A Coordinated Matrix of RFID Readers as Interactions Input

Maxime Louvel, François Pacull
CEA-LETI Minatec Campus, France

Netware 2013, Barcelona, Spain
Outline

2 « mediation based » applications using an RFID based « interactive » table

Applications
Framework
LINC middleware
Hardware
Hardware
Interactive Table

LCD screen

RFID matrix

box containing
- a laptop connected to an external display
- raspberry pi connected to the LCD
Interactive Table

6x4x16 = 384 rfid readers

4 tiles

6 tiles

LCD 42" HD (1080p)
Interactive Table

A laptop
- External display (LCD or video projector)
- Application objects

Raspberry Pi
- Table LCD
- Application objects
Coordination Middleware
LINC
Coordination Middleware

Bag paradigm

- rd()
- get()
- put()

Abstraction Layer

- External services
- Sensors / actuators
- OS launcher (standalone application)
- Database
- Tuplespace
- Event system
- ...

Netware 2013, Barcelona, Spain
Coordination rule

Production rules

precondition
::
performance

Based only on the 3 primitives

precondition → rd() for inference engine
performance → rd(), get(), put()
Coordination rule

Production rules

precondition
::
performance

Based only on the 3 primitives

precondition → rd() for inference engine
performance → rd(), get(), put()
Transaction

Performance is a sequence of transactions

precondition
::
{
  op1
  op2
}
{
  op3
  op4
}.

- op1 AND op2 are done or none of them
- op3 AND op4 are done or none of them

If op1 or op2 consumes (get) a resource required by op3 or op4, the second transaction will abort.

If op1 or op2 are known to normally consume a resource required by op3 or op4, the second transaction commit only if the first aborted.

If op1, op2, op3 and op4 do not compete for any resource, then both transactions are independent.
Basics

Bags are grouped into objects

Operation

\[
[\text{objectname}, \text{bagname}].rd(f1,f2,f3)\]

\[
.get(f1,f2)\]

\[
.put(f1,f2,f3,f4)\]

string constant or variable
Framework
Framework

4 Objects

RFID
encapsulates the matrix of rfid readers

2D_Engine
encapsulates an HTML5 renderer (svg, js, bitmap, …)
manages what is displayed on the table

Display
encapsulates video reader

3D_Engine
encapsulates a 3D renderer (2 versions Ogre and OSG)
manages what is displayed on an external screen
RFID

**LogicalTag**
(physicalTagId, tagId)
e.g. ("030209348393", "tag_1"); ("030839320934", "tag_f_pacull")

**Type**
(tagId, type)
e.g. ("tag_1", "tangible_object"); ("tag_f_pacull", "badge")

**Mapping**
(tagId, objectld)
e.g. ("tag_1", "hourglass"); ("tag_f_pacull", "francois_pacull_id_badge")
RFID

**Position**
(tagId, posX, posY)
e.g. ("tag_f_pacull", "3", "12")

**TagStatus**
(tagId, status)
status: "in" or "out"
e.g. ("tag_1", "out") ; ("tag_f_pacull", "in")

**Area**
(areaId, areaDefinition)
areas on the table are defined as a set of points defining a polygon
e.g. ("area_51", "4,7 ; 3,5 ; 4,8 ; 2,14 ; 1,10")

**PositionArea**
(tagId, areaId)
e.g. ("tag_f_pacull", "area_51")
Rule example

["Rfid","TagStatus"].rd(tagId,"in") &
["Rfid","Type"].rd(tagId,"badge") &
["Rfid","PositionArea"].rd(tagId,"area_51")
::
{
  
  ["Rfid","TagStatus"].rd(tagId,"in");
  ["Rfid","PositionArea"].rd(tagId,"area_51");
  #some actions

}. 
2D Engine

Sprites
(spriteld,x,y, svgfile):
display a sprite (svg image) at the position x,y

MoveSpriteGrid
(spriteld,x,y,duration, nbsteps,renderlist)

Visibility
(spriteld,percent)
opacity

Media
(tagId, filename)

Background
(filename)
Exemple of interaction

[“Rfid”,”TagStatus”].rd(tagId,”in”) &
[“Rfid”,”Type”].rd(tagId,”background”) &
[“2DEngine”,”Media”].rd(tagId,filename)
::
{
[“2DEngine”,”background”].put(filename);
}. 
Display

videoPlayer
(playerId, videoname, posX, posY, width, height, orientation, soundTrack)
e.g. ("vlc", "video_grenoble", "5", "4", "400", "300", "up", "no")

videoPlayerCommand
(videoname, command):
commands → "stop", "pause", "resume", "fs_on", "fs_off"
e.g. ("video_grenoble", "pause")

video
(videoname, status)
status → started, finished, paused
e.g. ("video_grenoble", "paused")
RFID + Display

Rfid

Position
(tagId, posX, posY)
e.g. ("tag_video_1", 5, 4)

Display

videoPlayer
(playerId, videoname, posX, posY, width, height, orientation, soundTrack)
e.g. ("vlc", "grenoble.avi", 5, 4, 400, 300, up, no)

videoPlayerCommand
(videoname, command):
commands → "stop", "pause", "resume", "fs_on", "fs_off"
e.g. ("vlc", "pause")
Rule example (2)

["Rfid","Status"]=rd(tagId,"in") &
["Rfid","Type"]=rd(tagId,"video") &
["Rfid","Position"]=rd(tagId,x,y) &
INLINE :x2=x+3 &
INLINE :y2=y+2 &
["2DEngine","Filename"]=rd(tagId,filename) &
::
{
  ["Rfid","Status"]=rd(tagId,"in") ;
  ["Rfid","Position"]=rd(tagId,x,y) ;
  ["Display","videoPlayer"]=rd("vlc",filename,x2,y2, "400","300","up","no")
}.
Application (1)
User Panel

n products

4 properties

m users

Each user gives her advice (positive / negative) for each of the n products according to the properties

The lighter the better
Example

Smartphones: ...

4 properties

Autonomy
Price
Impact (feeling, brand)
# of useful applications
Scenario

User registration

Video explaining the process

n rounds

   Video of smartphone (opt)

   4 rounds
       Video or data illustrating property for this smartphone (opt)

   vote collection

display result
Tagged objects

- badge for user id (m)
- registration card (start the registration phase)
- smartphone pictures (n)
- property cards (4)
- hourglass (start the timer for the vote)
- video cards (up to n * 5)
- modality cards for video (fs, sound, ...)
Registration

["Rfid","Status"].rd("tag_registration","in") &
["Rfid","Status"].rd(tagId,"in") &
["Rfid","Type"].rd(tagId,"badge") &
::
{
  ["Rfid","Status"].rd("tag_registration","in") &
  ["Rfid","Status"].rd(tagId,"in") &
  ["Application","Users"].put(tagId)
}.

start when registration card is placed on the table
stop immediately when registration card is removed
as result all the registered users are in the bag **Users**
Vote

["Rfid","Status"].rd("tag_vote","in") &
::
{
   ["Application","Step"].put("vote_started")
   ["2D_Engine","Background"].put("vote_bg")
}.

30 secs

["Application","Step"].rd("vote_started") &
SLEEP: 30
::
{
   ["Application","Step"].get("vote_started")
   ["2D_Engine","Background"].put("end_vote_bg")
}.

["Application","Step"].rd("vote_started") &
["Rfid","Status"].rd("tag_vote","out") &
::
{
   ["Application","Step"].get("vote_started")
   ["Rfid","Status"].rd("tag_vote","out") &
   ["2D_Engine","Background"].put("end_vote_bg")
}.
Vote

["Application","Step"].rd("vote_started") &
["Rfid","Status"].rd(tag_product,"in") &
["Rfid","Type"].rd(tag_product,"product") &
["Rfid","Status"].rd(tag_property,"in") &
["Rfid","Type"].rd(tag_property,"property") &
["Application","Users"].rd(user_tagId) &
["Rfid","Status"].rd(user_tagId,"in") &
::
{
["Application","Step"].rd("vote_started")
["Rfid","Status"].rd(tag_product,"in")
["Rfid","Status"].rd(tag_property,"in")
["Rfid","Status"].rd(user_tagId,"in")
["Rfid","Area"].rd("positive",user_tagId)
["Application","Vote"].put(tag_product, tag_property,user_tagId, "+")
}
{
["Application","Step"].rd("vote_started")
["Rfid","Status"].rd(tag_product,"in")
["Rfid","Status"].rd(tag_property,"in")
["Rfid","Status"].rd(user_tagId,"in")
["Rfid","Area"].rd("negative",user_tagId)
["Application","Vote"].put(tag_product, tag_property,user_tagId, "-"")
}.
Application (2)
Urban Mediation
Conclusion

* Reusability of framework
  Rfid, 2D_Engine, Display are the same in the 2 applications
  3D_Engine has been added for the second application
* Linc Middleware can accommodate of light CPU
  2D_Engine runs on a Raspberry Pi
* Hardware
  Promising innovation area in term of usage and mediation
* Future work
  Extend Urban Mediation Application