are fed into self calibration and rendering nodes
- **CameraPipeline graphlet**: this graphlet contains three sub-graphlets, `CameraSensor`, `CameraPreprocessingDemo`, and `CameraObjectDetectorDemo` graphlets
- **SelfCalibrationDemo graphlet**: this graphlet contains `dwSelfCalibrationNode` node
- **CGFDemo graphlet**: this graphlet is the main graphlet that connects all graphlets in this demo

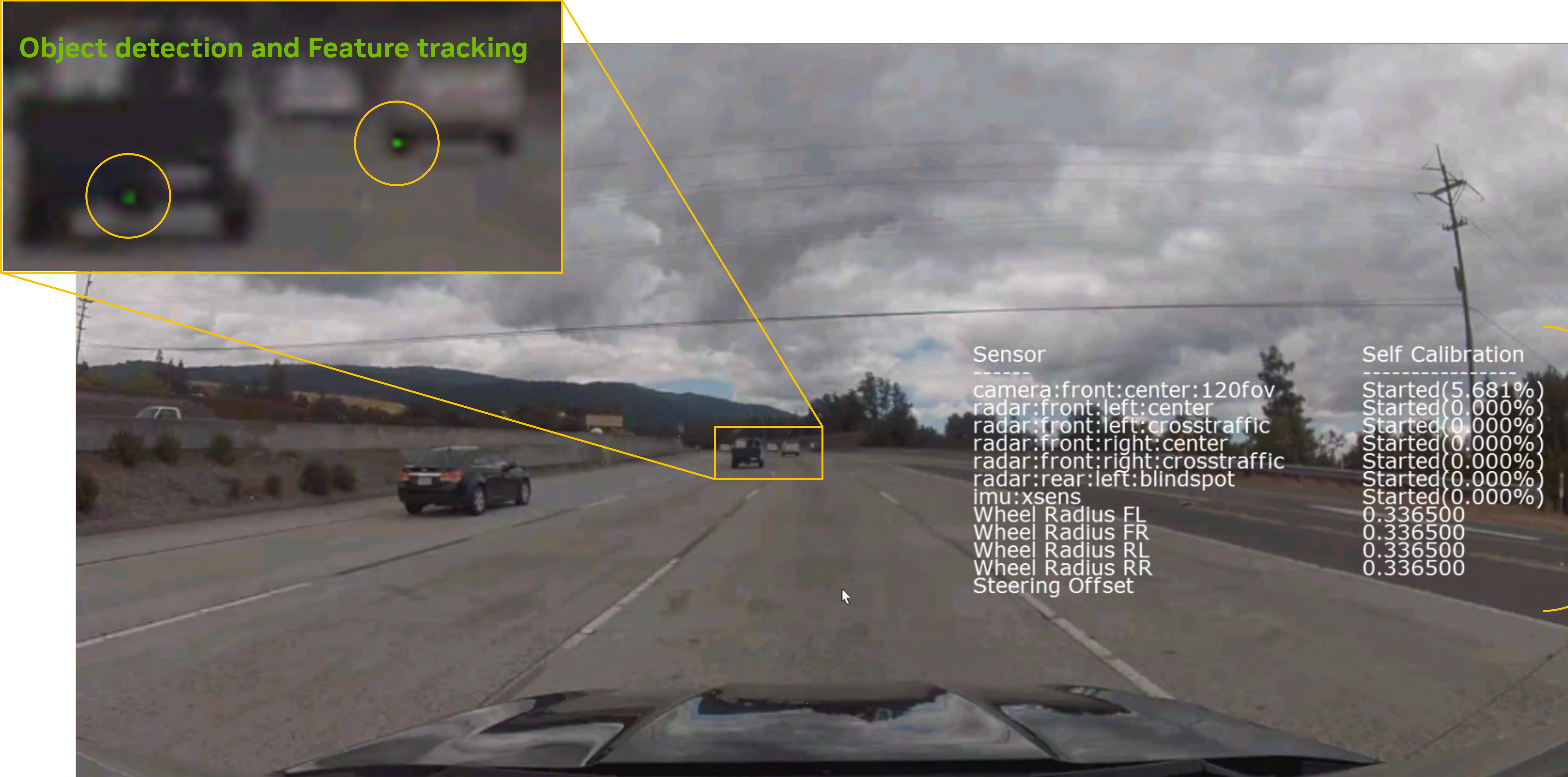
Please Refer:

developer.nvidia.com/docs/drive/drive-os/6.0.6/public/driveworks-nvcgf/cgf_samples_demo.html

Static Visualization — STMVIZSCHEDULE



CGFDemo Application

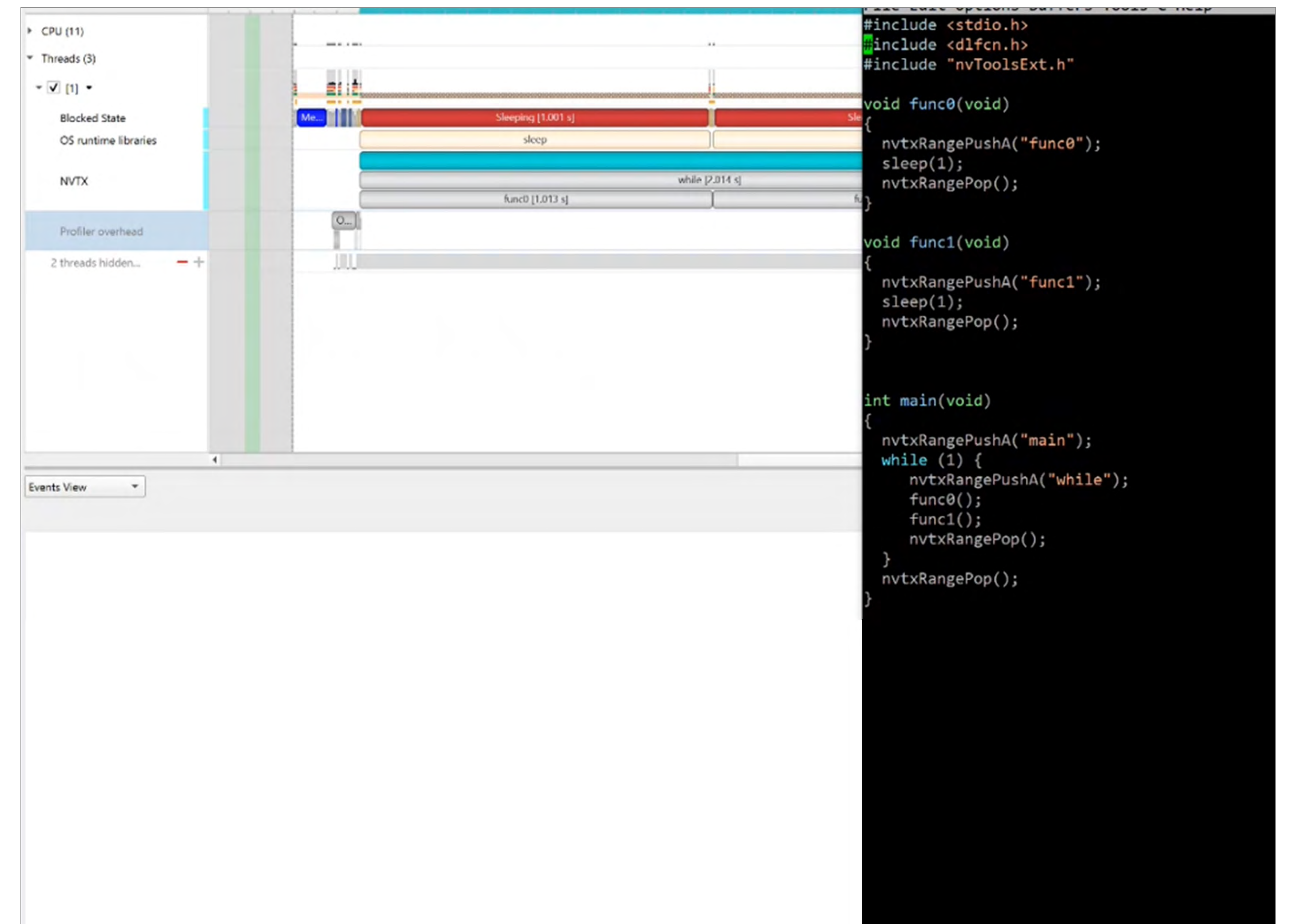
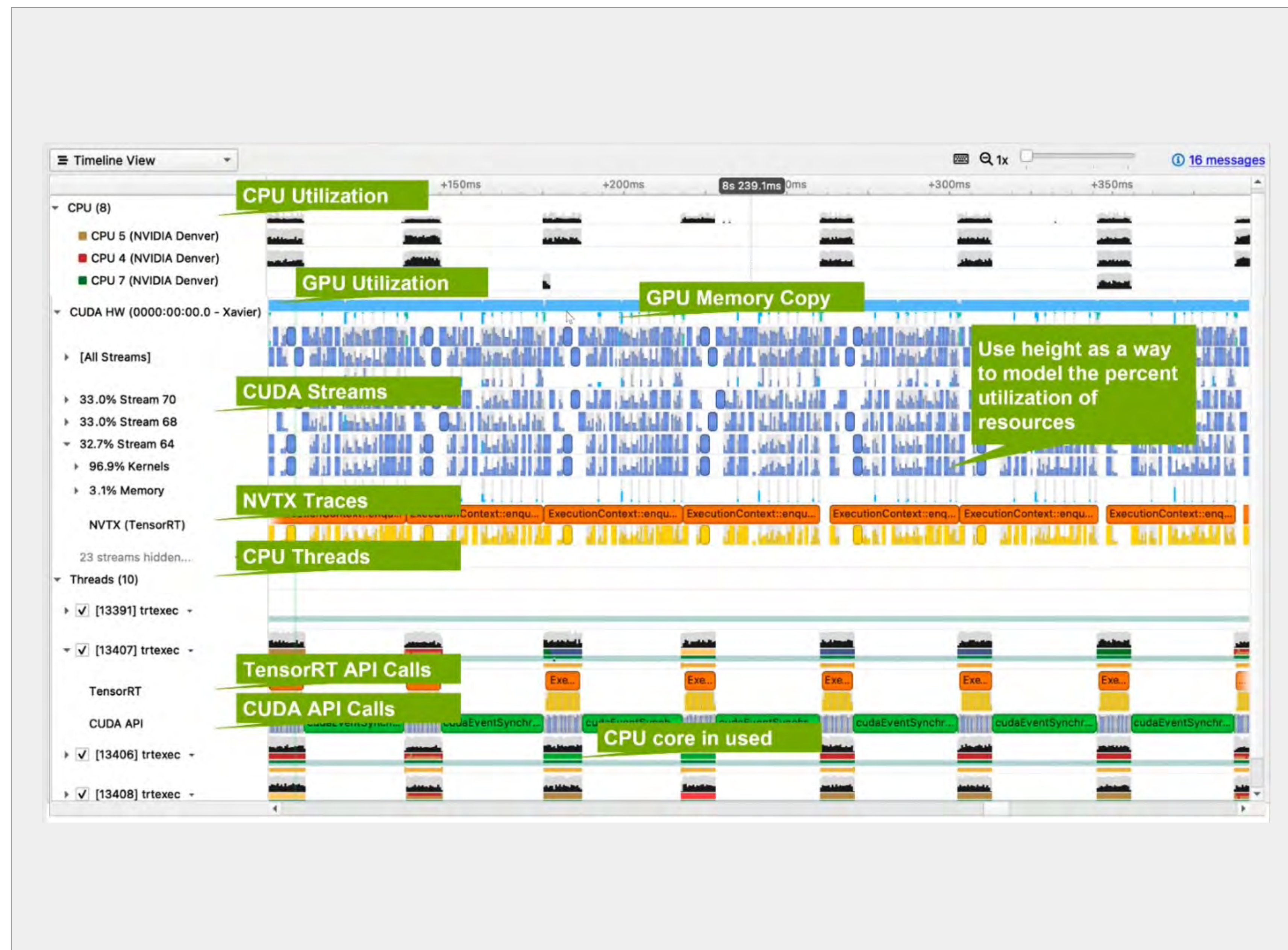


**Table of
Self-Calibration
Status**

Please Refer:

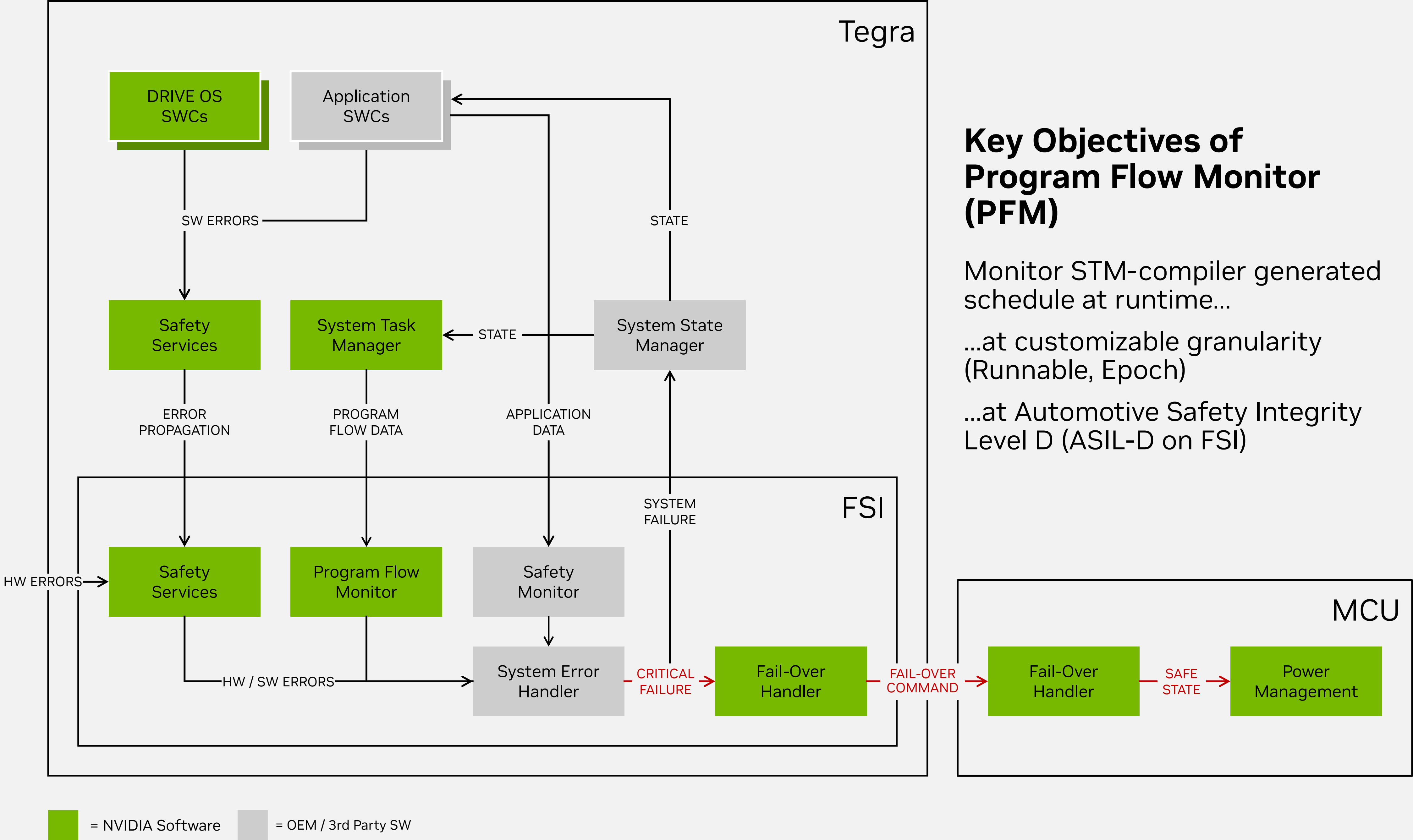
developer.nvidia.com/docs/drive/drive-os/6.0.6/public/driveworks-nvcgf/cgf_samples_demo.html

Profiling — Nsight Systems + NVTX



Please Refer : developer.nvidia.com/nsight-systems
docs.nvidia.com/gameworks/content/gameworkslibrary/nvtx/nsight-systems-nvtx-trace.htm

PFM in the Overall Safety Concept



Key Objectives of Program Flow Monitor (PFM)

Monitor STM-compiler generated schedule at runtime...

...at customizable granularity (Runnable, Epoch)

...at Automotive Safety Integrity Level D (ASIL-D on FSI)

Get Started with DRIVE SDK

Extensive documentation and training material available on NVIDIA Developer

Learn More

- Visit the [DRIVE training](#) page for webinars and other resources
- Check out information related to [DRIVE Hyperion](#), [DRIVE AGX Orin](#) and [DRIVE SDK](#)

Get Access

- Join the [DRIVE AGX SDK Program](#) on NVIDIA Developer
- [Read the docs](#) for DRIVE OS and DriveWorks documentation
- [Download DRIVE OS](#) which includes DriveWorks, NvMedia, CUDA, cuDNN and TensorRT

Contact Us

- Contact your distributor or NVIDIA's [Automotive Team](#)

