

cs121 - software development
software architecture
sai patterns

alexandre r.j. françois

visiting associate professor of computer science



outline

sai primitives and organizing principles

rendering

dynamic system

network receive and send

user interaction controls

music mixer and player

sai primitives & organizing principles

cell

processing unit (no state)

repository

shared persistent data

stream

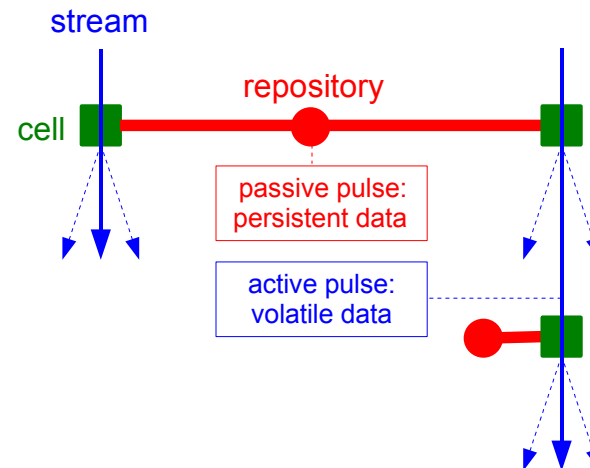
flow of volatile data

process dependency

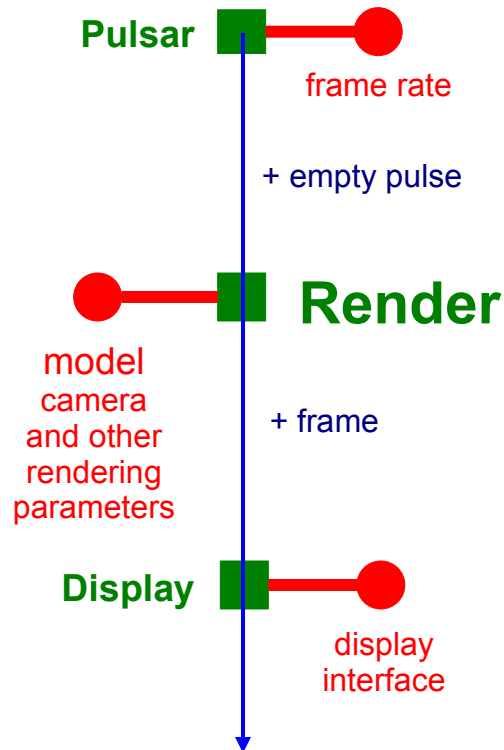
process trigger

pulse

active (volatile) vs. passive (persistent)



graphics rendering

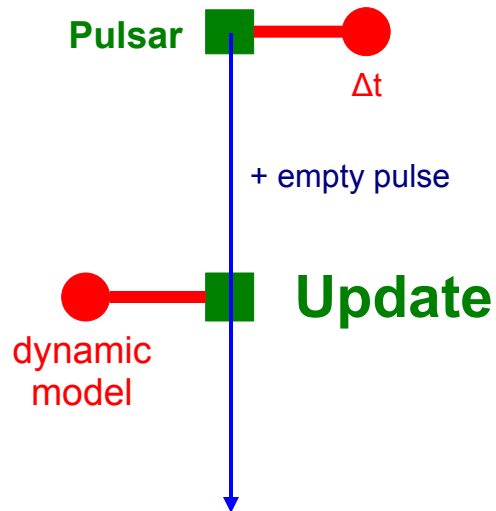


pulsar generates empty pulses that trigger the Render cell at regular time intervals

render cell creates a graphical depiction of the model held in the repository, and places the frame on the stream

display cell places the input frame on the screen

dynamic system



pulsar generates empty pulses that trigger the update cell at regular time intervals

update cell computes the state of the system after time increment

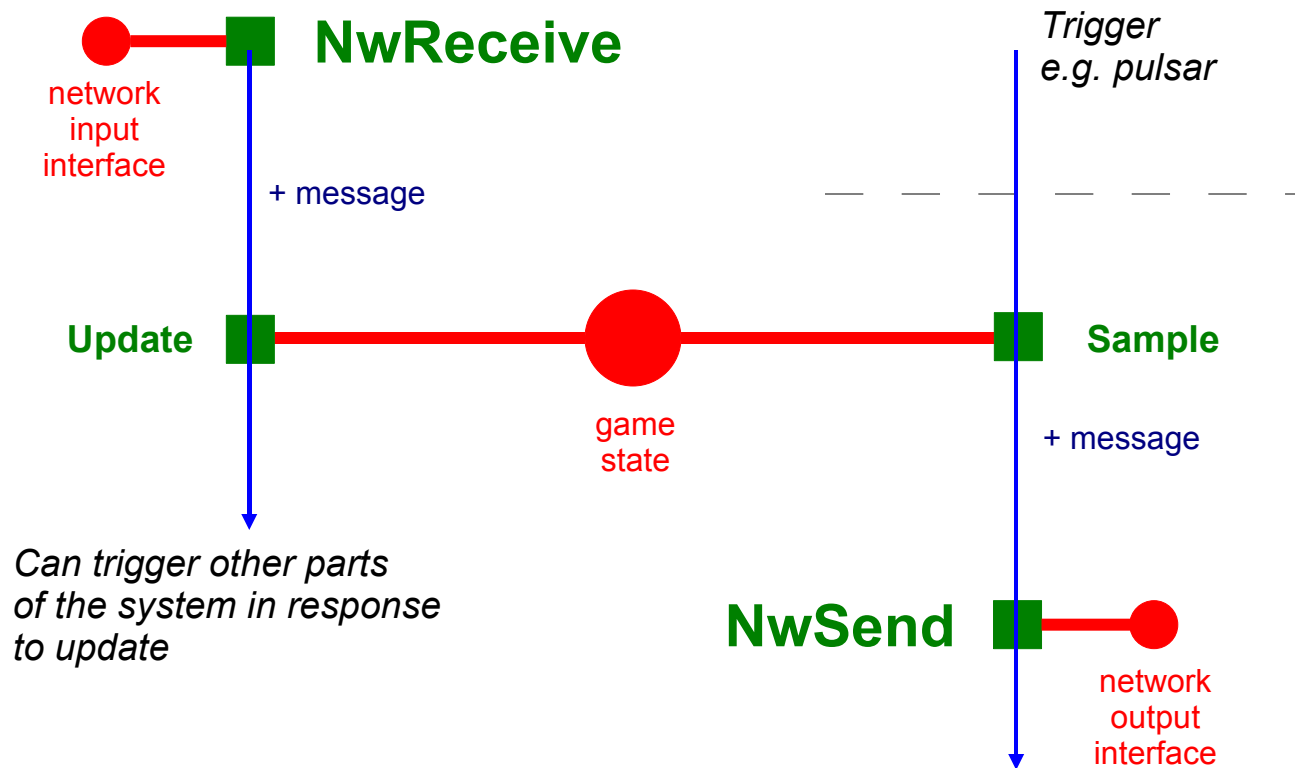
$$\text{example: } ma = \Sigma F$$

$$\text{accel.: } a \leftarrow g$$

$$\text{velocity: } v \leftarrow v + \Delta t a$$

$$\text{position: } p \leftarrow p + \Delta t v$$

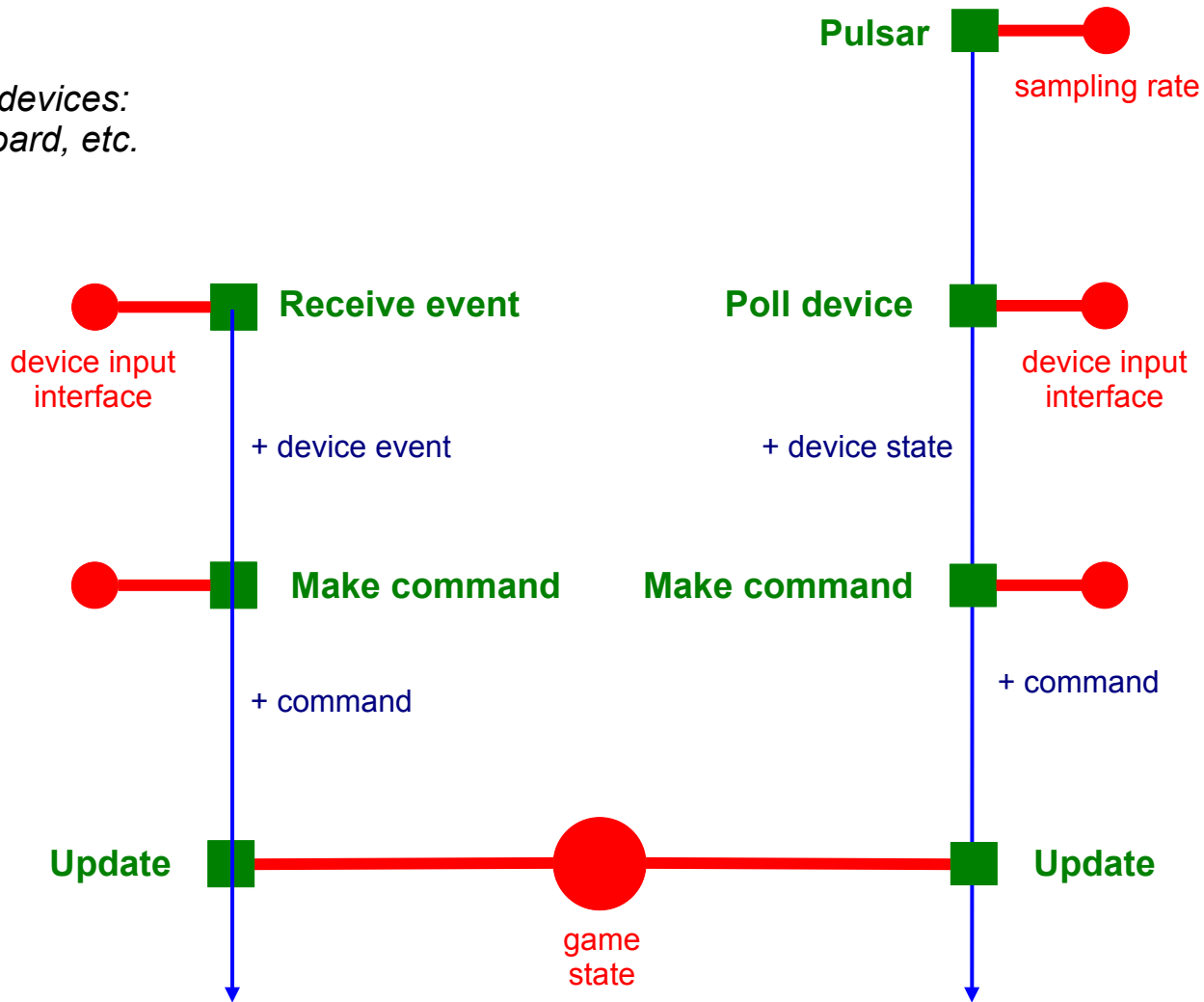
network receive and send



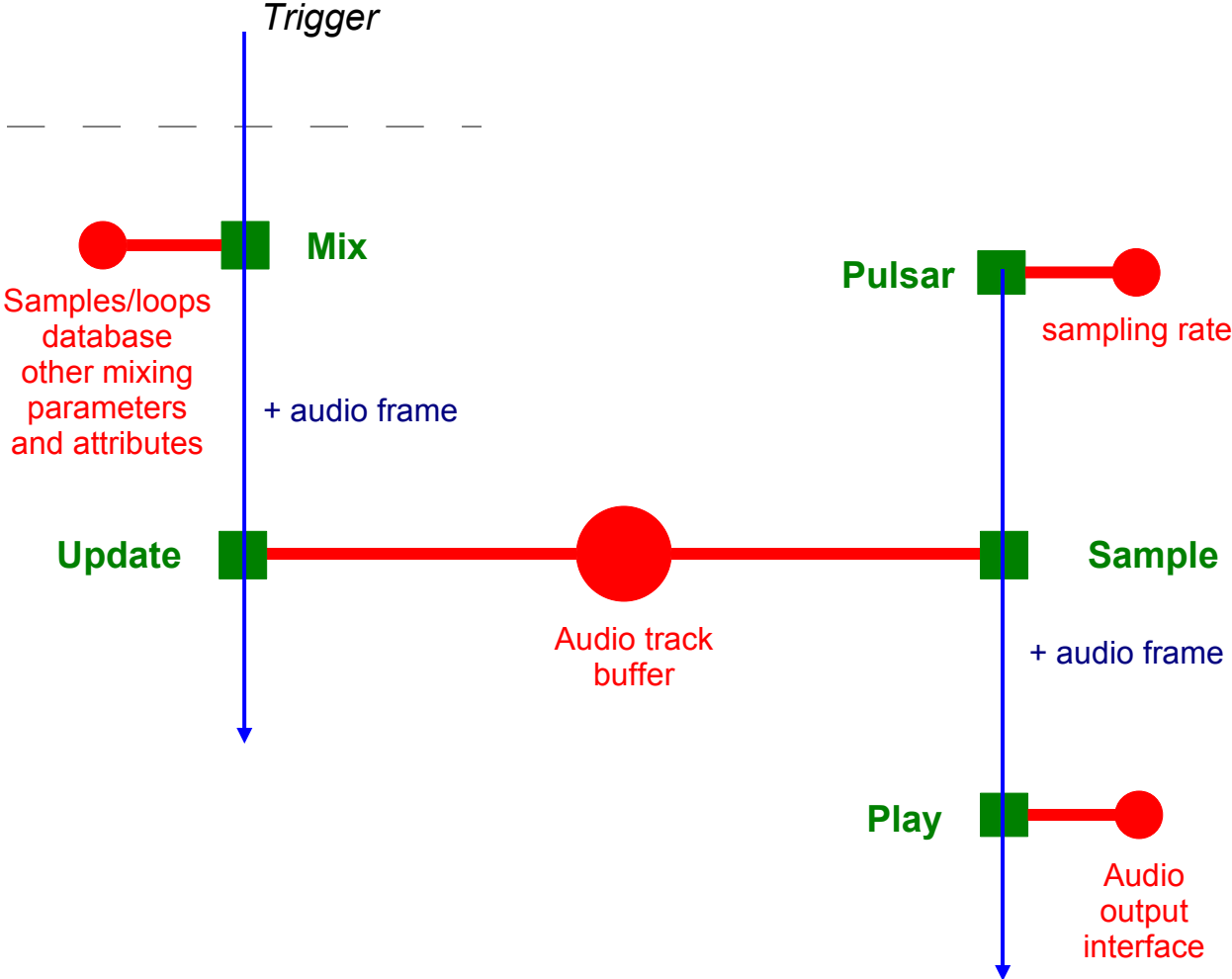
user interaction controls

State-based devices:
gamepad, sensors, etc

Event-based devices:
mouse, keyboard, etc.



music mixer and player



project architecture: a first draft...

client-server architecture

- game server

- player clients

 - platformer client

 - builder client

- maybe others...

game state

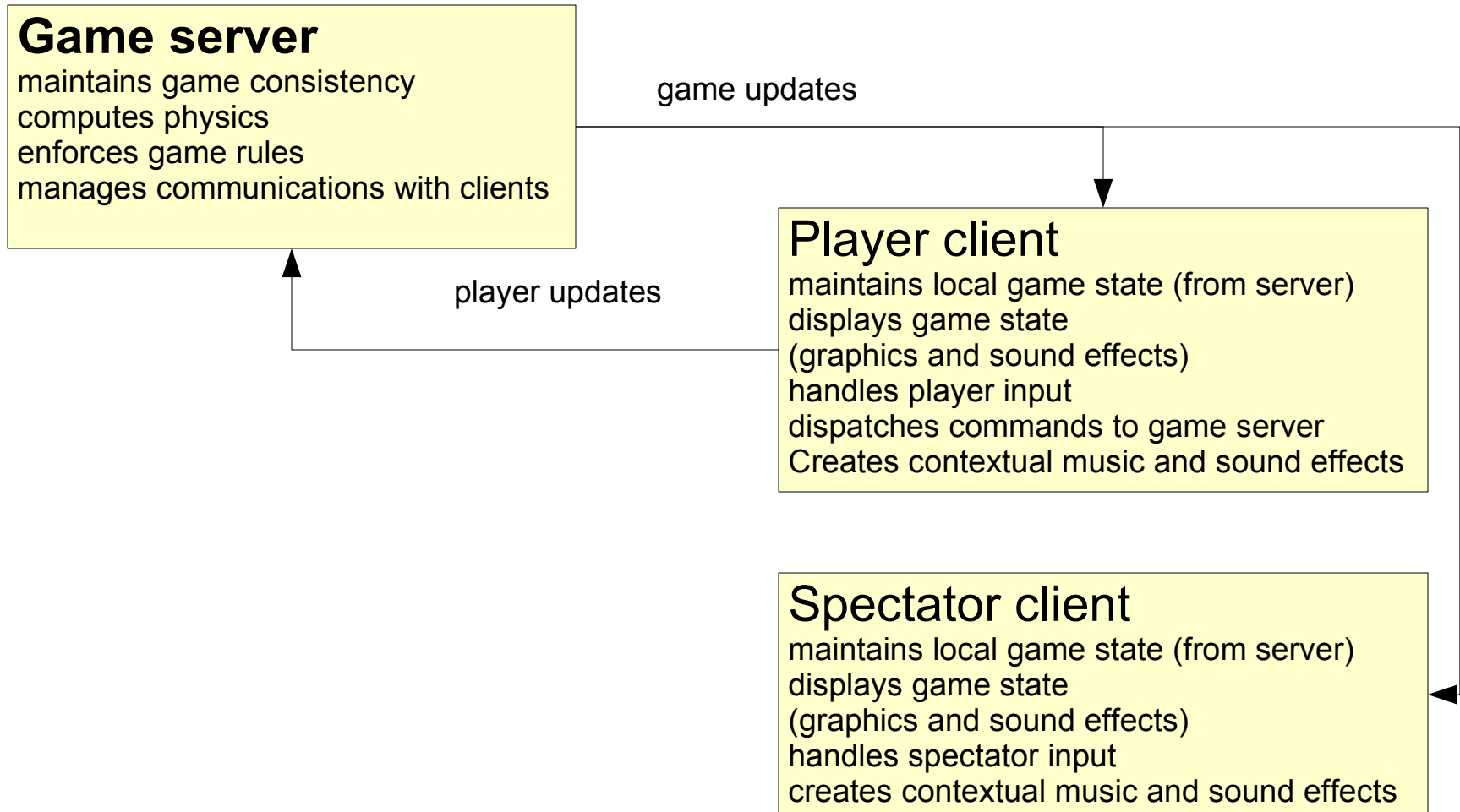
- description

- communication protocol for updates

other communication concerns

- chat? voice? video?

project architecture: servers and clients



modules

graphics

spectator / platformer / builder

physics and collision detection

controls

spectator / platformer/ builder - mouse / keyboard

game logic

game AI

music & sound

networking

player client architecture draft – conceptual level

Server Update

Graphics

+ Music&sound

Control

Pulsar



frame rate

+ empty pulse

ReceiveEvent



device input interface

+ device event

MakeCommand



+ command

MakeMessage



+ message

NwSendToServer



network output interface(s)

NwReceiveFromServer



network input interface(s)

+ message

N-update



Render



C-update



MakeMessage

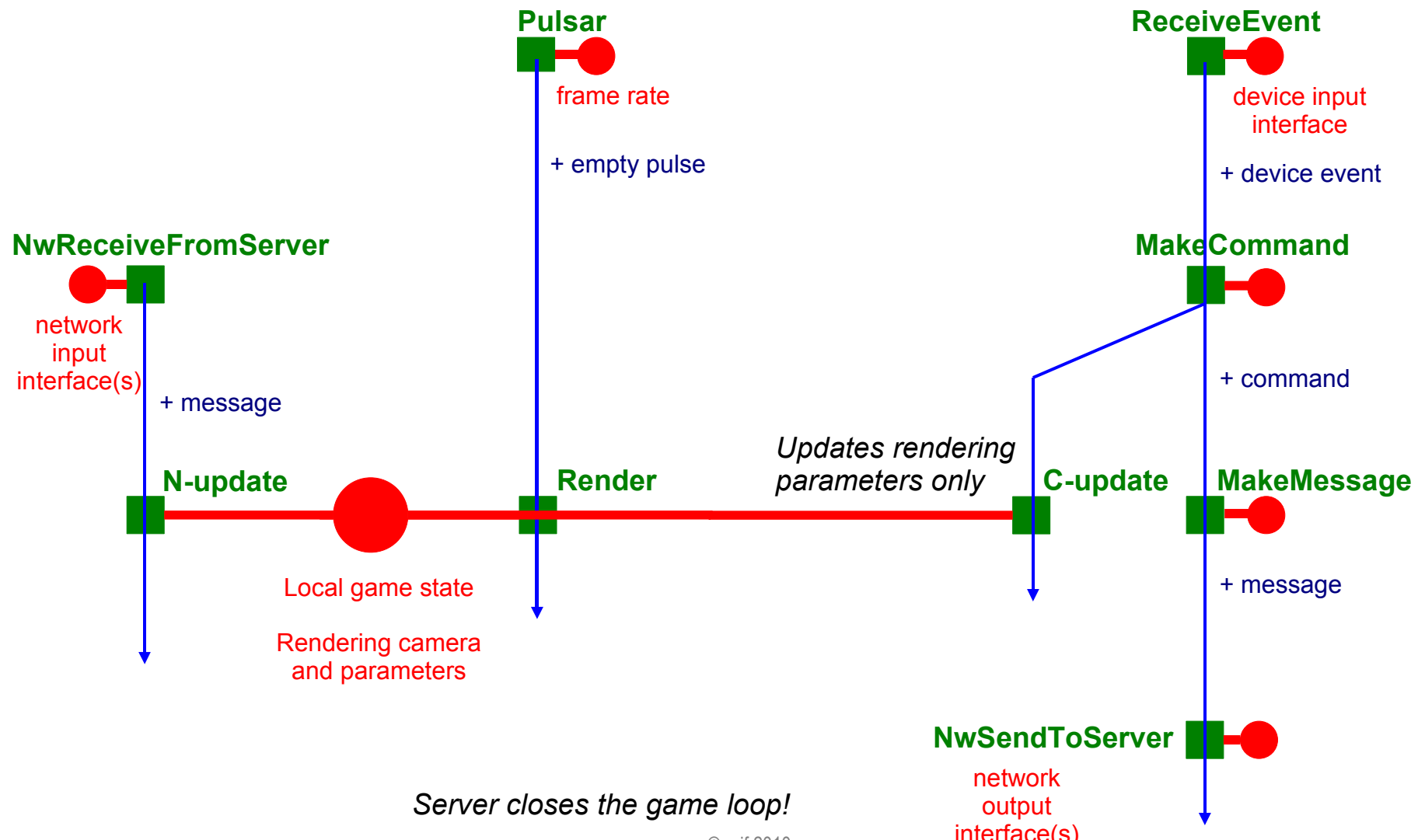


Local game state

Rendering camera and parameters

Updates rendering parameters only

Server closes the game loop!



music&sound engine architecture draft – conceptual level

