Objective

How would I go about applying *coreconf* in my application?
Key Terms

Representation Format: CORECONF

Serialization Format: CBOR/JSON
Background

- Architecture
- JSON data instance on the wire
Design your YANG model

Generate SIDs (and register them)

Generate & Implement C Stubs

Generate CBOR dump of your Model

Program into RIOT chatgpt*
Design your YANG model

Generate SIDs (and register them)

Generate & Implement C Stubs

Generate CBOR dump of your Model

Program into RIOT chatgpt*
Design your YANG model

- Pick up from previous example

```yang
container sensorObject {
  leaf statusLED {
    type enumeration {
      enum green {value 0;}
      enum yellow {value 1;}
      enum red {value 2;}
    }
  }
  leaf battery {
    type battery-level;
  }
  list sensorReadings {
    key "index";
    leaf index {
      type uint8;
    }
    leaf sensorValue {
      type uint32;
    }
  }
}
```
Design your YANG model

Generate & Implement C Stubs

Program into RIOT chatgpt*

30 mins ➔ 10 mins ➔ 30 mins ➔ 10 mins ➔ 0* mins

Generate SIDs (and register them)

Generate CBOR dump of your Model
Generate SIDs

- New Identifiers?
- If yes, generate & register them!
- Use pyang tool
Design your YANG model  

Generate SIDs (and register them)  

Generate & Implement C Stubs  

Generate CBOR dump of your Model  

Program into RIOT chatgpt*
Implement C Stubs

- Use tool to make your work easier
- Implement leaf-fetching/update functions
- Stubs are aware of the tree structure
```c
#include <stdbool.h>
#include <stdint.h>
#include <stdlib.h>
#include <string.h>

#include "nanocbor/nanocbor.h"

#define SID_BATTERY 60006
#define SID_INDEX 60008
#define SID_SENSORVALUE 60009
#define SID_STATUSLED 60010

#define read_sensorObject read_60005
#define read_battery read_60006
#define read_sensorReadings read_60007
#define read_sensorValue read_60009
#define read_statusLED read_60010

char* keyMapping = "\xa1\x19\xeag\x81\x19\xeah";

void read_sensorObject(void);
void read_battery(void);
void read_sensorReadings(uint8_t index);
uint32_t read_sensorValue(uint8_t index);
char* read_statusLED(void);
```

Header: sensor_implementation.h
```c
void read_sensorReadings(uint8_t index){
    // Initialize the leaf if it has a return type with a default value;

    // Do something with the leaf
}

/*
 * This is an autogenerated function associated to
 * SID: 60009
 * Module: data
 * Identifier: /sensor:sensorObject/sensorReadings/sensorValue
 * function params: /sensor:sensorObject/sensorReadings/index
 * Stable: false
*/
uint32_t read_sensorValue(uint8_t index){
    // Initialize the leaf if it has a return type with a default value;
    uint32_t sensorValueInstance = 0;

    // Do something with the leaf
    // Return the leaf
    return sensorValueInstance;
}
```

C Impl.: sensor_implementation.c
Design your YANG model
30 mins

Generate & Implement C Stubs
10 mins

Generate SIDs (and register them)
30 mins

Generate CBOR dump of your Model
10 mins

Program into RIOT chatgpt*
0* mins
Generate CBOR Dumps

- Dump your models and relevant SID information into header files
- Need Pycoreconf library!
Transformation

144 bytes

```
{
    "sensor:sensorObject": {
        "statusLED": "green",
        "battery": "sensor:med-level",
        "sensorReadings": [
            {
                "index": 0,
                "sensorValue": 42
            },
            {
                "index": 1,
                "sensorValue": 22
            }
        ]
    }
}
```

24 bytes

```
b'a119ea65a305000119ea40282aa2010002182aa201010216'
```
<table>
<thead>
<tr>
<th>Task</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design your YANG model</td>
<td>30 mins</td>
</tr>
<tr>
<td>Generate &amp; Implement C Stubs</td>
<td>10 mins</td>
</tr>
<tr>
<td>Generate CBOR dump of your Model</td>
<td>10 mins</td>
</tr>
<tr>
<td>Program into RIOT chatgpt* (Program)</td>
<td>0* mins</td>
</tr>
</tbody>
</table>

*Note: The time for Program into RIOT chatgpt* can vary depending on the specific implementation and setup.
RIOT Int.

HOST

CoAP Request

CoAP Response

Embedded Device (emulated)

Sensor (emulated)

Data Model Store (CORECONF)
Get subtree where SID key is 60007 and the value of the sid-key is 1
Demo Failed?

COAP Client sending payload

RIOT Emulated server

Response CBOR code: 2.05 Content
For more mortals who can't read CBOR, it is:

```
[600a5: {1: 600a4, 2: [{1: 0, 2: 42}, {1: 1, 2: 22}], 5: 'green'}],
{600a7: {1: 0, 2: 42}}
```

alonso@alonso-Nitro-AN515-57:~/projects/lpwan_examples/RIOT/examples/coreconf_simple$ python3 requestCBO.py

Sending a COAP Request with payload:

```
[600a5, 600a7]
```

```
... ...
```

```
2024-06-10 01:46:13,766 # (Child SID =60000) DLL: 60007,
2024-06-10 01:46:13,766 # (Child SID =60007) DLL: 60005,
2024-06-10 01:46:13,766 # (Child SID =60005) DLL: 6,  
2024-06-10 01:46:13,766 # (Child SID =60000) DLL: 60007,
2024-06-10 01:46:13,767 # (Child SID =60007) DLL: 60005,
2024-06-10 01:46:13,767 # (Child SID =60005) DLL: 6,  
2024-06-10 01:46:13,767 # (Child SID =60000) DLL: 60007,
2024-06-10 01:46:13,767 # Initialized server
2024-06-10 01:46:13,767 # gcoap example app
2024-06-10 01:46:13,767 # All up, running the shell now
> 2024-06-10 01:46:19,198 # Request Payload in CBOR Hex: 8219ea6f58219ea6780
2024-06-10 01:46:19,198 # Deserialized Coreconf:
2024-06-10 01:46:19,198 # [60005, [60007, 0]]
2024-06-10 01:46:19,198 # To reach your SID, the following SIDs are traversed:
2024-06-10 01:46:19,198 # parentsSID = 60005 [
2024-06-10 01:46:19,198 # Coreconf subtree after traversal:
2024-06-10 01:46:19,199 # Key: 6007 Value: 60005,  
2024-06-10 01:46:19,200 # Key: 1 Value: 60005,  
2024-06-10 01:46:19,200 # ParentSID = 60007 DLL: 60008,
2024-06-10 01:46:19,200 # Start serialization
2024-06-10 01:46:19,200 # Encoder initialized
```

```
From University Of Grenoble-Alps

Lead by Prof. Didier Donsez, LIG ERODS Team
Notes

ccoreconf github library
riot-integrated-example
masters-thesis

Wanna Contribute?

● Engineering work:
  ○ Profiling, memory checks for ccoreconf (like valgrind)
  ○ Integration with other zephyr

● Security considerations for ccoreconf are absent

● Scientific work:
  ○ Dynamic yang models
  ○ Correctness of stubs
  ○ LWM2M / SenML int.
Fin.

Thank you!