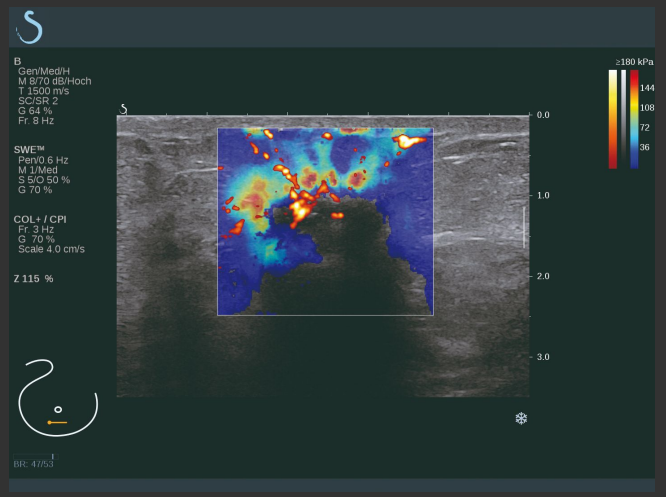
A SYSTEM THAT I USED TO KNOW

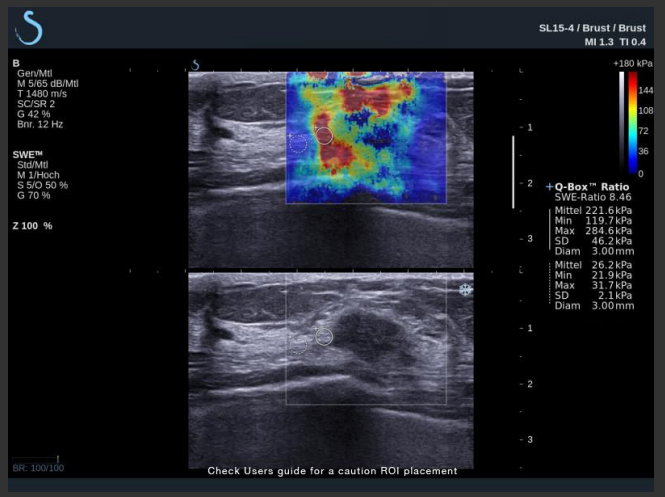
From "Hello World" to ShearWave Elastography

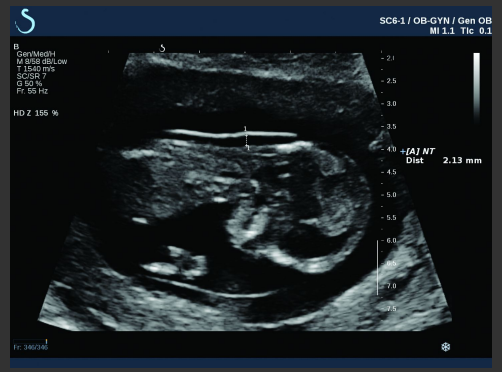
Benoit Chauvin

Benoit.chauvin[at] gmail.com

AN ULTRASOUND SYSTEM



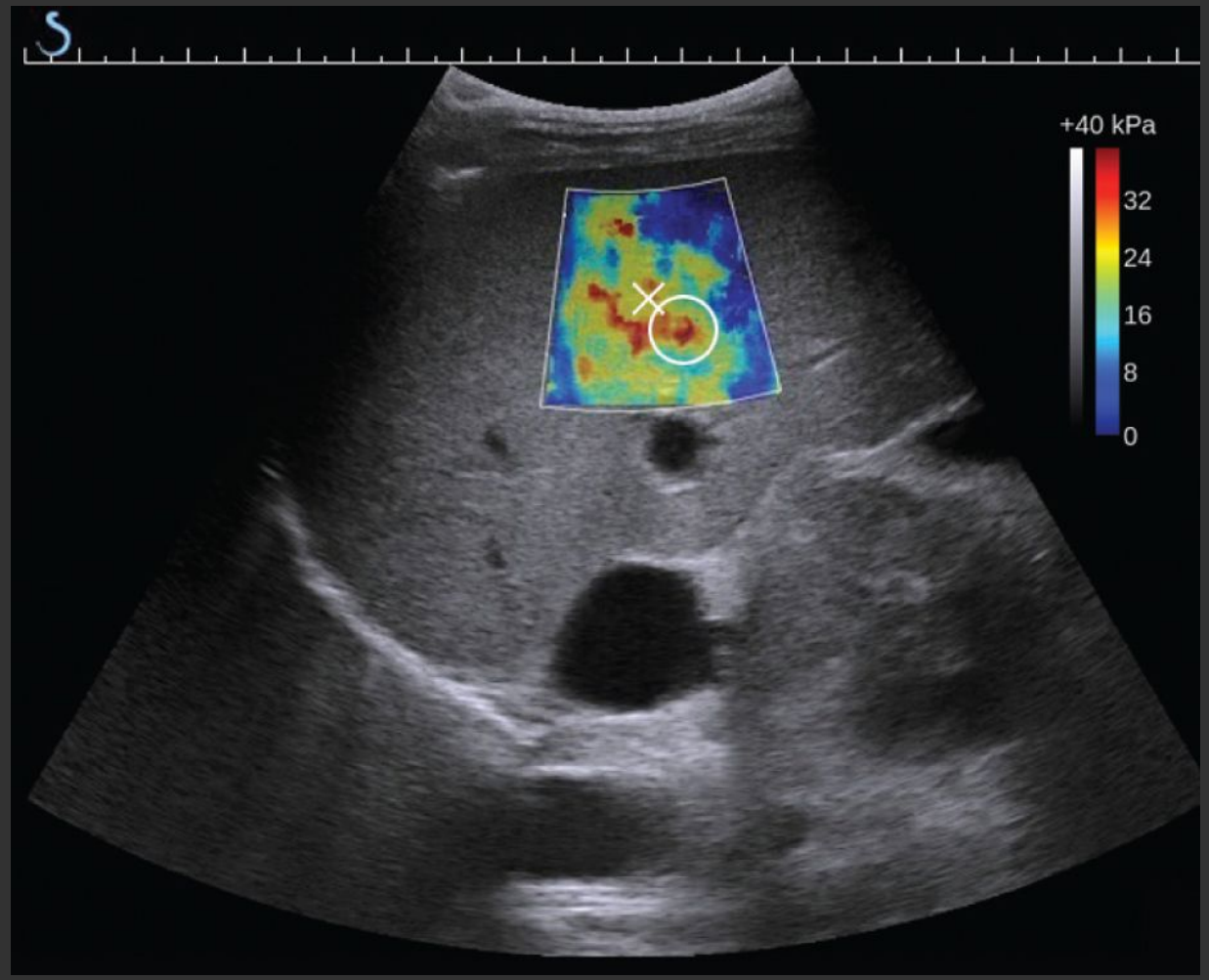




CONTEXT

- Start-up in the south of France
- Founded by well known names in the field
- Ambitious project:
  - GNU/Linux based software centric ultrasound system
  - New tech for ultrasound
  - New hardware

SHEARWAVE ELASTOGRAPHY 1/2



SHEARWAVE ELASTOGRAPHY 2/2

PUSH

- Depth oriented
- Wave creation

IMAGING

- Wave imaging
- High frequency

DISPLAY

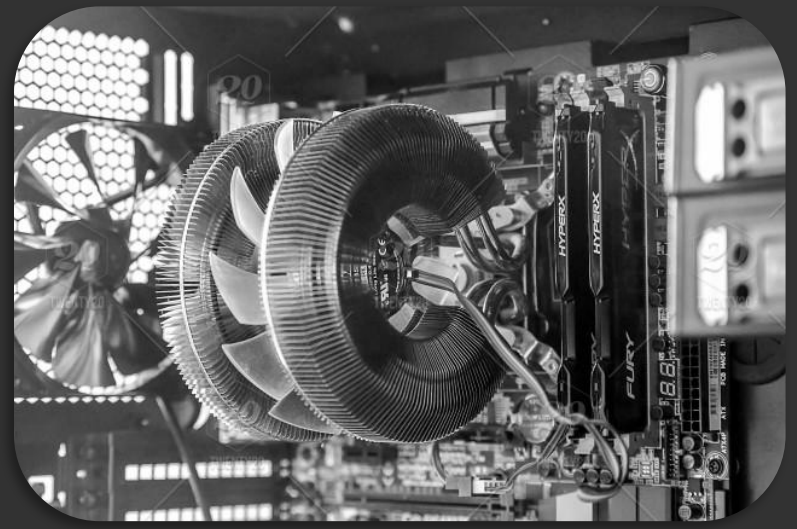
- LUT for density
- Quantification

CONCEPTUAL CONSTRAINTS

- KISS principle
  - Maintainability
  - Readability
  - Ease of change
- No need to reinvent the wheel
  - Use third party libraries
  - Rely on the standard library
  - Avoid unnecessary code
- Trust but verify
  - Changes to be heavily peer reviewed

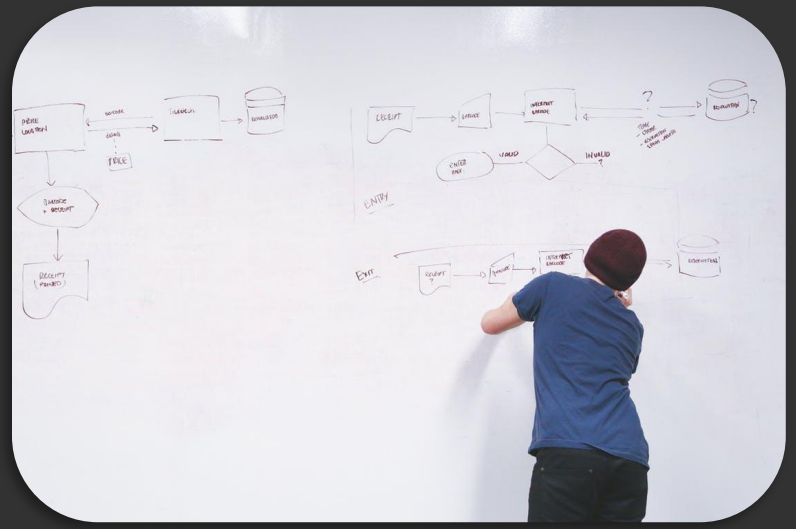
TECHNICAL CONSTRAINTS

- Language C/C++
  - Good performance
  - Available libraries
- GNU/Linux
  - Efficiency
  - Cost of changing OS low-ish
  - Driver development

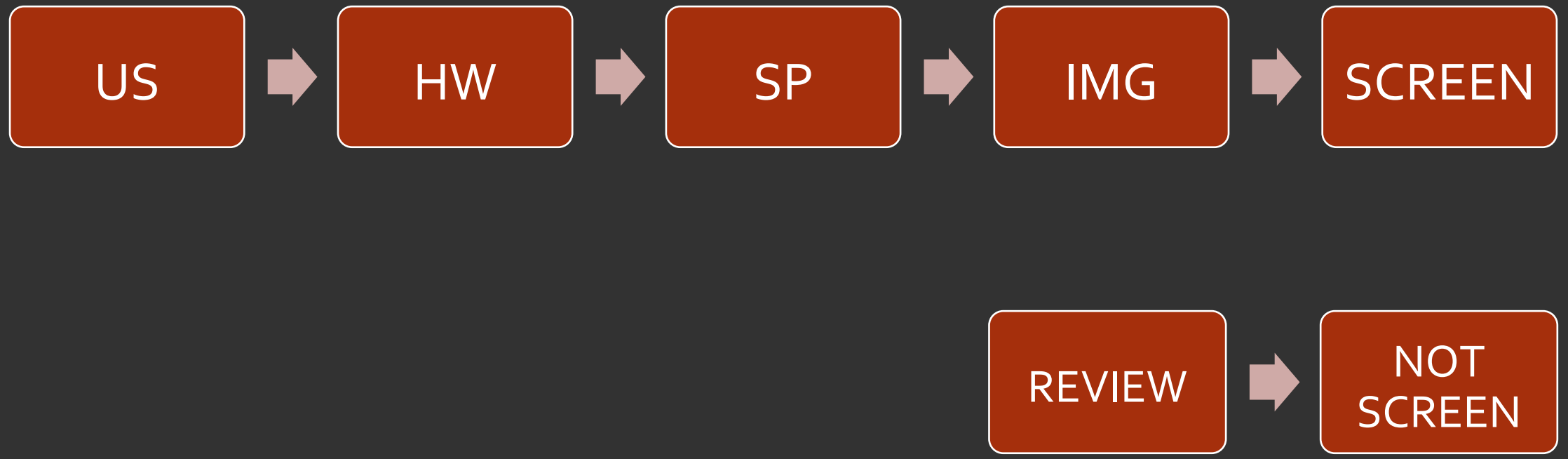


DESIGN CONSTRAINTS

- Modular
  - Each module is a process
  - Manager for scheduling / debugging
- Event based system
  - State machines
  - “Easy” to add a new transversal path
- Last moment processing
  - Ultrasound images are noisy
  - User parameters changes at review for tuning

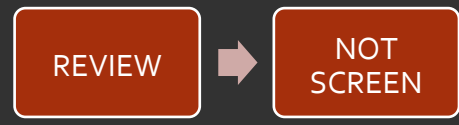


A “STATE OF THE ART” PIPELINE



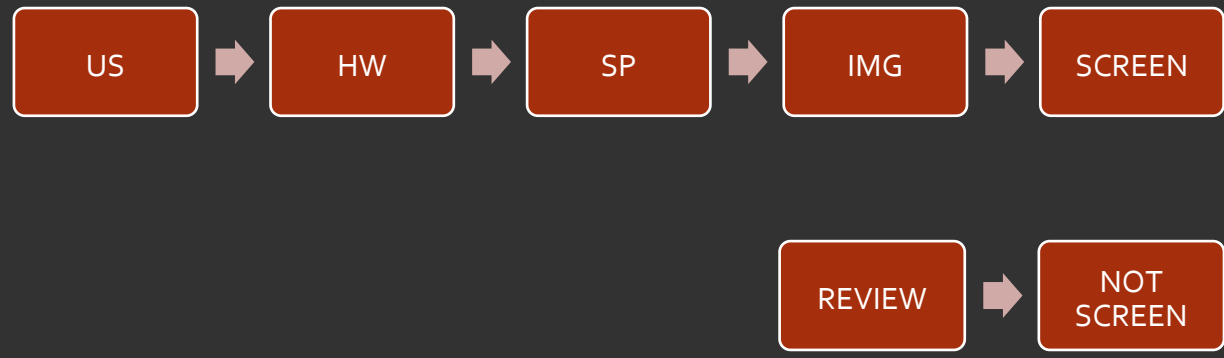
WHAT DOES WHAT? 1/2

- US
  - Controls ultrasound parameters from user requests
  - Programming sequences for the hardware
- HW
  - Executing the sequences from US
  - Giving the data back to the signal processing unit
- SP
  - Process the raw data to grayscale images
  - Minimal image processing

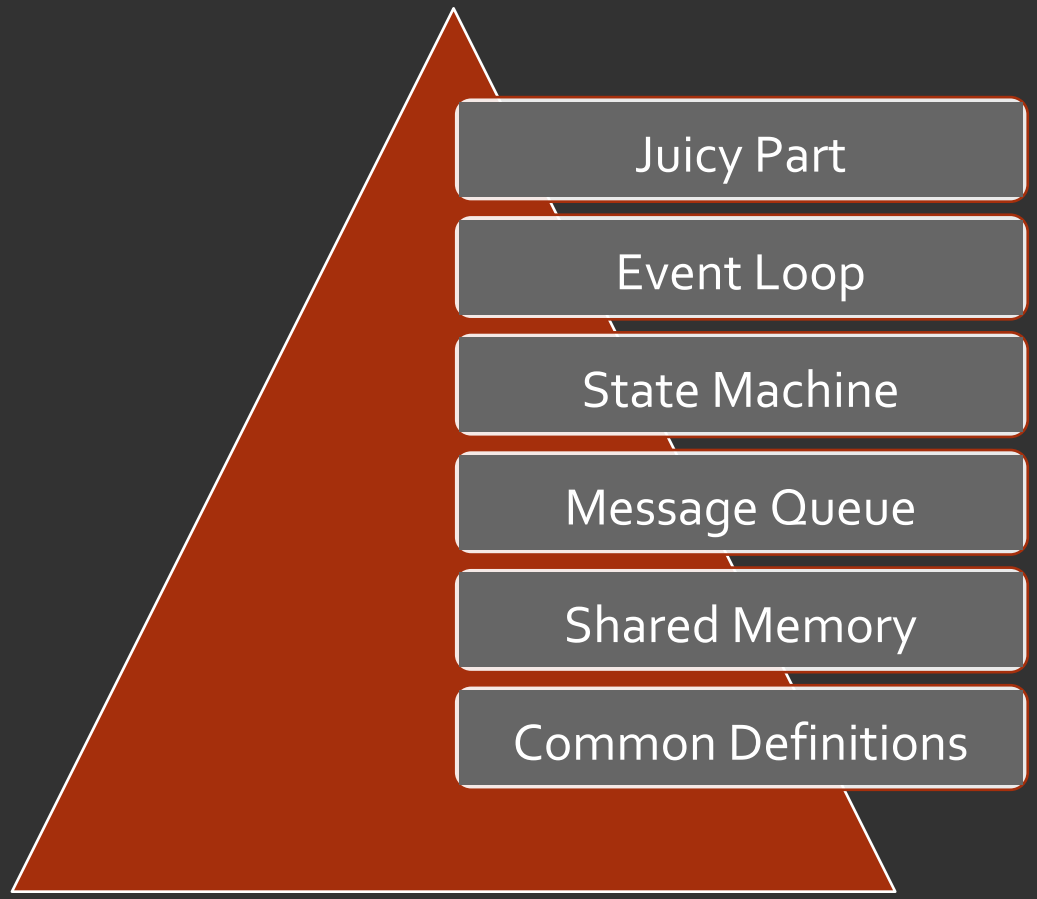


WHAT DOES WHAT? 2/2

- IMG
  - User control display parameters
  - Controls the screen
  - Request US changes
- SCREEN
  - Image processing
  - Display the image to the user
- REVIEW / NOT-SCREEN
  - User control display parameters when not imaging
  - Controls off screen rendering

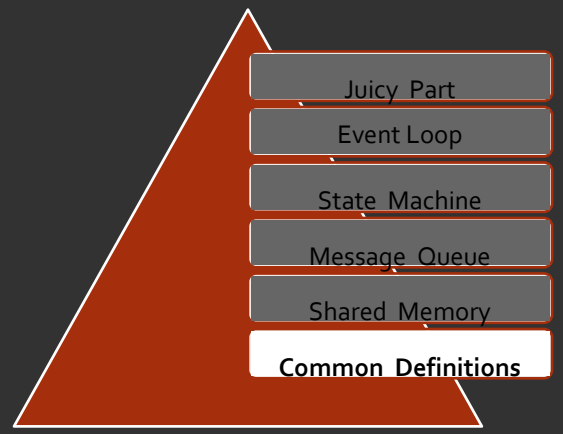


MODULE ARCHITECTURE



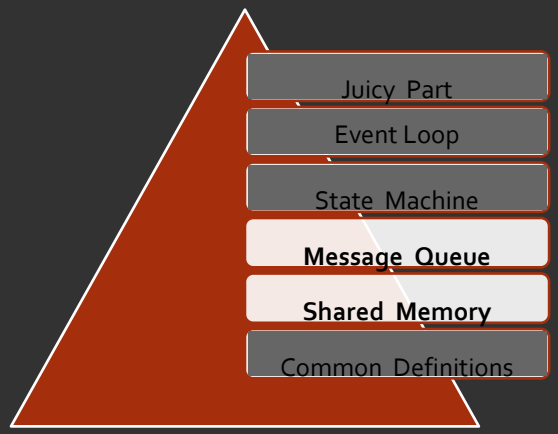
COMMON DEFINITIONS

- Identifiers
  - Message queues ID
  - Shared memories ID
  - Shared memory chunks ID
- Structures
  - Shared memories structures
  - Messages structures
- Classes
  - Mutexes
  - Shared memory class
  - Module class
  - State machine system



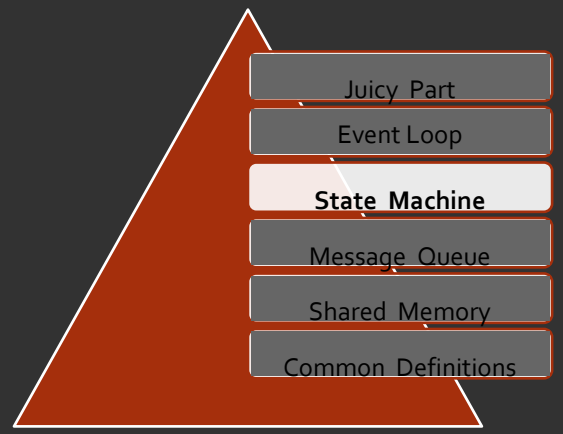
SHARED MEMORY/MESSAGE QUEUES

- Shared Memory
  - One for the raw dump from the hardware
  - One for the processed data
  - Shared amongst all modules
- Message Queues
  - One per module
  - All modules know all the messages
  - Small messages, one command, one SHM chunk ID
  - All the functionalities encoded in a xml state



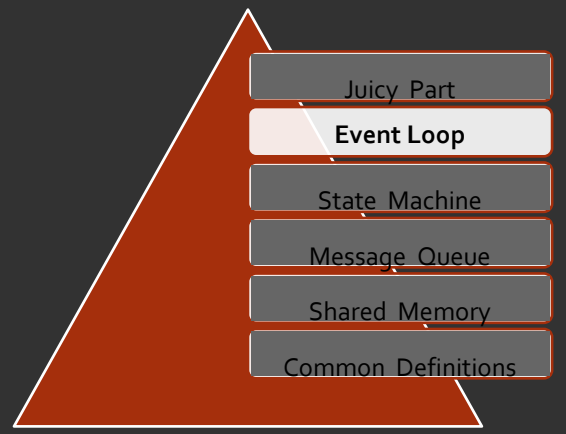
POWER TO THE STATES

- State machine centralised the behaviours
- Really simple to add/change/remove path
- Pitfalls:
  - Really difficult to debug live
  - No documentation up to date
- Need for live debugging tools
  - State machine live display
  - Sequence diagram recorder



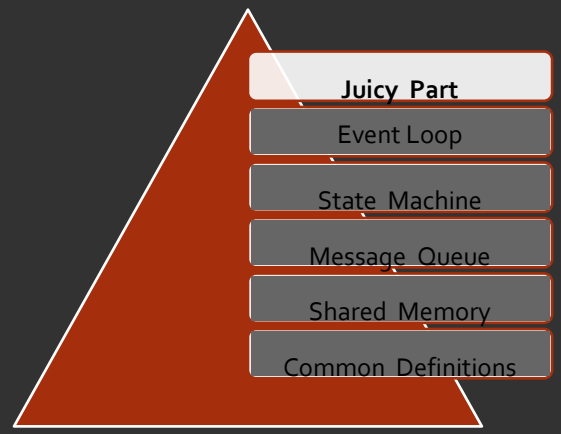
EVENT LOOP

- Event loops reacts to messages and drives the system
- Only the concept of current data
  - Current image
  - Current mode
  - Really Buddhist “now”
- Processing unit retains minimum information



MY JUICY PARTS SCREEN-IMG-REVIEW

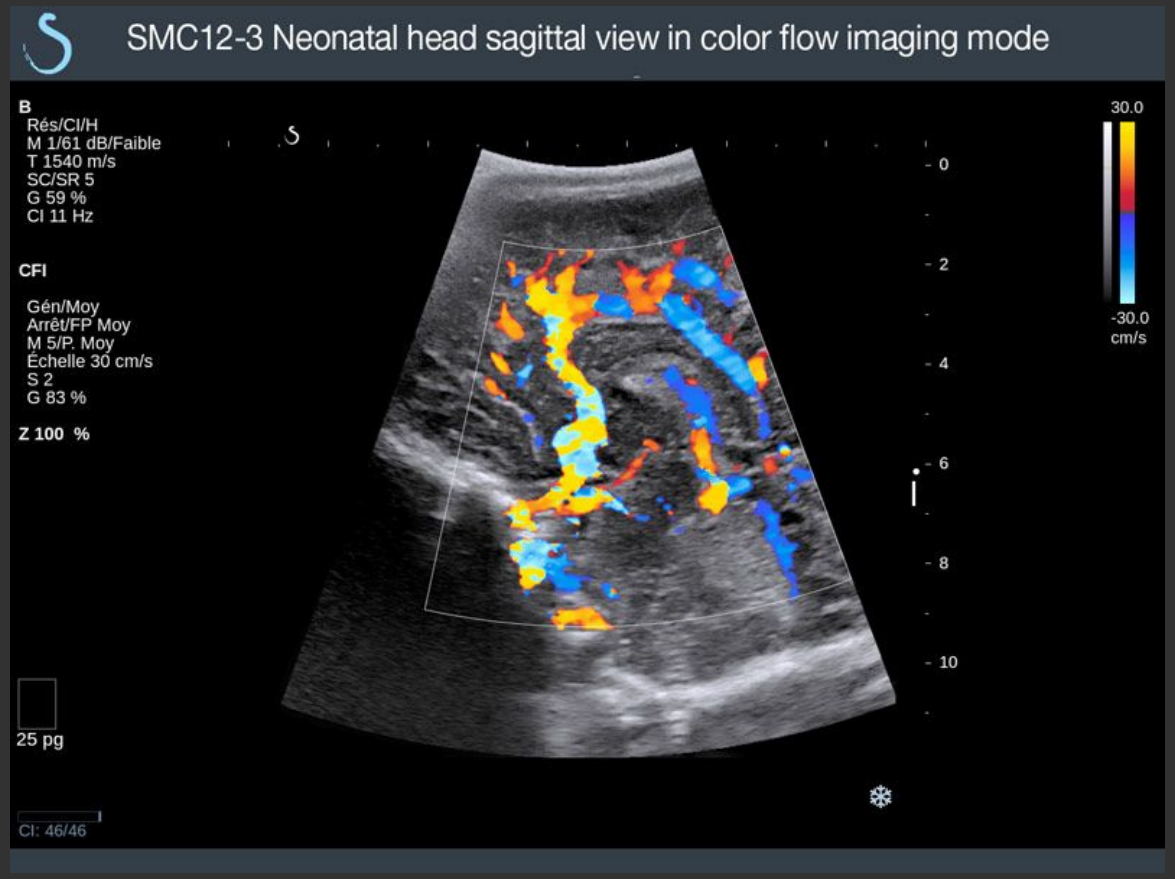
- Constraints
  - 1% of one CPU
  - Full use of the GPU
  - Highest frame rate possible
  - Readability of the UI as the goal
- Personal challenges
  - Just graduated
  - Never done a UI before
  - Never done any C++
  - Never touched openGL
  - Other team members had 15+ years experience

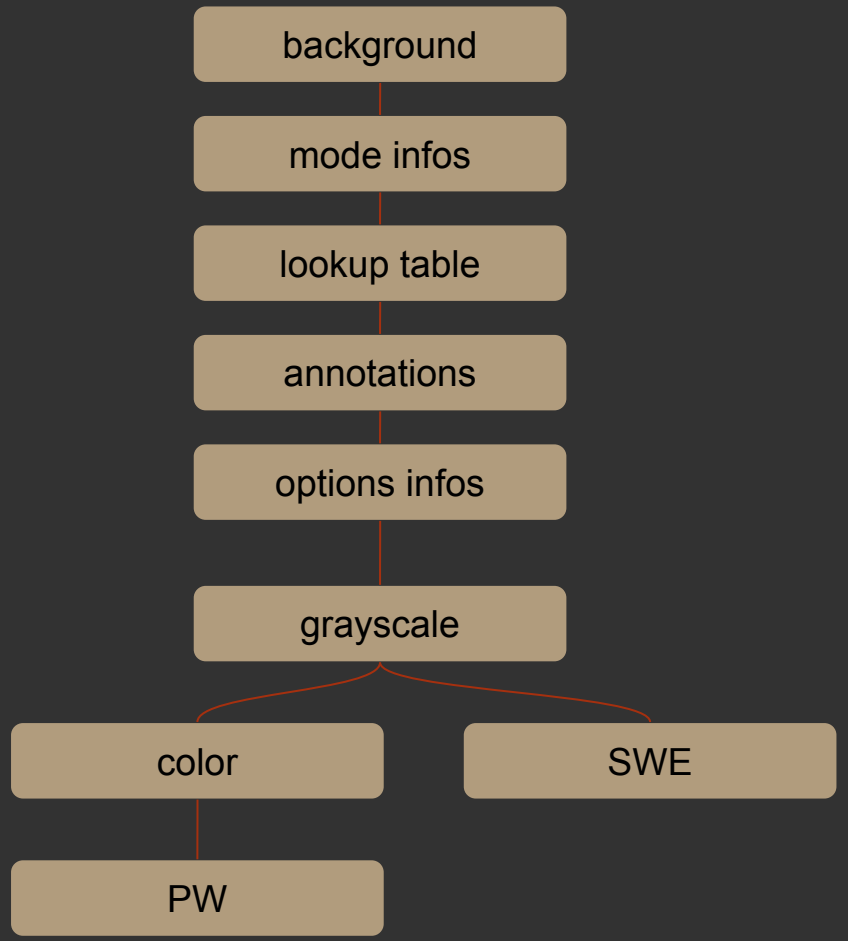


SCREEN 1/2

- One window
  - SDL
  - GTKmm / Cairo
- Graphics libraries
  - openGL
  - CUDA
- Modular system based on inheritance
- One big HUB for updates

SCREEN 2/2


SMC12-3 Neonatal head sagittal view in color flow imaging mode
B
Rés/CI/H
M 1/61 dB/Faible
T 1540 m/s
SC/SR 5
G 59 %
CI 11 Hz
CFI
Gén/Moy
Arrêt/FP Moy
M 5/P. Moy
Échelle 30 cm/s
S 2
G 83 %
Z 100 %
30.0
-30.0
cm/s
0
2
4
6
8
10
25 pg
CI: 46/46


background
mode infos
lookup table
annotations
options infos
grayscale
color
SWE
PW

CONCLUSION

- We succeeded in launching Aixplorer on the market
- Company growth from 4 to 80 people in the meantime
- Each team had it's module to take care of
  - US engineers able to experiment easily
- We were able to add modules easily
  - Measurement
  - Reporting
  - DICOM
- I learnt a lot in a lot of different domains
- This shaped my way of architecturing software

THE QUESTIONS SLIDE

- Now
- Later on at the pub
- Anytime: benoit.chauvin[at].com