Directed Testing of ORAN using a Partially Specified Declarative Digital Twin

Alan Gatherer
Cirrus360
Dallas, USA
gatherer@cirrus3sixty.com

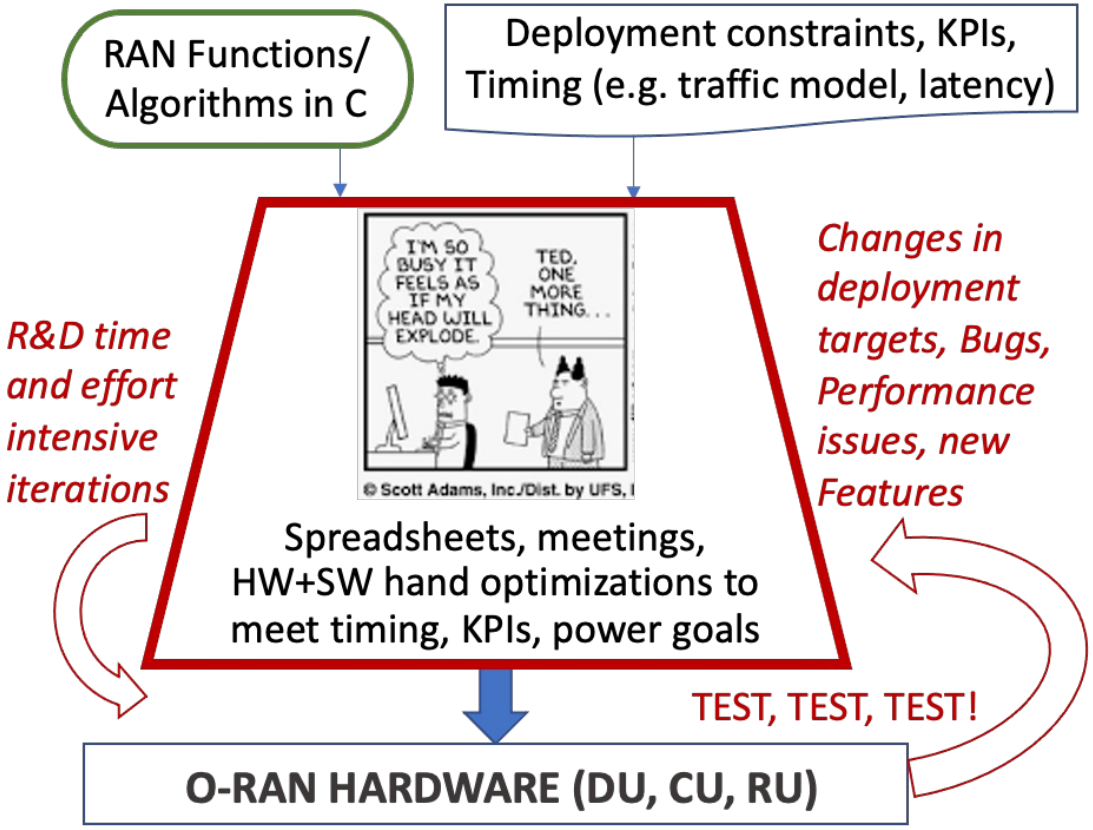
Chaitali Sengupta
Cirrus360
Dallas, USA
chaitali@cirrus3sixty.com

Sudipta Sen
Cirrus360
Dallas, USA
sudipta@cirrus3sixty.com

Jeffery H. Reed
Cirrus360 and Virginia Tech
Blacksburg, USA
reedjh@cirrus3sixty.com

Supported by NTIA
PWSCIF under award
48-60-IF006

Real time System Performance of RAN



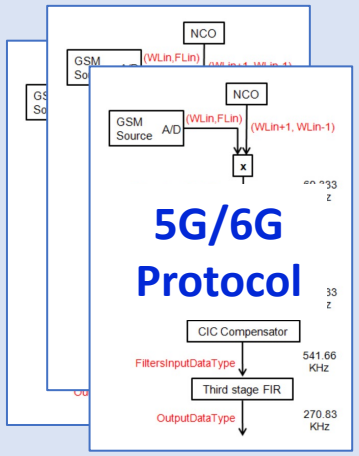
A competitive high-availability (HA) RAN is a finely tuned real time system, operating near its capacity

Any mistiming of the hardware or software can lead to unexpected, hard-to-track, functional errors, e.g., buffer overflows, timing errors

Impacts energy efficiency AND performance

➔ Requires automation, to direct testing towards specific HW/SW implementation & target deployment. But, not addressed by current open RAN testing strategies

Why is Real Time System Testing Hard?

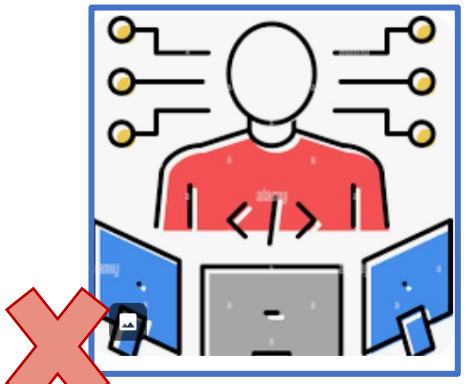


Deployment Requirements

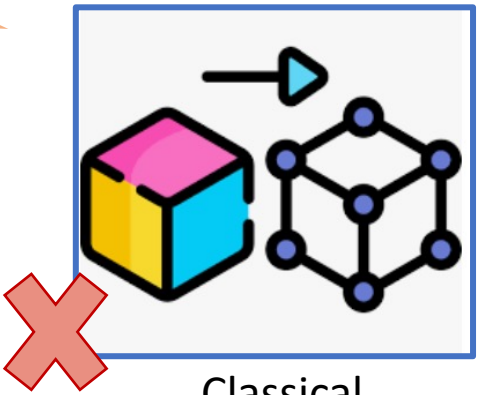
Complexities:

COTS HW/ cRAN/ vRAN,
multivendor,
partial knowledge,
deployment optimized solutions

Which tests are likely to cause the system to fail?



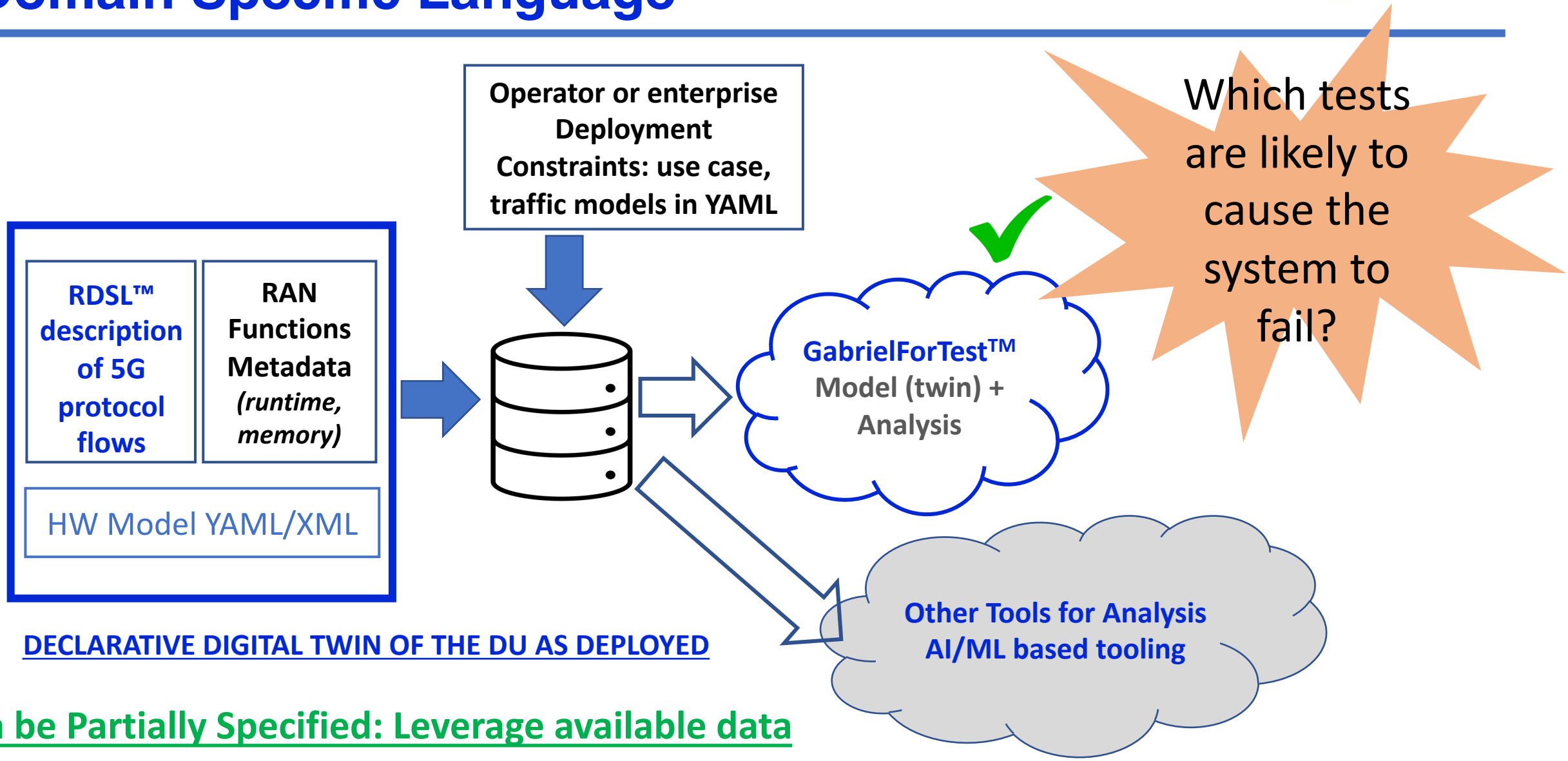
Traditional approach
Manual, experts driven



Classical
Digital Twin approach

A Declarative Digital Twin, powered by a RAN Domain Specific Language

Cirrus360



DECLARATIVE DIGITAL TWIN OF THE DU AS DEPLOYED

Can be Partially Specified: Leverage available data

Domain Specific Language (RDSL™): At the heart of the Declarative Digital Twin

Cirrus360

RDSL

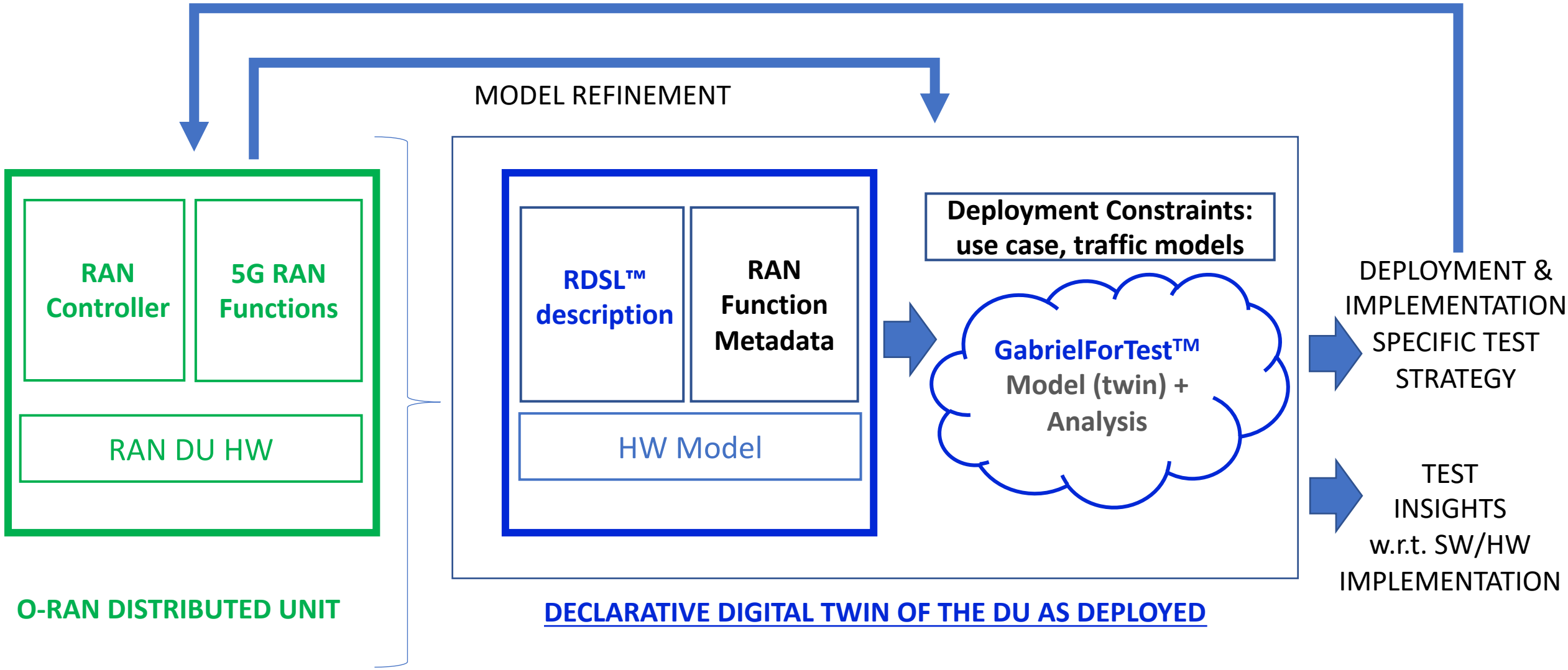
(.rdsl, Augmented with .yaml, .xml)

- Separate hardware and implementation independent description
- Declarative, Intent based → adding new requirements does not break the "twin"
- No vendor IP exposure
- Extensible, to meet future needs

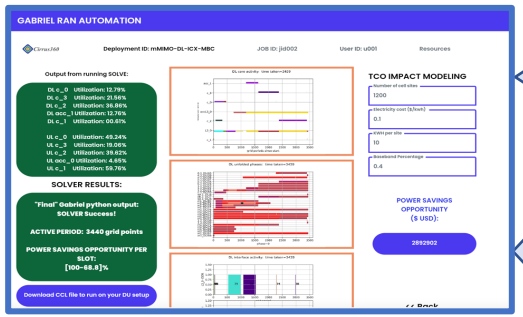
Will enable

- Simpler and formalized specification
- *Automation of testing / digital twins*
- *ALSO:*
  - *Automation of multi-vendor RAN integration*
  - Enable RIC apps by automating addition of KPIs
  - Integrates smoothly with higher-level automation and control

A Declarative Digital Twin Approach to Testing Multi-vendor Open RAN Solutions



How Cirrus360's Declarative Digital Twin works with industry standard testing solutions



HW under test evaluation and INSIGHTS

Model level evaluation and INSIGHTS

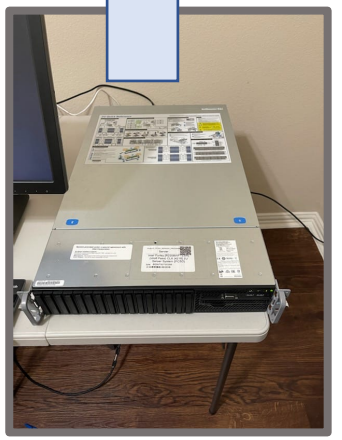
Deployment requirements,
Existing test cases

Model of HW under test

Digital Twin based analysis

Recommended Test Cases

Focus scope and add to test cases



Open RAN DU Under Test

Example input to the Declarative Digital Twin: How to test for Cache Sensitivity of DU running on COTS HW

```
flow srsChest_ueSpecific
    srsIQSymbols                :  stream[MAX_NUM_RX_ANT]{type = in}
    ueSpecific_srsInfo      :   stream[AVG_NUM_SRS_UE]{type = in}  % Maximum Number of UEs in a given SRS bandwidth
    error_streams_type1     :   stream[AVG_NUM_SRS_UE][MAX_NUM_RX_ANT][3]
    error_streams_type2     :   stream[AVG_NUM_SRS_UE][2]
    %Internal Stream Definitions
    perUE_srsChEst          :   stream[AVG_NUM_SRS_UE][MAX_NUM_RX_ANT]% These streams goes to non-RDSL domain
    %Flows Multi-instantiation of UE specific SRS Chest Flows for each Symbols
    srsChestProc_perUE_perRxAnt_flow{i = 1:AVG_NUM_SRS_UE, j = 1:MAX_NUM_RX_ANT,
                                     srsIQSymbols_perRxAnt_in = srsIQSymbols[j],
                                     perUE_srsInfo_in = ueSpecific_srsInfo[i],
                                     perUE_srsChEst_out = perUE_srsChEst[i][j],
                                     error_s = error_streams_type1[i][j]}
    % Flow with Stub Modifier that do data pre-processing/compression on SRS Chest
    % before sending UE specific SRS Chest Data to MAC
    sendSrsChEst_to_MAC_flow{i = 1:AVG_NUM_SRS_UE,
                             perUE_srsChest_in = perUE_srsChEst[i],
                             perUE_srsInfo_in = ueSpecific_srsInfo[i],
                             error_s = error_streams_type2[i]}
```

1. RDSL description of DU Test Case: 10MHz Downlink-only Based on Intel® FlexRAN™ Reference Architecture

```
apiVersion: rdsl/v0
kind: timing equation
metadata:
  name: <name>
spec:
  equation: 0 <= A*370 + B < 500
    C: grid_period
    A: num_ue1
    B: gp_base
    unit: clock
```

3. System Constraints: e.g. Timing

```
<pattern name="big_delay.c_0.L3_0.DDR_0.L3_0">
  <defining_memory>L3_0</defining_memory>
  <observing_memory>L3_0</observing_memory>
  <exclusive_define_with>
    <member>big_delay.c_0.L3_0.DDR_0.L3_0</member>
    <member>big_delay.c_0.L3_0.DDR_0.accL3_0</member>
    <member>pipeline.c_0.L3_0</member>
    <member>L2toL2.c_0.L3_0.accL3_0</member>
  </exclusive_define_with>
  <shares_L2_II_with>
  ...
  </shares_L2_II_with>
  <shares_L2_IO_with>
  ...
  </shares_L2_IO_with>
  <shares_L2_OI_with>
  ...
  </shares_L2_OI_with>
  <shares_L2_OO_with>
    <member>big_delay.c_0.L3_0.DDR_0.L3_0</member>
    <member>big_delay.c_0.L3_0.DDR_0.accL3_0</member>
    <member>L2toL2.c_0.L3_0.accL3_0</member>
    <member>big_delay.c_1.L3_0.DDR_0.L3_0</member>
    <member>big_delay.c_1.L3_0.DDR_0.accL3_0</member>
    <member>L2toL2.c_1.L3_0.accL3_0</member>
    <member>big_delay.c_2.L3_0.DDR_0.L3_0</member>
    <member>big_delay.c_2.L3_0.DDR_0.accL3_0</member>
    <member>L2toL2.c_2.L3_0.accL3_0</member>
    <member>big_delay.c_3.L3_0.DDR_0.L3_0</member>
    <member>big_delay.c_3.L3_0.DDR_0.accL3_0</member>
    <member>L2toL2.c_3.L3_0.accL3_0</member>
  </shares_L2_OO_with>
  <shares_L3_II_with/>
  <shares_L3_IO_with/>
  <shares_L3_OI_with/>
  <shares_L3_OO_with/>
  <can_observe>
  ...
  </can_observe>
</pattern>
```

2. Hardware Resources description: 2GHz Intel Icelake Xeon Gold 6338

```
apiVersion: rdsl/v0
kind: SDK
metadata:
  name: NR5G1_DL_PDSCH_SYM
spec:
  available patterns:
  - big_delay.c_0.L3_0.DDR_0.L3_0
  - big_delay.c_0.L3_0.DDR_0.accL3_0
  - pipeline.c_0.L3_0
  - L2toL2.c_0.L3_0.accL3_0
  - big_delay.c_1.L3_0.DDR_0.L3_0
  - big_delay.c_1.L3_0.DDR_0.accL3_0
  - pipeline.c_1.L3_0
  - L2toL2.c_1.L3_0.accL3_0
  - big_delay.c_2.L3_0.DDR_0.L3_0
  - big_delay.c_2.L3_0.DDR_0.accL3_0
  - pipeline.c_2.L3_0
  - L2toL2.c_2.L3_0.accL3_0
  - big_delay.c_3.L3_0.DDR_0.L3_0
  - big_delay.c_3.L3_0.DDR_0.accL3_0
  - pipeline.c_3.L3_0
  - L2toL2.c_3.L3_0.accL3_0
  elementsize: 2000000
  internalsize: 8000000
  runtime: 7200
```

4. Functional block Constraints: Cache sensitivity i.e. what gets evicted

Cache Sensitivity of DU running on COTS HW: Testing recommendations from Digital Twin

Cirrus360

	Strategy	Latency (clock cycles)	Result from GabrielForTest™
1	No added constraints related to cache	207,800	Test can successfully complete on a hardware slice of 4 cores well within available slot time
2	Cache eviction occurs for a set of small buffers (all < 10KB)	239,400 (15% increase)	Small impact, robust to this cache effect
3a	Large shared buffer (DL Config) eviction	420,000 (102% increase)	Big impact! Requires this test scenario to be reproduced on actual DU. May be a candidate for code change to avoid this situation in the field
3b	Large data buffer cache eviction	464,600 (124% increase)	Some additional impact, Digital Twin can explain why!

Table 1: Results for different pattern management (cache sensitivity) constraints: Impact on Latency

Conclusions

- System Performance testing is hard! Takes expertise, R&D time, and effort but essential for multivendor ORAN to be competitive
- RDSL™ driven DECLARATIVE DIGITAL TWIN can accelerate system testing:
  - Large repository of FACTS or CONSTRAINTS about the deployed ORAN component(s)
  - That AUTOMATION can analyze for HW-SW implementation related corner cases
  - Enables directed system testing to accelerate testing time and investment for ORAN

Contact: Chaitali Sengupta, chaitali@cirrus3sixty.com , +1 214 733 2803

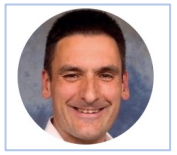
ADDITIONAL SLIDES

OCTOBER 2024

Team

CHAITALI SENGUPTA (PhD), CEO

- Sr. Dir. Qualcomm, VP Eng. Reliance Jio India, Founder SNRLabs (startup), CTO SEVEN
- Distinguished Member Tech Staff Texas Instruments, Board Member Reliance Jio JV
- MIT Tech Review top innovators (TR35), 13 patent families



ALAN GATHERER (PhD), CTO

- Senior VP, 5G Baseband SoCs, Futurewei/Huawei (10 years rise to market leader)
- IEEE Fellow (1% of members), 90+ patents
- Texas Instruments Fellow, drove TI's domination for >10 years of 3G/4G

SUDIPTA SEN, VP Product

- VP Product Reliance Jio. Sr. Dir. Qualcomm. TI. Shipped 100s million wireless products
- Founder SNRLabs, acquired by SEVEN. WiFi-cellular mobility product & core IP

PROF. JEFF REED, Advisor

- Virginia Tech, IEEE Fellow (Software Defined Radio)
- Co-founder of Federated Wireless, Advisor US spectrum policies