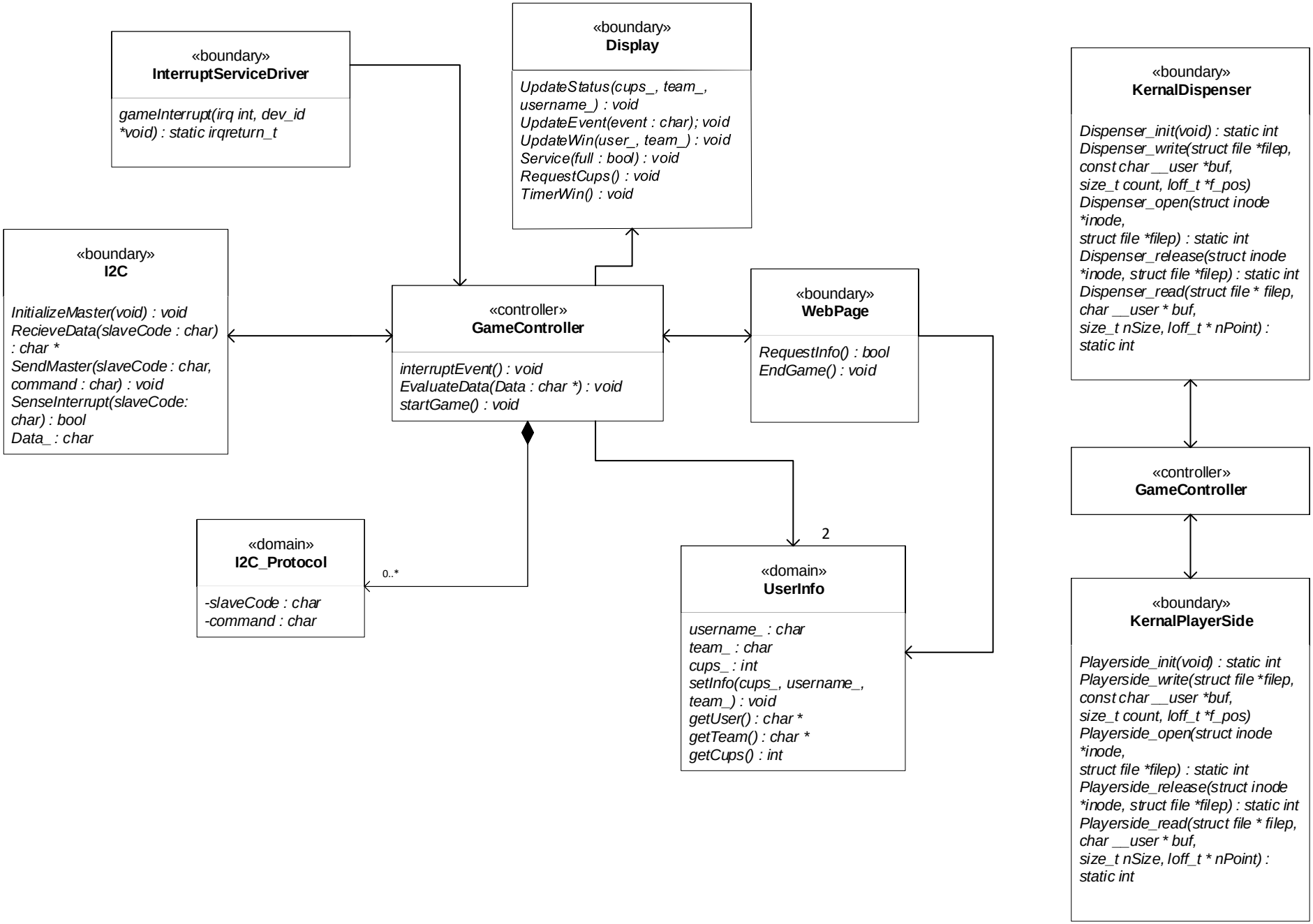
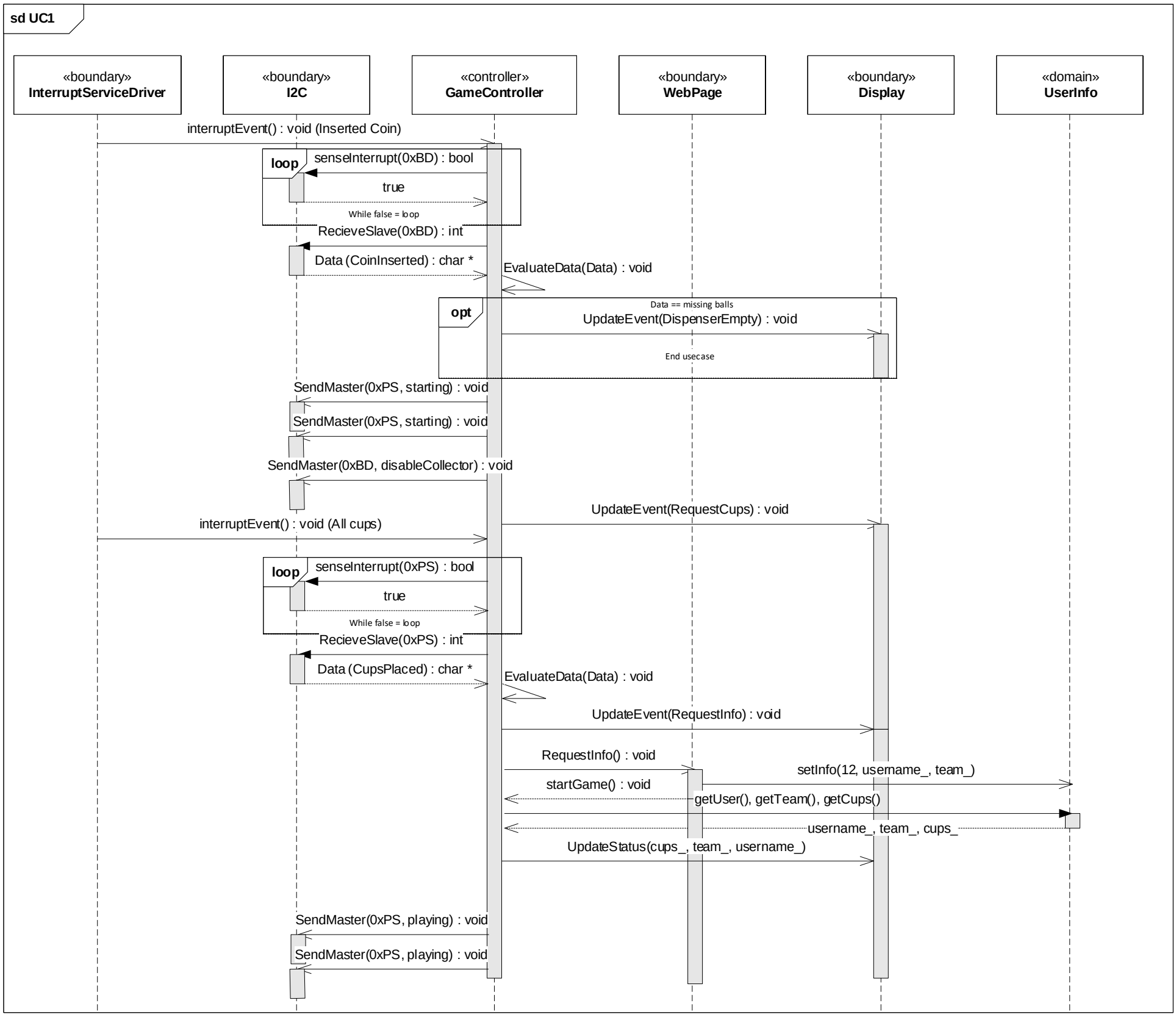
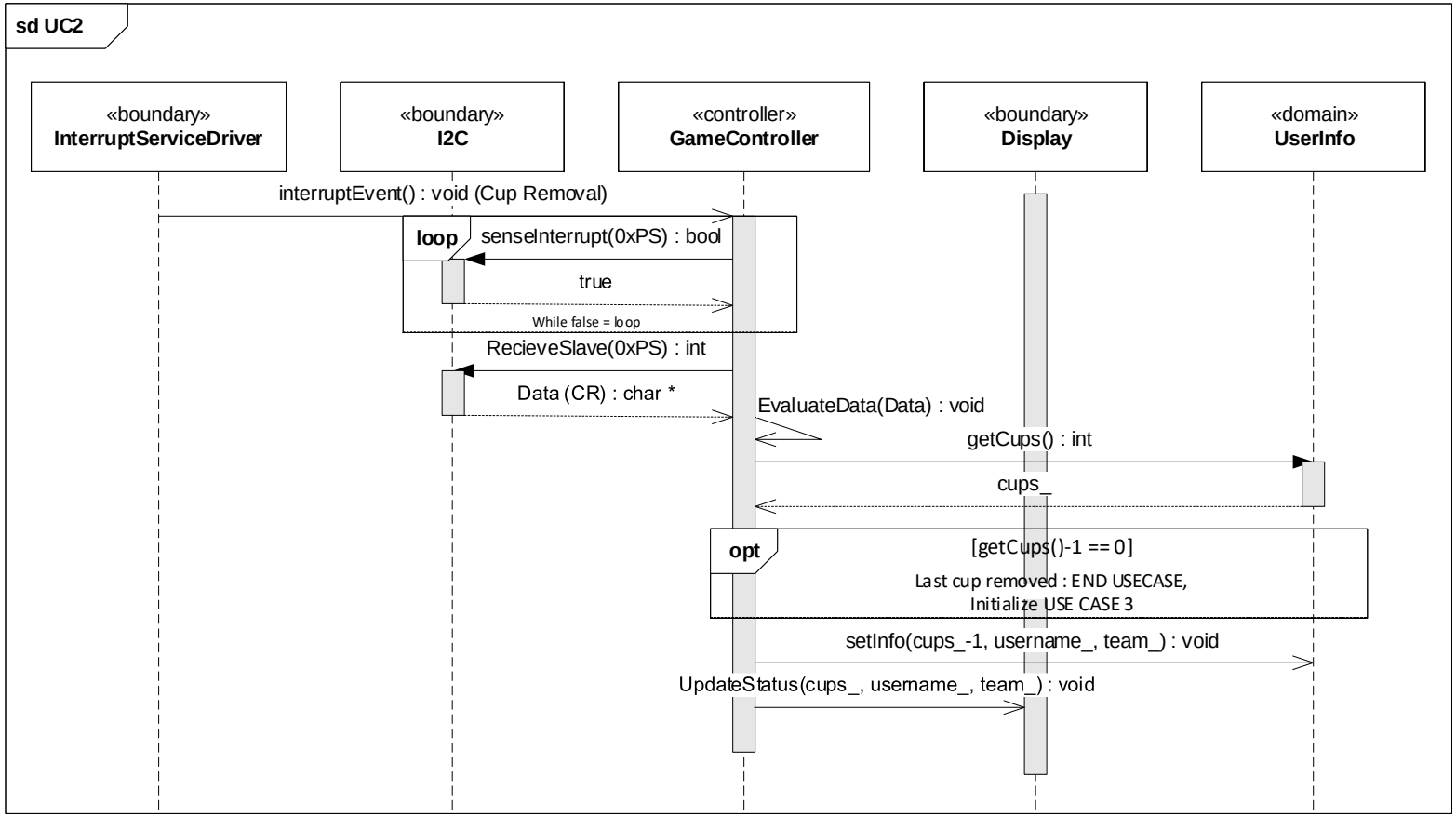
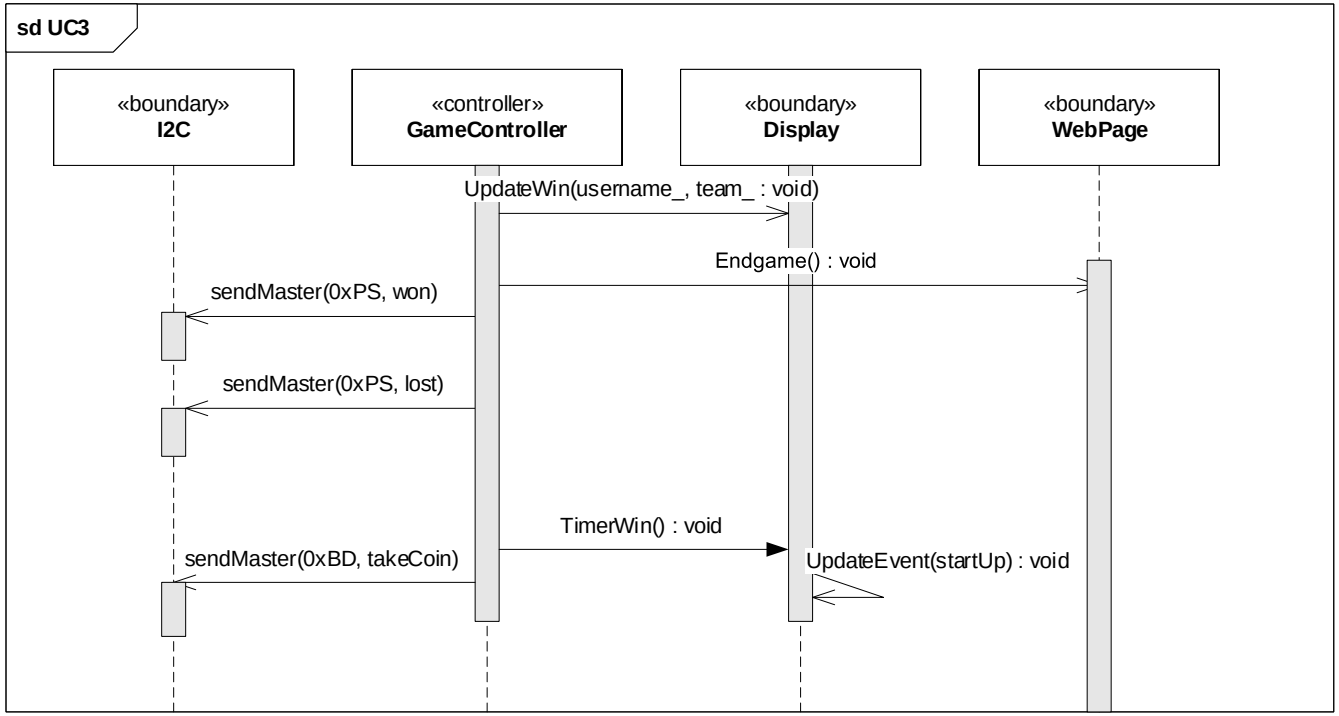
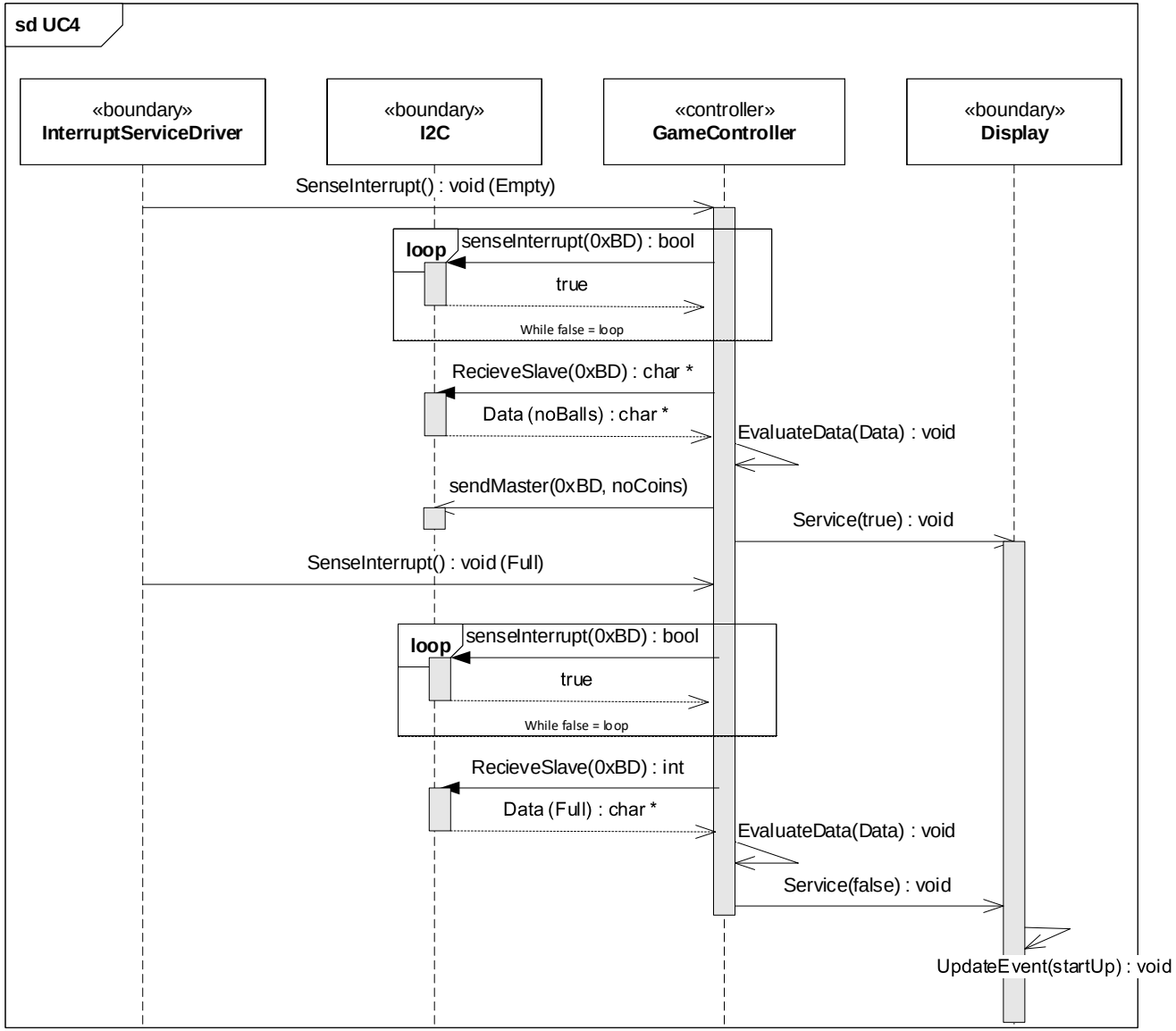
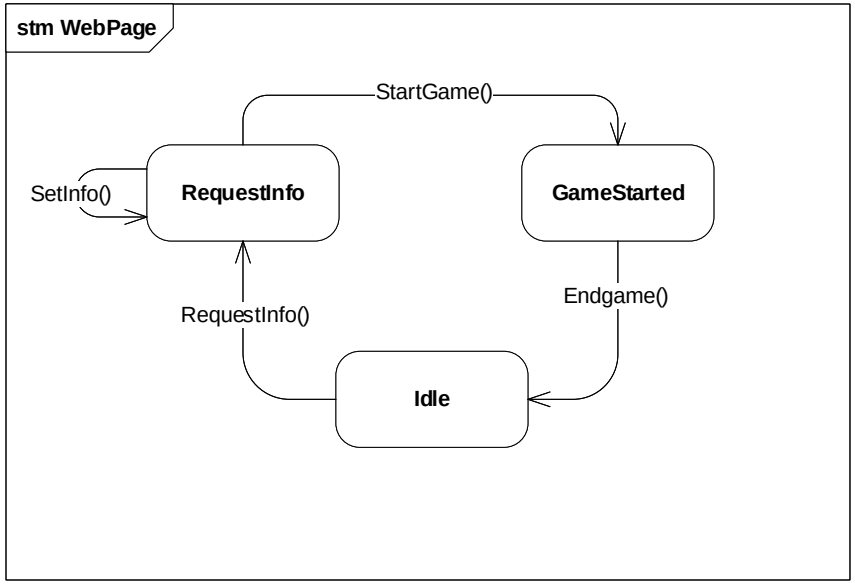

«boundary»
InterruptServiceDriver
gameInterrupt(irq int, dev_id *void) : static irqreturn_t
«boundary»
Display
UpdateStatus(cups_, team_, username_) : void
UpdateEvent(event : char); void
UpdateWin(user_, team_) : void
Service(full : bool) : void
RequestCups() : void
TimerWin() : void
«boundary»
I2C
InitializeMaster(void) : void
RecieveData(slaveCode : char) : char *
SendMaster(slaveCode : char, command : char) : void
SenseInterrupt(slaveCode: char) : bool
Data_ : char
«controller»
GameController
interruptEvent() : void
EvaluateData(Data : char *) : void
startGame() : void
«boundary»
WebPage
RequestInfo() : bool
EndGame() : void
«domain»
I2C_Protocol
-slaveCode : char
-command : char
0..*
2
«domain»
UserInfo
username_ : char
team_ : char
cups_ : int
setInfo(cups_, username_, team_) : void
getUser() : char *
getTeam() : char *
getCups() : int
«boundary»
KernalDispenser
Dispenser_init(void) : static int
Dispenser_write(struct file *filep, const char __user *buf, size_t count, loff_t *f_pos)
Dispenser_open(struct inode *inode, struct file *filep) : static int
Dispenser_release(struct inode *inode, struct file *filep) : static int
Dispenser_read(struct file * filep, char __user * buf, size_t nSize, loff_t * nPoint) : static int
«controller»
GameController
«boundary»
KernalPlayerSide
Playerside_init(void) : static int
Playerside_write(struct file *filep, const char __user *buf, size_t count, loff_t *f_pos)
Playerside_open(struct inode *inode, struct file *filep) : static int
Playerside_release(struct inode *inode, struct file *filep) : static int
Playerside_read(struct file * filep, char __user * buf, size_t nSize, loff_t * nPoint) : static int

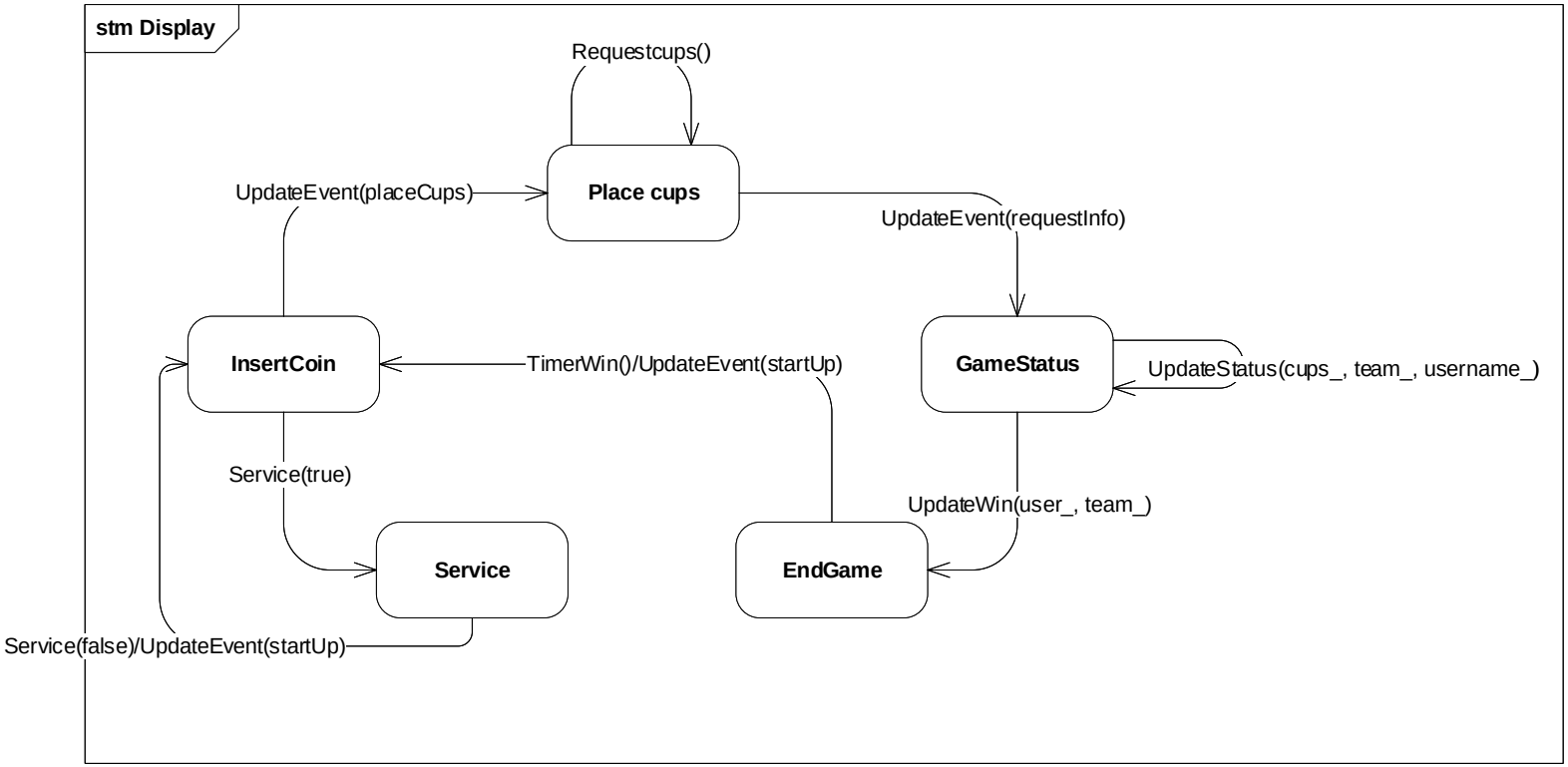

sd UC1
«boundary»
InterruptServiceDriver
«boundary»
I2C
«controller»
GameController
«boundary»
WebPage
«boundary»
Display
«domain»
UserInfo
interruptEvent() : void (Inserted Coin)
loop
senseInterrupt(0xBD) : bool
true
While false = loop
RecieveSlave(0xBD) : int
Data (CoinInserted) : char *
EvaluateData(Data) : void
opt
Data == missing balls
UpdateEvent(DispenserEmpty) : void
End usecase
SendMaster(0xPS, starting) : void
SendMaster(0xPS, starting) : void
SendMaster(0xBD, disableCollector) : void
UpdateEvent(RequestCups) : void
interruptEvent() : void (All cups)
loop
senseInterrupt(0xPS) : bool
true
While false = loop
RecieveSlave(0xPS) : int
Data (CupsPlaced) : char *
EvaluateData(Data) : void
UpdateEvent(RequestInfo) : void
RequestInfo() : void
setInfo(12, username_, team_)
startGame() : void
getUser(), getTeam(), getCups()
username_, team_, cups_
UpdateStatus(cups_, team_, username_)
SendMaster(0xPS, playing) : void
SendMaster(0xPS, playing) : void

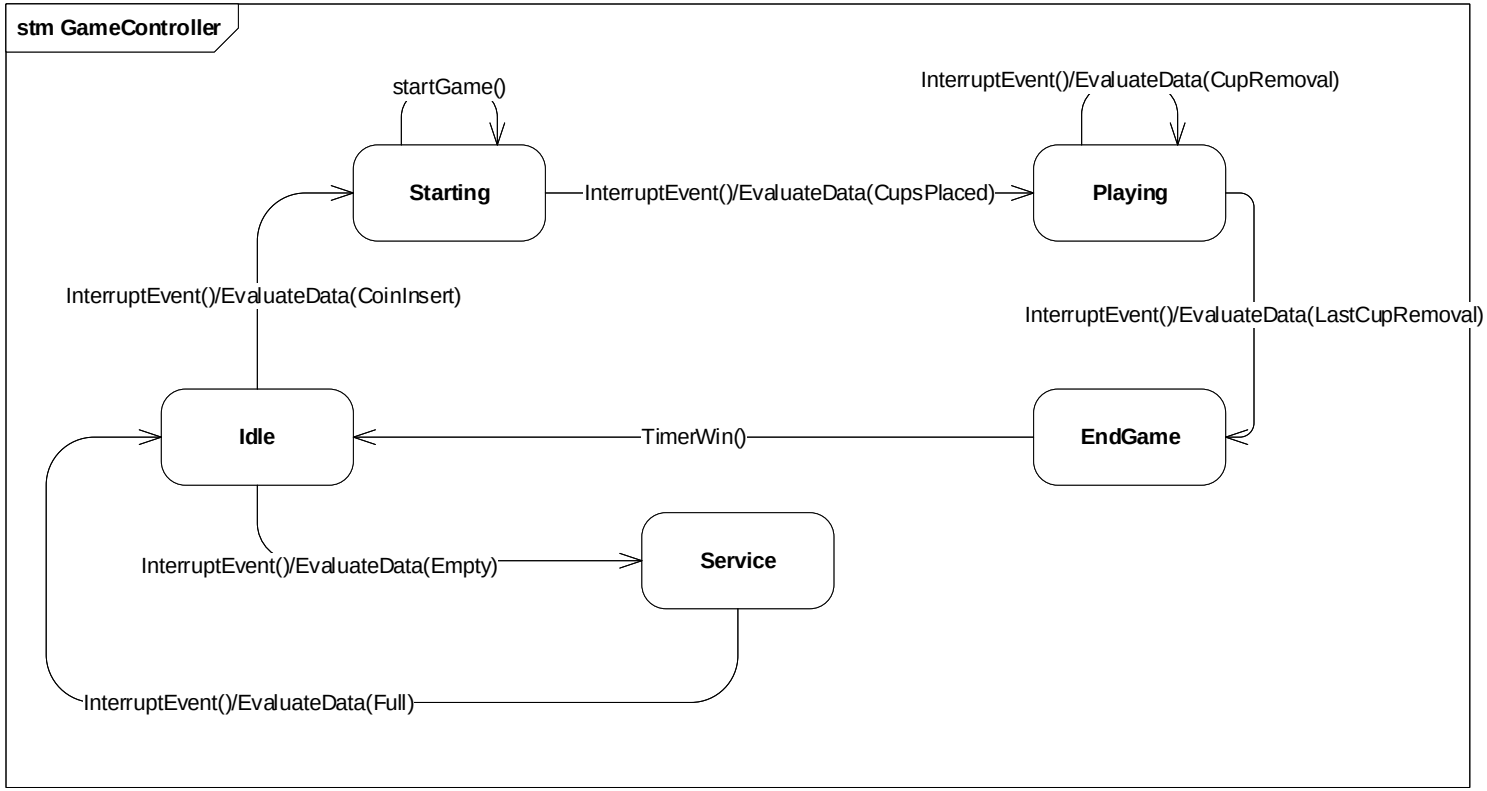

sd UC2
«boundary»
InterruptServiceDriver
«boundary»
I2C
«controller»
GameController
«boundary»
Display
«domain»
UserInfo
interruptEvent() : void (Cup Removal)
loop
senseInterrupt(0xPS) : bool
true
While false = loop
RecieveSlave(0xPS) : int
Data (CR) : char *
EvaluateData(Data) : void
getCups() : int
cups_
opt
[getCups()-1 == 0]
Last cup removed : END USECASE,
Initialize USE CASE 3
setInfo(cups_-1, username_, team_) : void
UpdateStatus(cups_, username_, team_) : void


sd UC3
«boundary»
I2C
«controller»
GameController
«boundary»
Display
«boundary»
WebPage
UpdateWin(username_, team_ : void)
Endgame() : void
sendMaster(0xPS, won)
sendMaster(0xPS, lost)
TimerWin() : void
sendMaster(0xBD, takeCoin)
UpdateEvent(startUp) : void


sd UC4
«boundary»
InterruptServiceDriver
«boundary»
I2C
«controller»
GameController
«boundary»
Display
SenseInterrupt() : void (Empty)
loop
senseInterrupt(0xBD) : bool
true
While false = loop
RecieveSlave(0xBD) : char *
Data (noBalls) : char *
EvaluateData(Data) : void
sendMaster(0xBD, noCoins)
Service(true) : void
SenseInterrupt() : void (Full)
loop
senseInterrupt(0xBD) : bool
true
While false = loop
RecieveSlave(0xBD) : int
Data (Full) : char *
EvaluateData(Data) : void
Service(false) : void
UpdateEvent(startUp) : void


stm WebPage
StartGame()
SetInfo()
RequestInfo
GameStarted
Endgame()
RequestInfo()
Idle


stm Display
Requestcups()
Place cups
UpdateEvent(placeCups)
UpdateEvent(requestInfo)
InsertCoin
TimerWin()/UpdateEvent(startUp)
GameStatus
UpdateStatus(cups_, team_, username_)
Service(true)
UpdateWin(user_, team_)
Service
EndGame
Service(false)/UpdateEvent(startUp)


stm GameController
startGame()
InterruptEvent()/EvaluateData(CupRemoval)
Starting
InterruptEvent()/EvaluateData(CupsPlaced)
Playing
InterruptEvent()/EvaluateData(CoinInsert)
InterruptEvent()/EvaluateData(LastCupRemoval)
Idle
TimerWin()
EndGame
InterruptEvent()/EvaluateData(Empty)
Service
InterruptEvent()/EvaluateData(Full)