UNIVERSITETET
I OSLO

INF5150 H07

Obligatorisk oppgave 2:

Survival of the SMSest — the game

Gruppe 1:

Finn Christian Brøndal (finnc)
Flemming Hansen
Hans Aage Huru (hansahu)
Tore Engvig (toreen)
Tore Vatnan (torevatn)

26. november 2007

1. Innledning

Dette er et overlevelsesspill hvor spillerne bruker mobiltelefoner for å sende SMS for å prøve å slå ut de andre spillerne og for å beskytte seg mot angrep fra andre.

2. Bruk av spillet

2.1 Registrering

For å melde seg på spillet må man sende en sms til ”2034”, sms skal inneholde følgende;

- ”stud1 konto finnc reg ” brukernavn

F.eks. ”stud1 konto finnc reg hans”

Om brukernavnet ikke er registrert før, og du ikke har meldt deg på tidligere skal du motta meldingen;

- "Welcome to TheGame. You are registered as: hans”

Du kan ikke registrere deg flere ganger.

2.2 Start av spillet

Administrator starter spillet, og du som deltager må vente på en melding, hvor du blir bedt om å bli med eller ikke.

Meldingen i fra administrator skal se ut som følger;

- "Would you like to participate in a game? If not send no or don't respond to this sms."

Om du ønsker å delta sender du en sms til “2034” som er av typen;

- ”stud1 konto finnc yes”.

Når administrator finner at han vil starte spillet, skal du motta meldingen

- "The game has begun. Happy hunting :)”

Og da er spillet i gang.

2.3 Spillets gang

Spillet går ut på å finne de andre deltagerne og deretter ”strike” dem, slik at de dør, eller mister poeng. Alle spillere får utdelt 1000 poeng og 500 ”shieldpoeng” ved oppstart, og den som har mest poeng ved spillets slutt, eller har ”striket” alle andre ut, er vinneren.

2.3.1 Finne andre spillere (light up)

For å finne andre spillere kan man sende en SMS med kommandoen **lightup**. Om det er spillere i nærheten, vil man motta en SMS med informasjon om dette, og man kan deretter ”strike” disse.

Lightup kommandoen fungerer slik at man legger ved en parameter for å angi distansen man vil dekke. Distansen angir radien på en sirkel hvis du som utfører kommandoen er i sentrum. Ved å angi ”lightup 400” vil alle medspillere som er innefor en radius på 400 meter i forhold til deg få melding om at de er sett, og du får en melding som inneholder brukernavnene på alle som er innenfor samme radius. Nå har du mulighet til å ”strike” disse.

Prisen for en ”lightup” er kvadratroten av radiusen du spesifiserer. M.a.o vil ”lightup 400” koste deg 20 poeng.

Om du prøver å lyse opp et område som er større enn det du har poeng til, vil det opplyste området bli redusert til det maksimum du har råd til. Du vil beholde ett poeng uansett, slik at du fremdeles er med i spillet.

Kommandoen for dette er i sin helhet;

- ”stud1 konto finnc lightup ” radius.

2.3.2 Beskytte seg selv (shield)

For å beskytte seg selv mot andres ”strike” kan man sette opp et beskyttende skjold. Dette gjøres enkelt ved en sms melding på formen;

- ”stud1 konto finnc shield ” styrke tid.

Styrken angir antall poeng som skjoldet får, og som eventuelle ”strikere” må slå igjennom før de går løs på poengene dine.

Tiden angir levetiden på skjoldet i sekunder.

Kostnaden for et skjold er (styrke * tid) / 200.

Skjoldet er aktivt så lenge tiden ikke er utløpt og forringes ikke av eventuelle ”strikes” med mindre ”striken” er større enn skjoldet.

2.3.3 Skyte andre (strike)

For å skyte de andre må det gis en sms melding på formen;

- ”stud1 konto finnc strike ” brukernavn styrke.

Hvor brukernavn er brukernavnet til den motspilleren ”striker”.

Styrke er styrken på ”striken”.

Effekten av en ”strike” avhenger av avstanden til deg og motspilleren, og styrken du har angitt. Styrken beregnes på følgende måte; Styrken / ((kvadratroten av avstanden) / 10).

Om motspilleren du ”striker” har et skjold oppe, og dette er større en kraften i din ”strike”, vil ”striken” ikke ha noen effekt.

Hvis derimot styrken på ”striken” er større enn skjoldet til motspilleren, eller motspilleren ikke har noe skjold, vil poengene til motspilleren minske med (styrke - skjold) poeng.

Om du er heldig eller dyktig og redusere motspillerens poeng ned til 0 eller mindre, mottar du alle poengene til denne motspilleren, mens motspilleren dør.

2.3.4 Tilfeldig element

I systemet er det lagt inn et tilfeldig element, slik at en ”striker” vil i snitt hver trettisjette gang ”drepe” motspilleren uansett forhold mellom poeng og styrke.

2.4 Vinner av spillet

Når alle dine motspillere er ”døde” har du vunnet spillet.

Om spillet avsluttes av administrator før dette skjer vinner den spilleren som har mest poeng.

3. Administrering av spillet

Den spilleren som er registrert som ”admin” er den spilleren som kan administrere spillet.

Administratoren er ansvarlig for å annonsere spill, starte spill og stoppe spill. I tillegg har administrator mulighet til å se hvor de andre spillerne er.

Den spilleren som registrerer seg som ”admin” er administrator.

3.1 Annonsere spill

Administrator kan annosere et nytt spill gjennom SMS-meldingen;

- ”stud1 konto finnc announce”.

3.2 Starte spill

Administrator kan starte et annonsert spill gjennom sms meldingen;

”stud1 konto finnc start”.

3.3 Stoppe spill

Administrator kan starte et annonsert spill gjennom sms meldingen;

”stud1 konto finnc stop”.

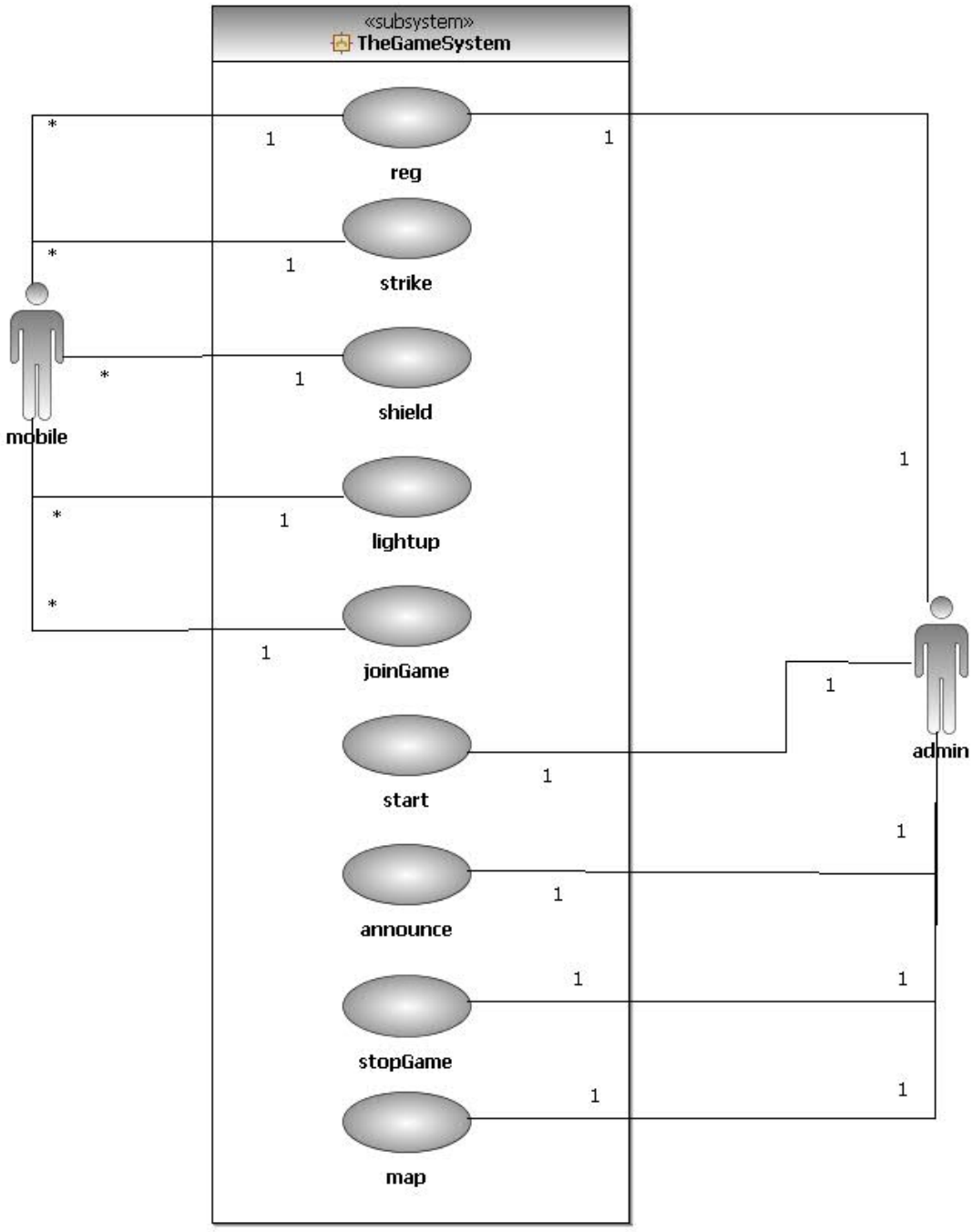
3.4 Se spillere (map)

Administrator kan se hvor de andre spillerene befinner seg v.h.a sms meldingen;

- ”stud1 konto finnc map”.

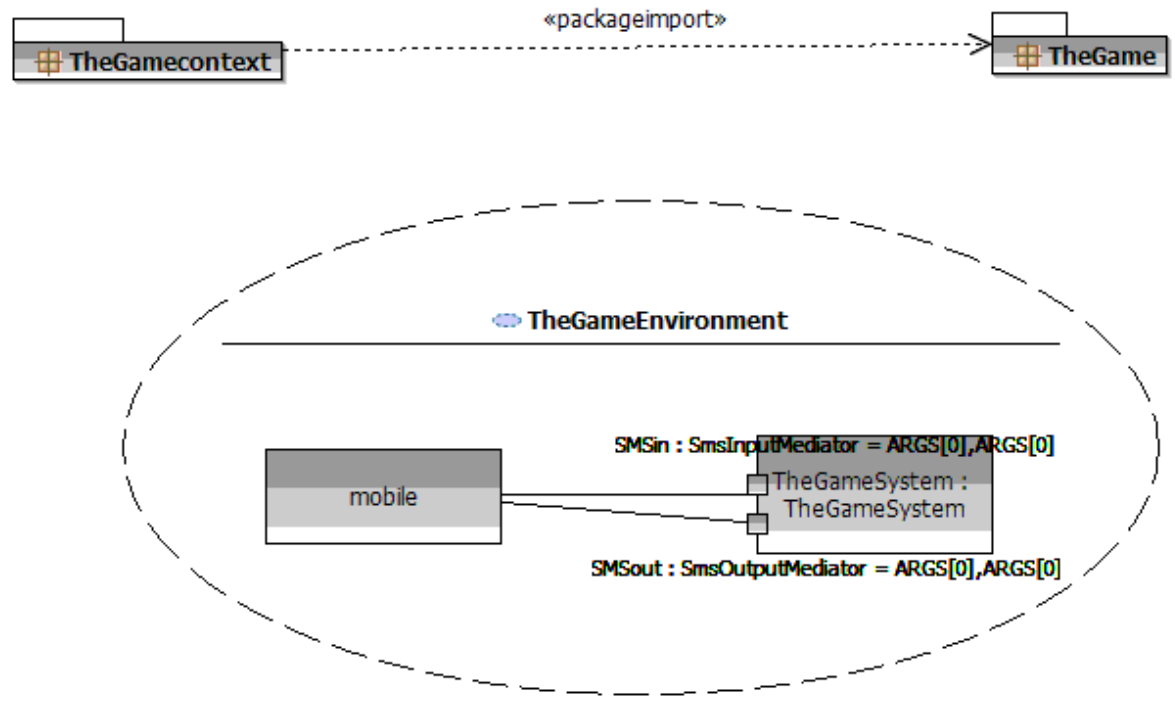
Det vil da produseres en kml fil på system området som kan lastes inn i Google Earth, som administrator kan åpne, og få oversikten.

4. Brukshistorier



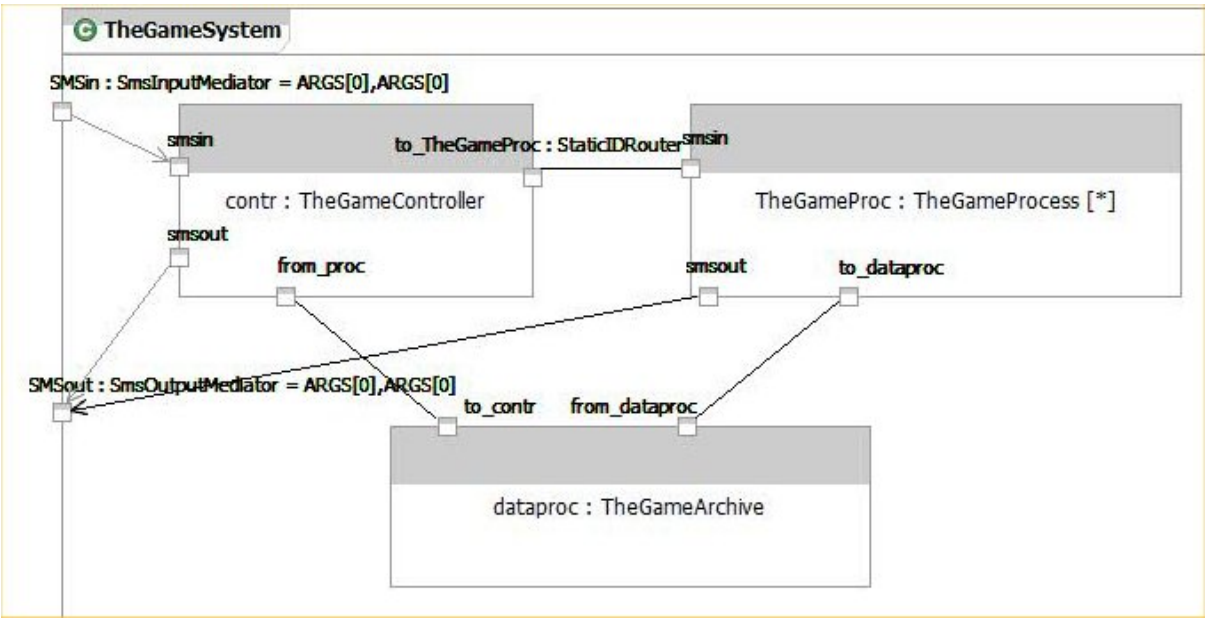
5. Arkitektur

5.1 Samarbeidsdiagram



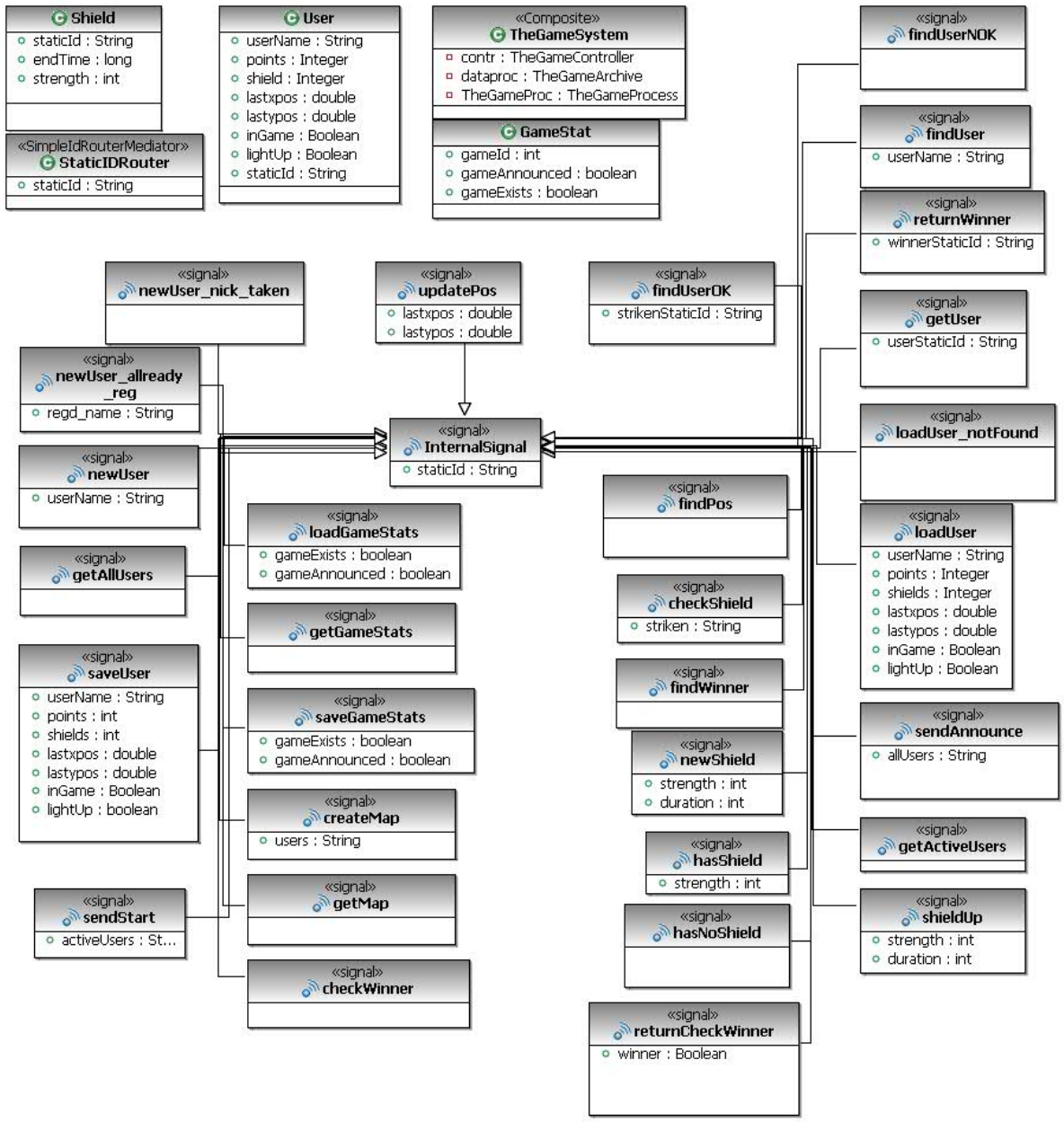
Den stiplede ovale sirkelen viser forholdet mellom komponentene **mobile** og **TheGameSystem**. Linjene mellom komponentene representerer informasjonsflyten. Inn- og utgående informasjon for systemet skjer via porter (representert ved de små kvadratene).

5.2 Kompositttrukturdiagram



Kompositttrukturdiagrammet viser hvordan systemet er bygd opp av deler. Innkommende data fra PATS går til **smsin** i **contr** før de rutes videre til **TheGameProc**. Herfra går det signaler dels tilbake til PATS via **smsout**, dels videre til **dataproc** og så tilbake til **contr**.

5.3 Klassediagram

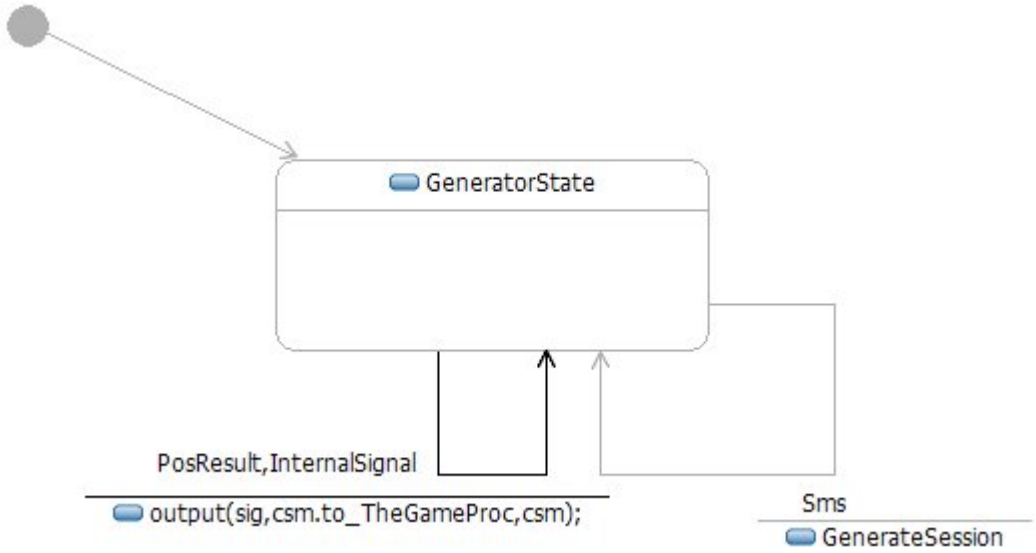


Diagrammet viser klassene som inngår i systemet.

6. Sekvensdiagrammer og tilstandsmaskiner

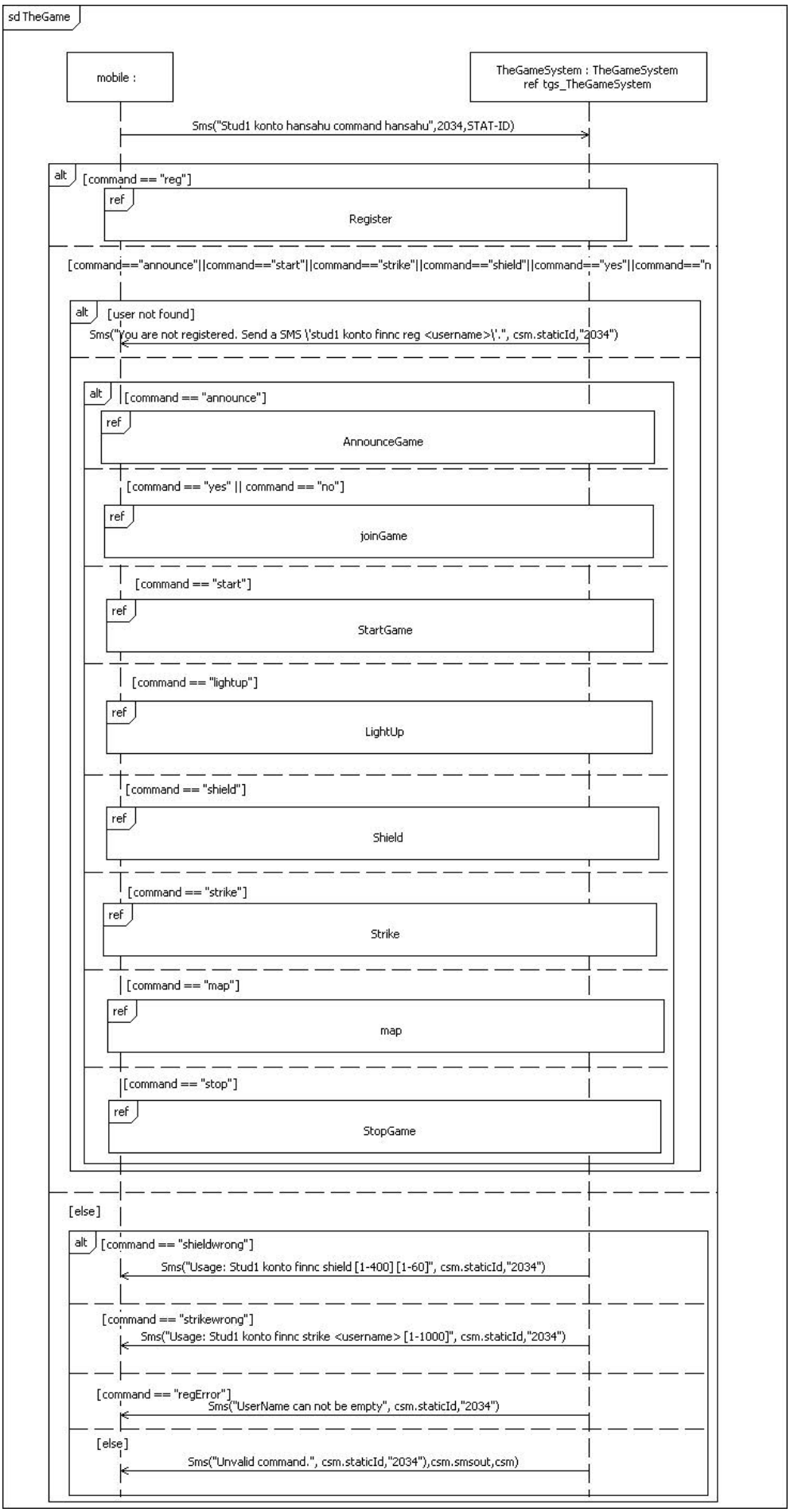
I alle sekvensdiagrammer skal **alt** forstås som **xalt**.

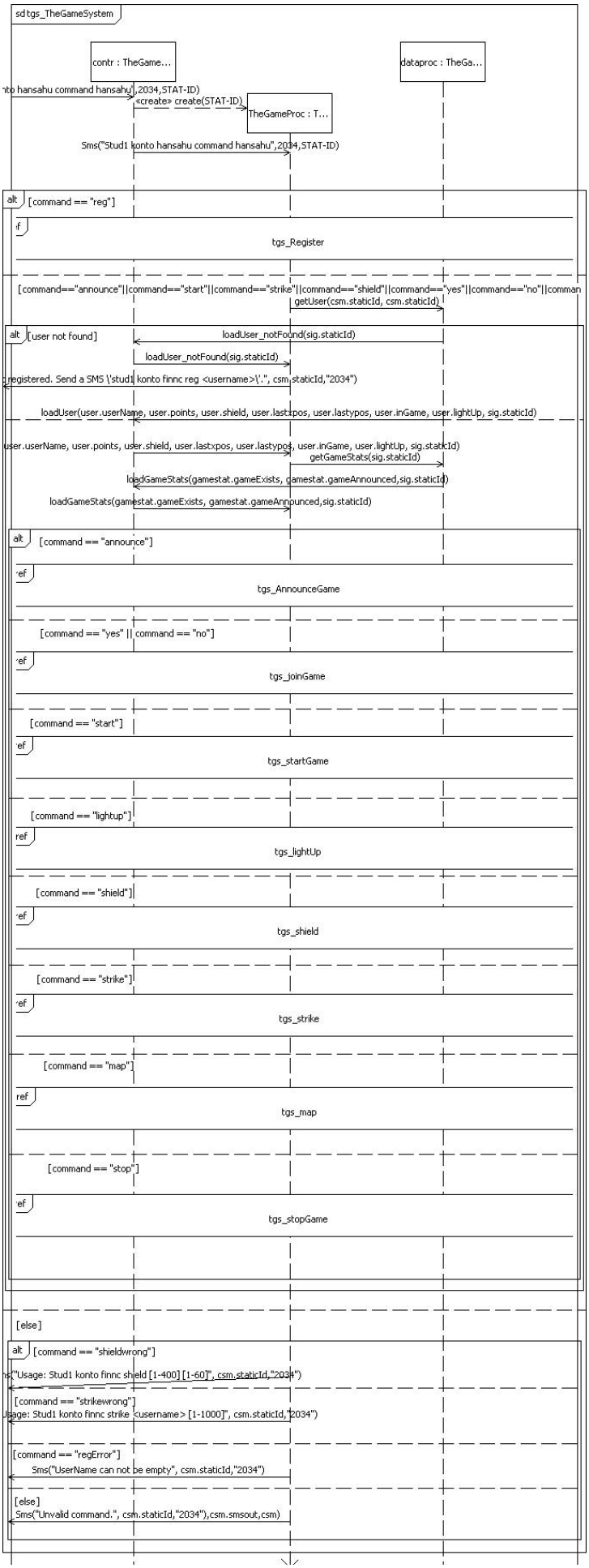
6.1 Controller

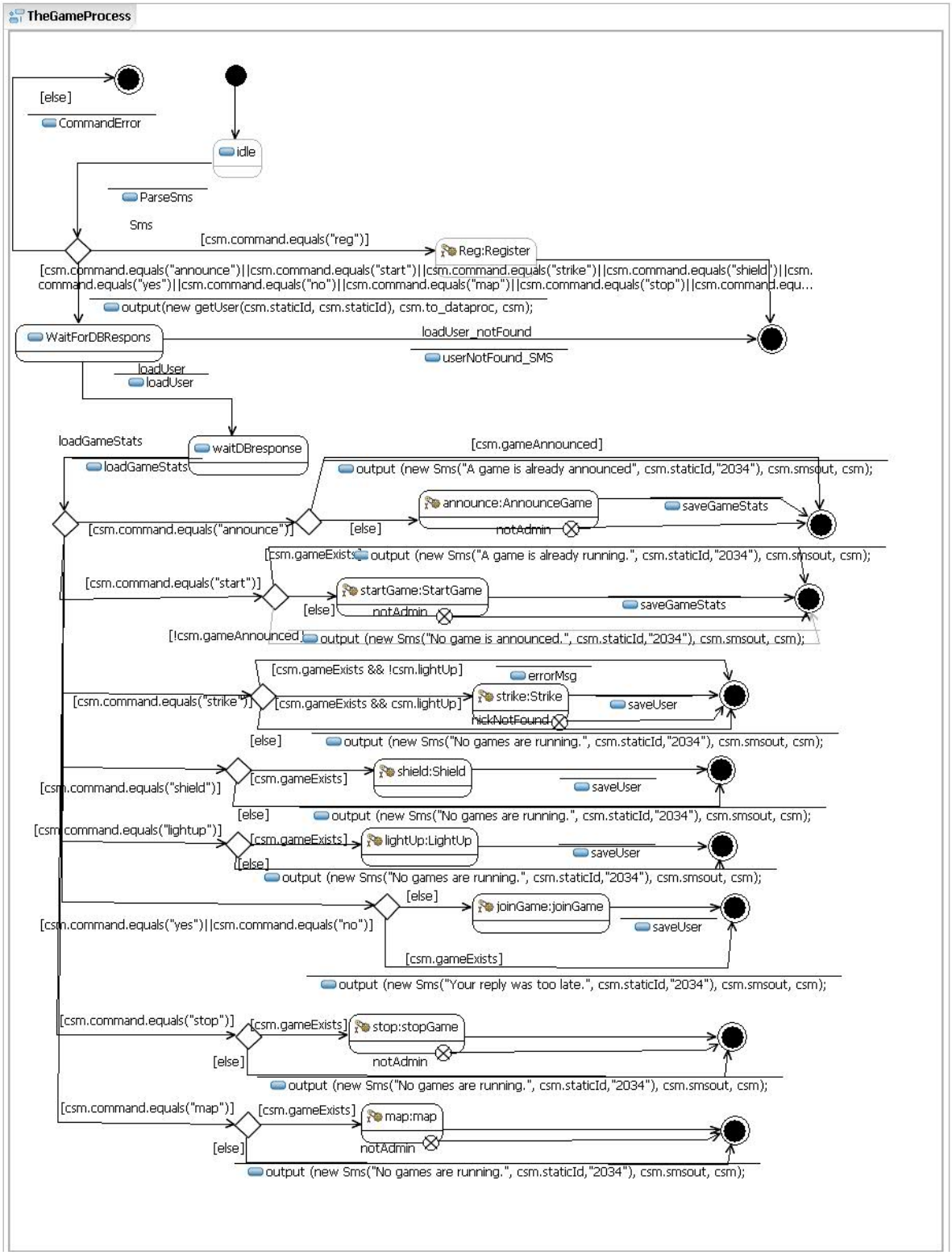


I denne tilstandsmaskinen sjekkes det om identiteten (”static id”) allerede er i bruk. Hvis så, går det en beskjed tilbake til mobiltelefonen. Ellers genereres det en ny sesjon med en tilhørende prosess, og meldingen fra mobiltelefonen sendes videre til denne.

6.2 TheGameProcess

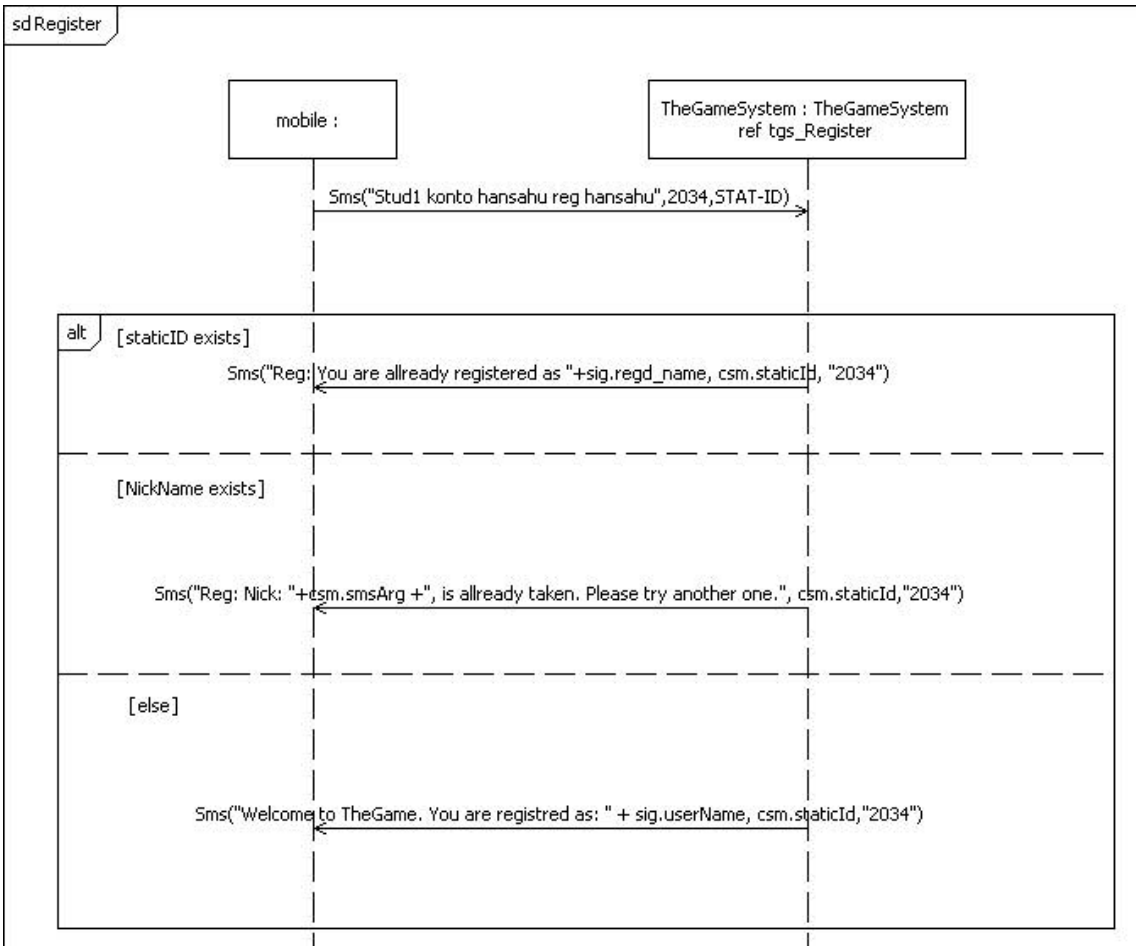


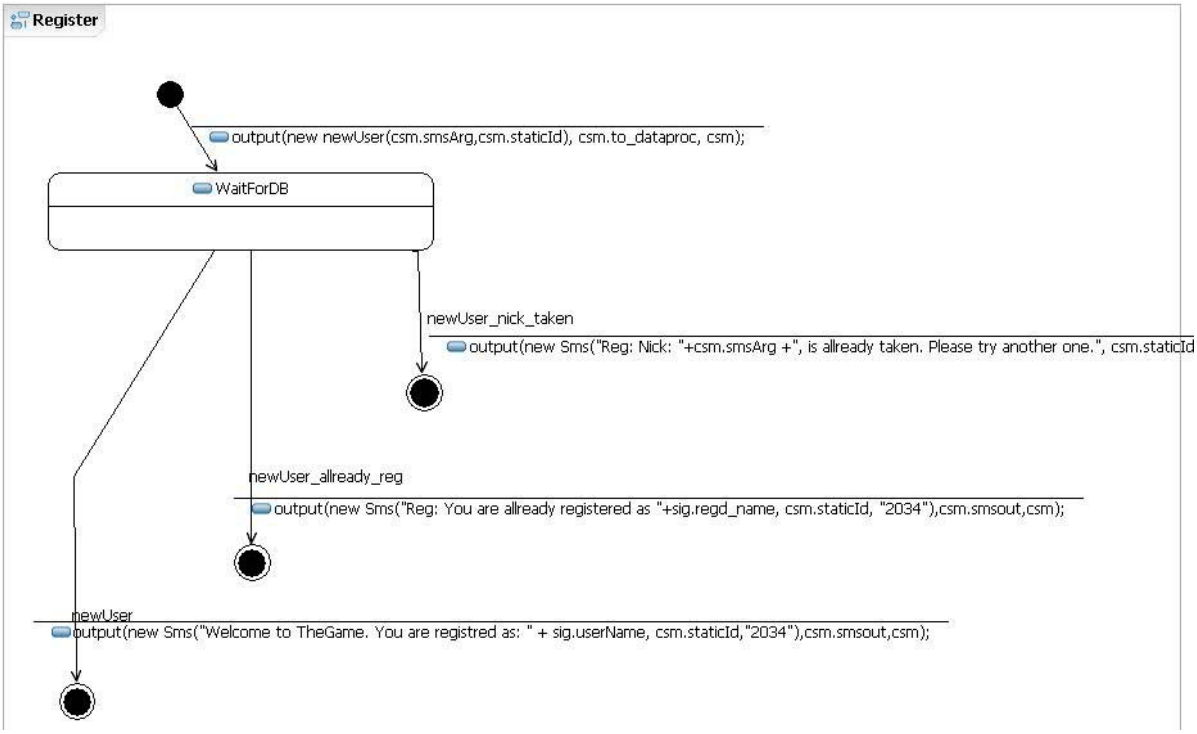

sd tgs_TheGameSystem
contr : TheGame...
dataproc : TheGa...
nto hansahu command hansahu",2034,STAT-ID)
«create» create(STAT-ID)
TheGameProc : T...
Sms("Stud1 konto hansahu command hansahu",2034,STAT-ID)
alt
[command == "reg"]
if
tgs_Register
[command=="announce"||command=="start"||command=="strike"||command=="shield"||command=="yes"||command=="no"||comman
getUser(csm.staticId, csm.staticId)
alt
[user not found]
loadUser_notFound(sig.staticId)
loadUser_notFound(sig.staticId)
registered. Send a SMS \'stud1 konto finnc reg <username>\'.", csm.staticId,"2034")
loadUser(user.userName, user.points, user.shield, user.lastxpos, user.lastypos, user.inGame, user.lightUp, sig.staticId)
user.userName, user.points, user.shield, user.lastxpos, user.lastypos, user.inGame, user.lightUp, sig.staticId)
getGameStats(sig.staticId)
loadGameStats(gamestat.gameExists, gamestat.gameAnnounced,sig.staticId)
loadGameStats(gamestat.gameExists, gamestat.gameAnnounced,sig.staticId)
alt
[command == "announce"]
ref
tgs_AnnounceGame
[command == "yes" || command == "no"]
ref
tgs_joinGame
[command == "start"]
ref
tgs_startGame
[command == "lightup"]
ref
tgs_lightUp
[command == "shield"]
ref
tgs_shield
[command == "strike"]
ref
tgs_strike
[command == "map"]
ref
tgs_map
[command == "stop"]
ref
tgs_stopGame
[else]
alt
[command == "shieldwrong"]
s("Usage: Stud1 konto finnc shield [1-400] [1-60]", csm.staticId,"2034")
[command == "strikewrong"]
sage: Stud1 konto finnc strike <username> [1-1000]", csm.staticId,"2034")
[command == "regError"]
Sms("UserName can not be empty", csm.staticId,"2034")
[else]
Sms("Unvalid command.", csm.staticId,"2034"),csm.smsout,csm)



TheGameProcess representerer sesjonen. Det opprettes en sesjon for hver henvendelse. Sesjonen terminerer når melding er sendt tilbake til brukeren. Maskinen starter med å sjekke hva slags type melding brukeren har sendt. Dersom brukerens kommando ikke er kjent av systemet, sendes det en feilmelding tilbake til brukeren.

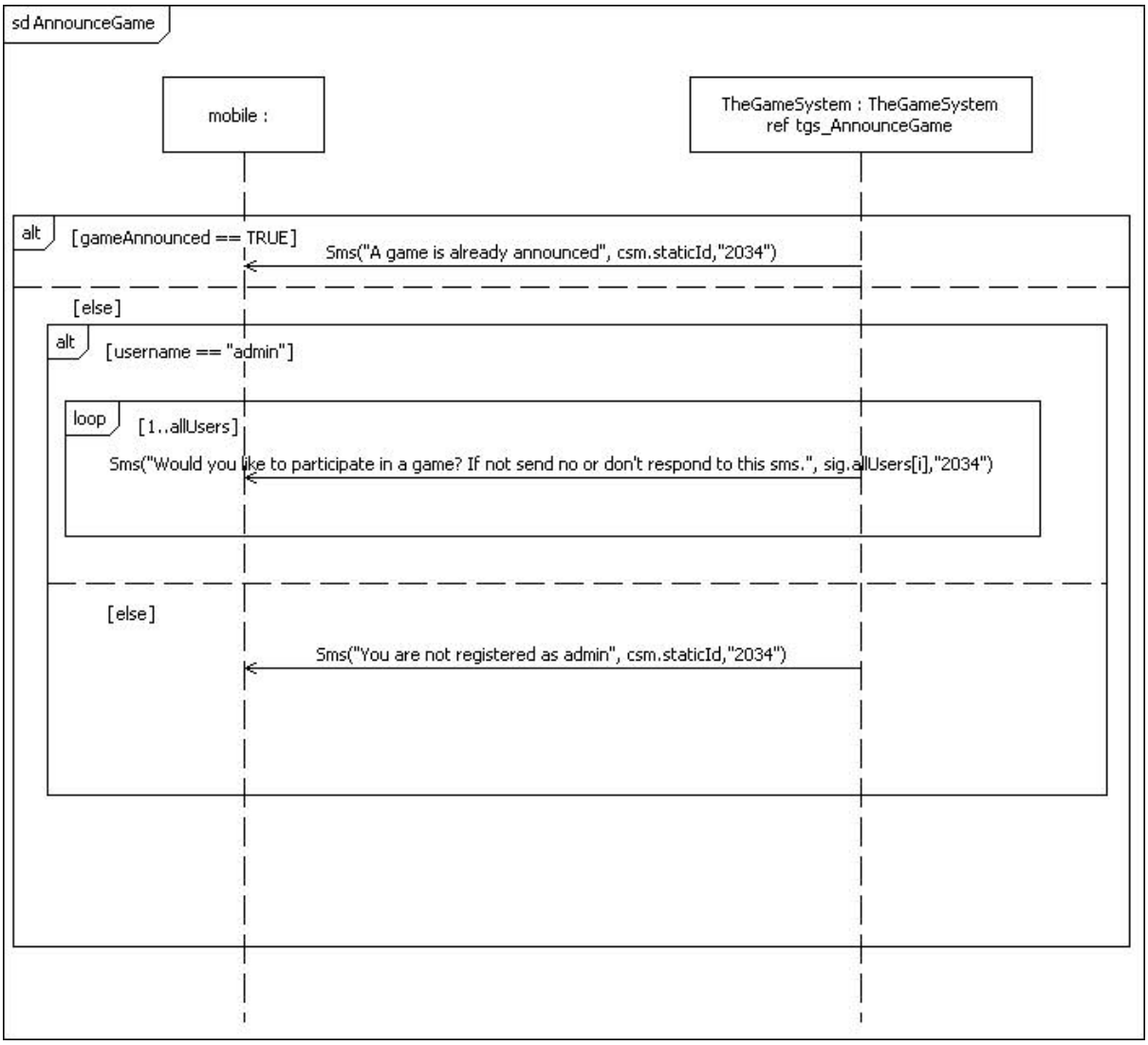
6.2.1 Register

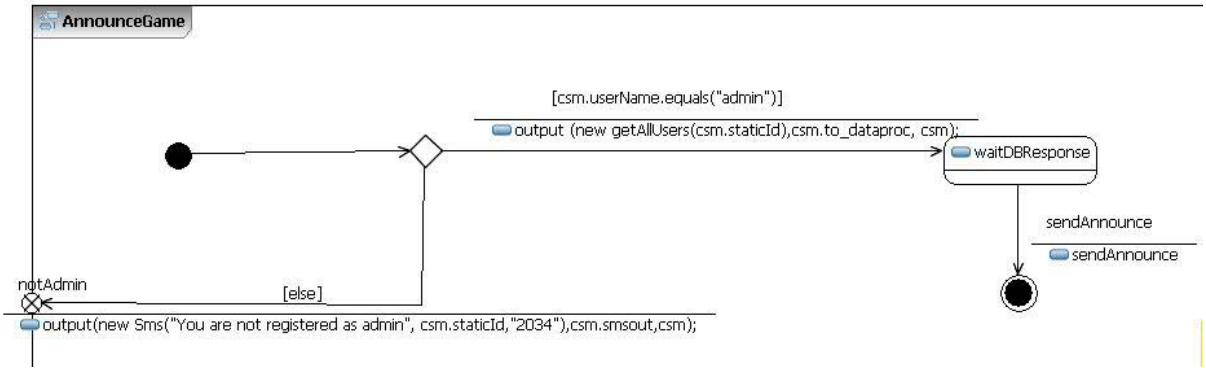




I denne subtilstandsmaskinen registreres nye brukere. Det gjøres først et oppslag i database. Dersom kallenavnet allerede er i bruk, sendes melding tilbake til brukeren om dette. Tilsvarende hvis brukeren allerede er registrert. Ellers så opprettes et nytt brukerobjekt, og det sendes en bekreftelse om dette tilbake til brukeren.

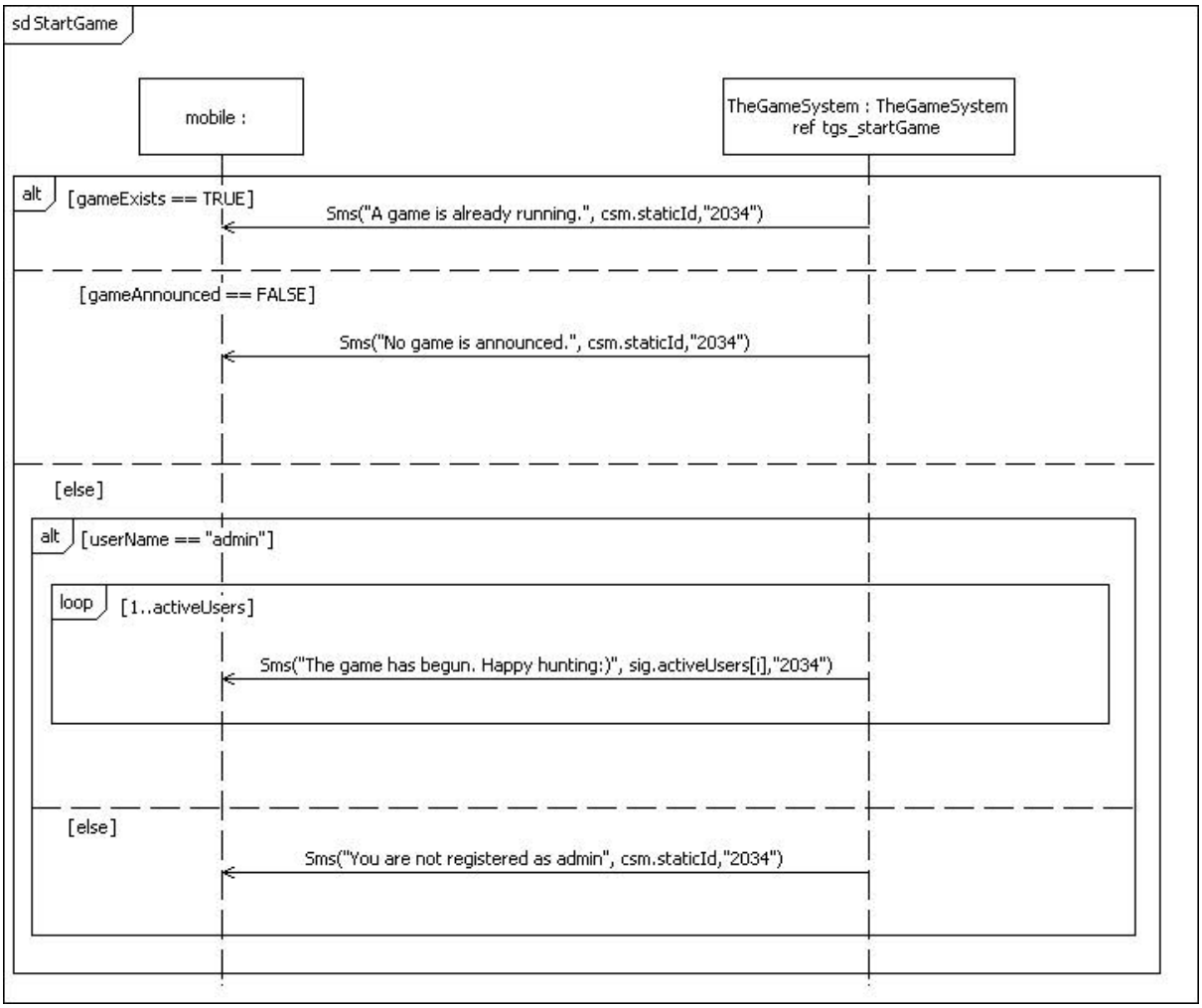
6.2.2 AnnounceGame

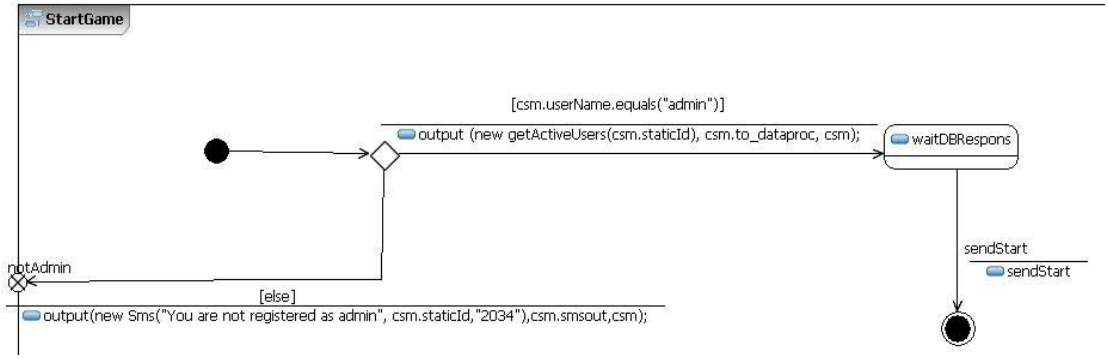




Hvis kommandoen er **announce**, og annonsering allerede er gjort, sendes det melding til brukeren (administrator) om dette. Ellers utføres annonsering, og det registreres i databasen at dette er gjort.

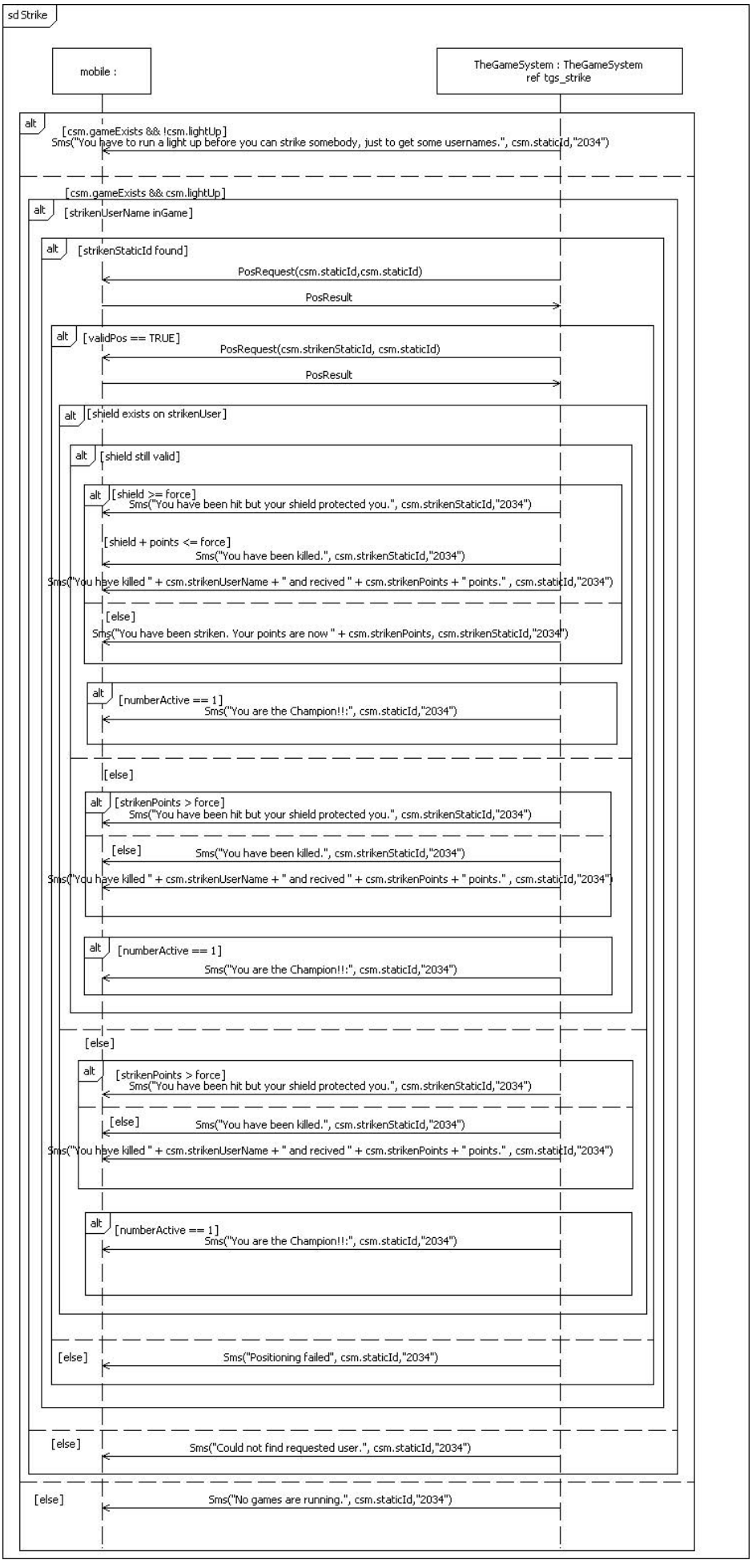
6.2.3 StartGame

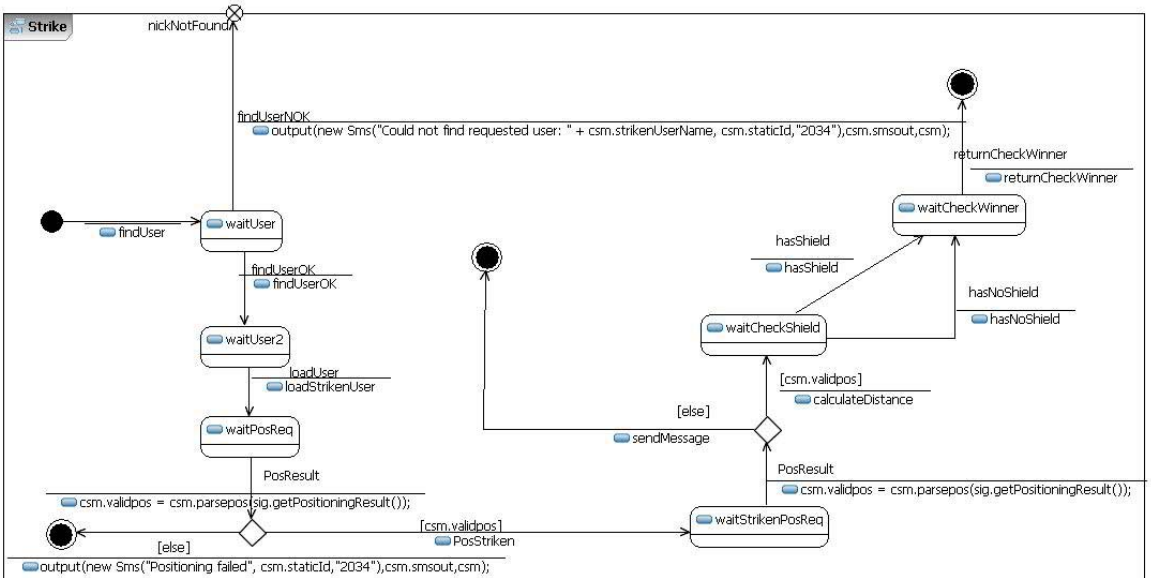




Hvis bruker ikke er administrator (brukernavn = ”admin”), sendes feilmelding til brukeren om dette. Ellers hentes det en liste fra arkivet over aktive brukere, og hver av dem mottar en SMS om at spillet er i gang.

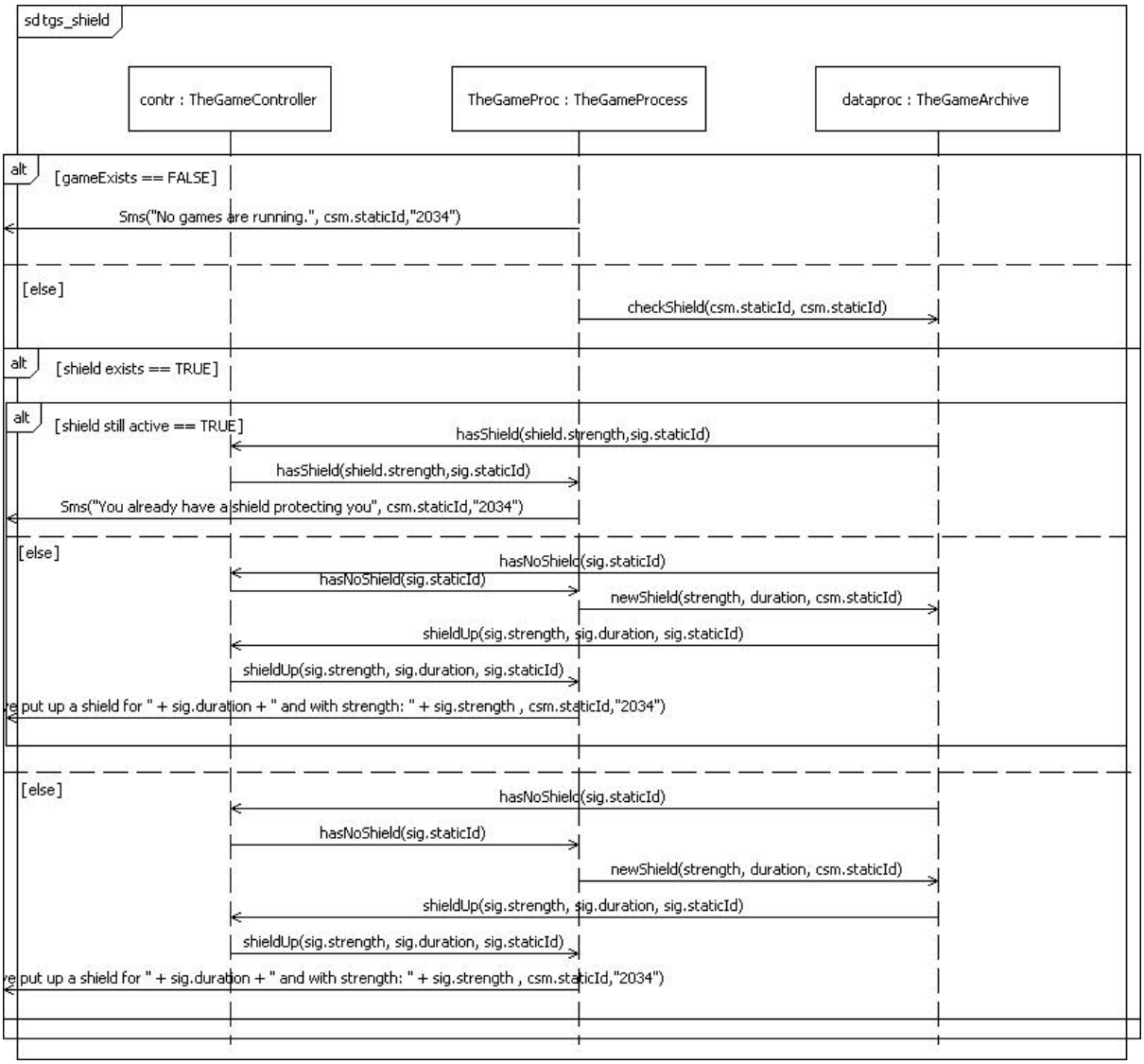
6.2.4 Strike

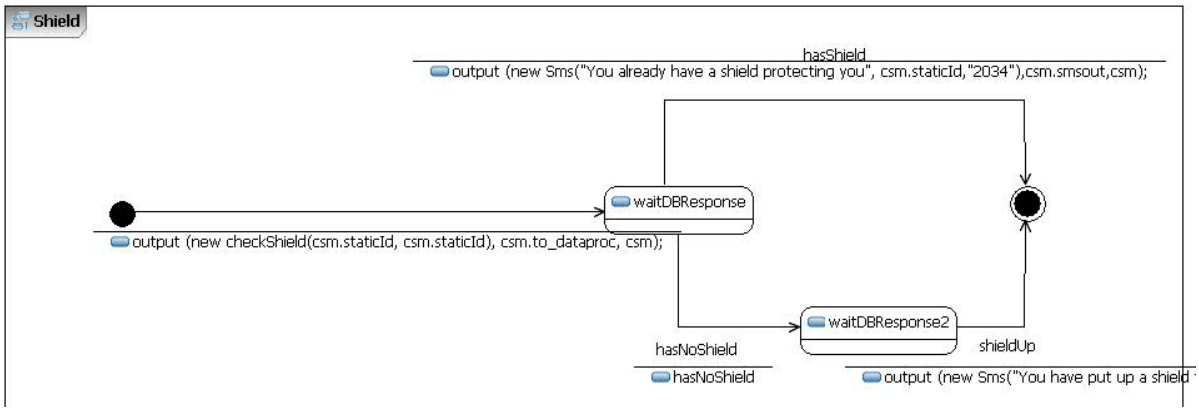

sd Strike
mobile :
TheGameSystem : TheGameSystem
ref tgs_strike
alt
[csm.gameExists && !csm.lightUp]
Sms("You have to run a light up before you can strike somebody, just to get some usernames.", csm.staticId,"2034")
[csm.gameExists && csm.lightUp]
alt
[strikenUserName inGame]
alt
[strikenStaticId found]
PosRequest(csm.staticId,csm.staticId)
PosResult
alt
[validPos == TRUE]
PosRequest(csm.strikenStaticId, csm.staticId)
PosResult
alt
[shield exists on strikenUser]
alt
[shield still valid]
alt
[shield >= force]
Sms("You have been hit but your shield protected you.", csm.strikenStaticId,"2034")
[shield + points <= force]
Sms("You have been killed.", csm.strikenStaticId,"2034")
Sms("You have killed " + csm.strikenUserName + " and recived " + csm.strikenPoints + " points." , csm.staticId,"2034")
[else]
Sms("You have been striken. Your points are now " + csm.strikenPoints, csm.strikenStaticId,"2034")
alt
[numberActive == 1]
Sms("You are the Champion!!:", csm.staticId,"2034")
[else]
alt
[strikenPoints > force]
Sms("You have been hit but your shield protected you.", csm.strikenStaticId,"2034")
[else]
Sms("You have been killed.", csm.strikenStaticId,"2034")
Sms("You have killed " + csm.strikenUserName + " and recived " + csm.strikenPoints + " points." , csm.staticId,"2034")
alt
[numberActive == 1]
Sms("You are the Champion!!:", csm.staticId,"2034")
[else]
alt
[strikenPoints > force]
Sms("You have been hit but your shield protected you.", csm.strikenStaticId,"2034")
[else]
Sms("You have been killed.", csm.strikenStaticId,"2034")
Sms("You have killed " + csm.strikenUserName + " and recived " + csm.strikenPoints + " points." , csm.staticId,"2034")
alt
[numberActive == 1]
Sms("You are the Champion!!:", csm.staticId,"2034")
[else]
Sms("Positioning failed", csm.staticId,"2034")
[else]
Sms("Could not find requested user.", csm.staticId,"2034")
[else]
Sms("No games are running.", csm.staticId,"2034")



Det gjøres forsøk på å identifisere brukeren. Dersom det ikke lykkes, sendes det en SMS om dette. Ellers så hentes brukerdata fra databasen, og det gjøres et forsøk på posisjonering. Mislykket posisjonering medfører SMS om dette til brukeren. Etter vellykket posisjonering kalkuleres avstanden. Så gjøres det en sjekk på om angrepet har slått ut motstanderen eller ikke. Motstanderen får beskjed om angrepet, og om han er slått ut eller hvor mange poeng han har tapt. Til slutt sjekkes det om angrepet har medført at angriperen har vunnet spillet. Hvis så, sendes det SMS til vedkommende om dette.

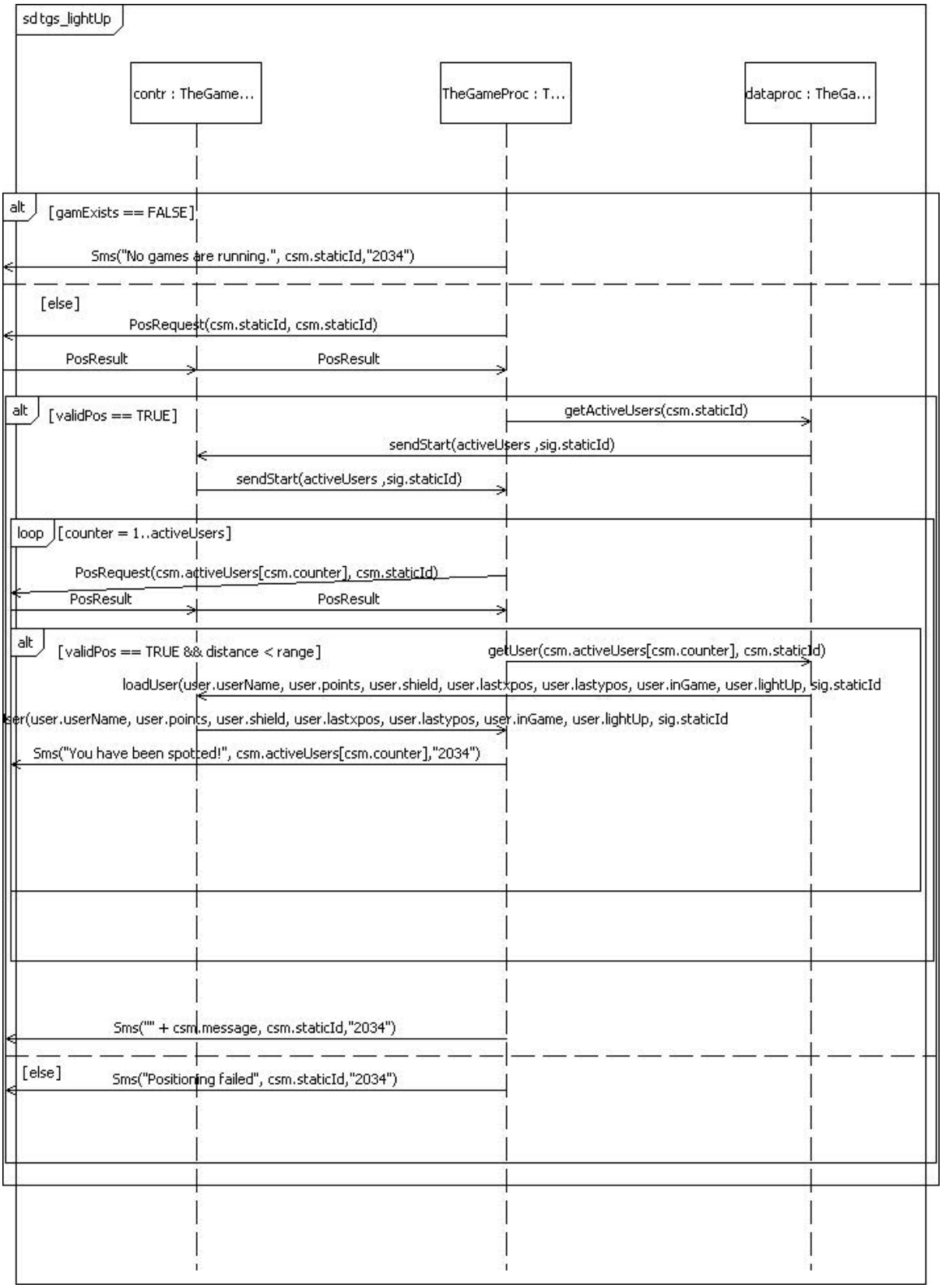
6.2.5 Shield

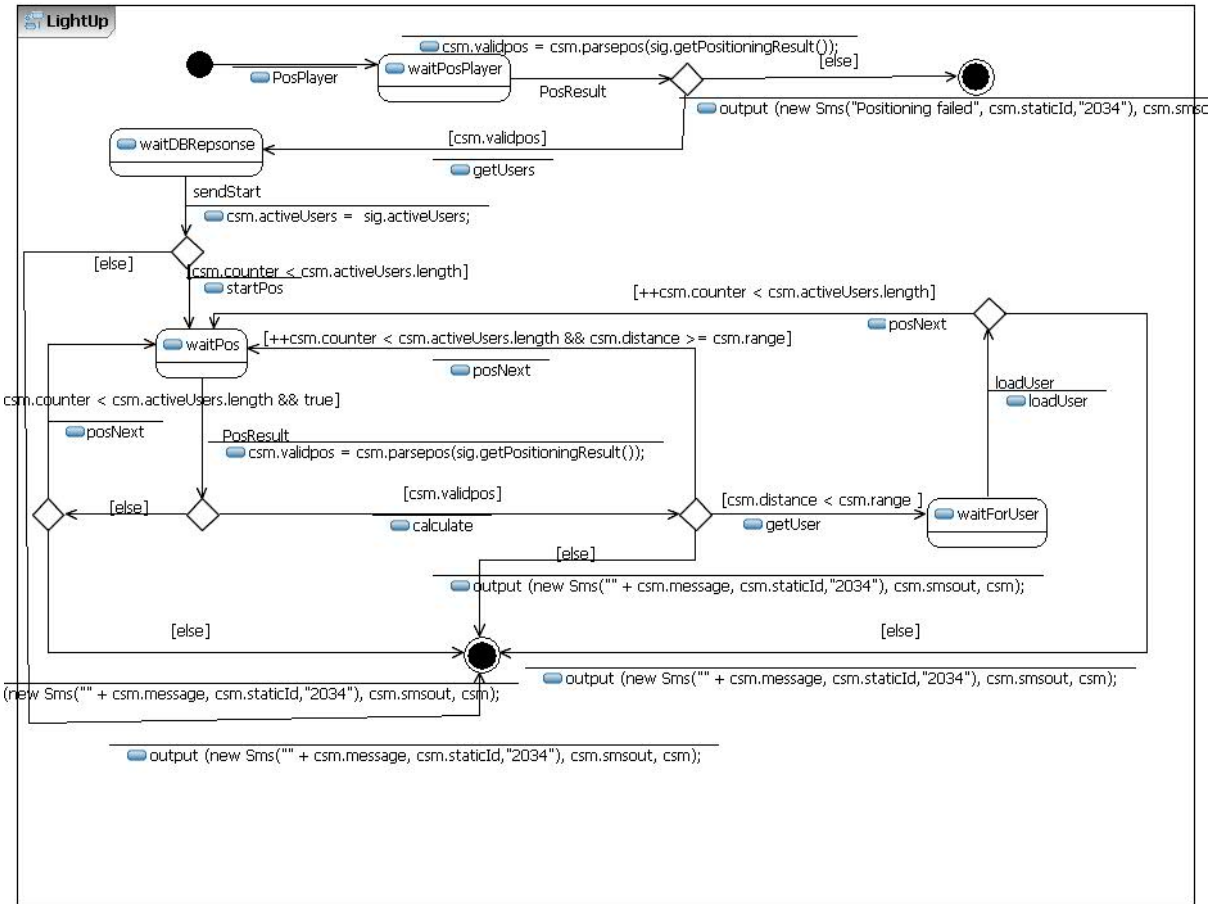




Det sjekkes i arkivet om brukeren har et skjold, og hvor sterkt dette eventuelt er. Hvis vedkommende ikke har et skjold, etableres dette, og brukeren får en bekreftelse på SMS.

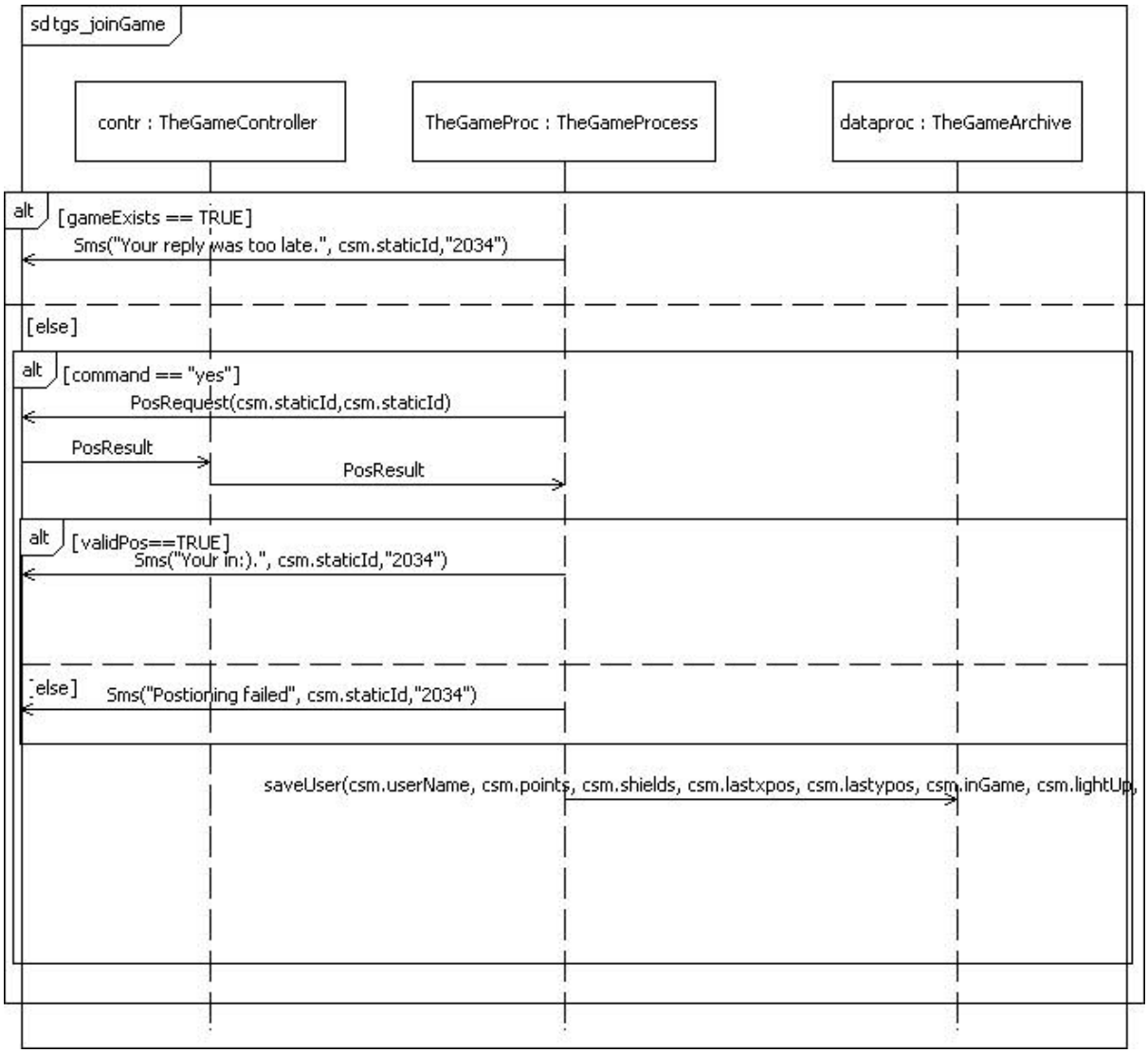
6.2.6 LightUp

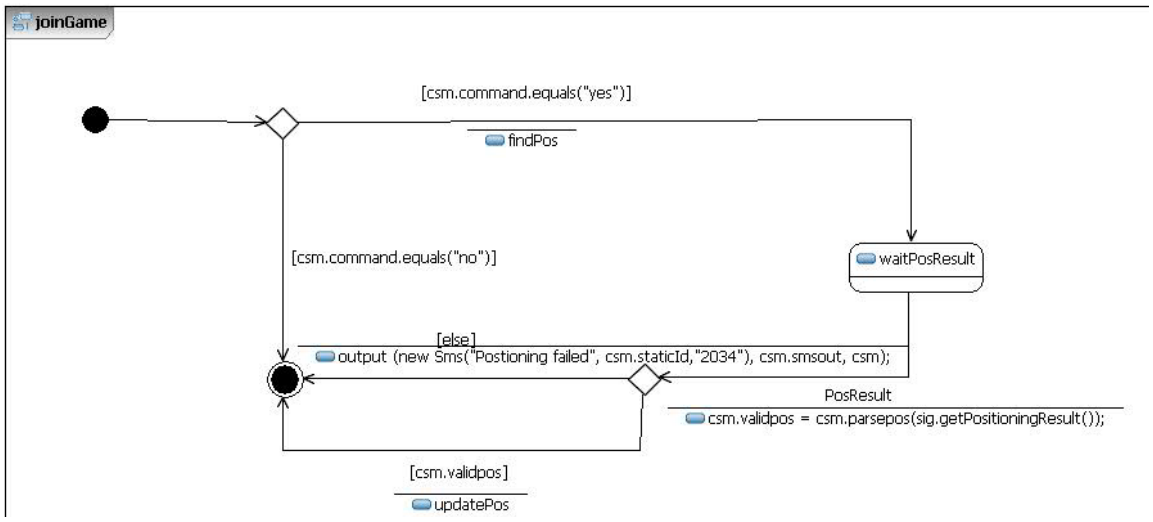

sd tgs_lightUp
contr : TheGame...
TheGameProc : T...
dataproc : TheGa...
alt
[gamExists == FALSE]
Sms("No games are running.", csm.staticId,"2034")
[else]
PosRequest(csm.staticId, csm.staticId)
PosResult
PosResult
alt
[validPos == TRUE]
getActiveUsers(csm.staticId)
sendStart(activeUsers ,sig.staticId)
sendStart(activeUsers ,sig.staticId)
loop
[counter = 1..activeUsers]
PosRequest(csm.activeUsers[csm.counter], csm.staticId)
PosResult
PosResult
alt
[validPos == TRUE && distance < range]
getUser(csm.activeUsers[csm.counter], csm.staticId)
loadUser(user.userName, user.points, user.shield, user.lastxpos, user.lastypos, user.inGame, user.lightUp, sig.staticId
ser(user.userName, user.points, user.shield, user.lastxpos, user.lastypos, user.inGame, user.lightUp, sig.staticId
Sms("You have been spotted!", csm.activeUsers[csm.counter],"2034")
Sms("" + csm.message, csm.staticId,"2034")
[else]
Sms("Positioning failed", csm.staticId,"2034")



Spilleren posisjoneres. Så posisjoneres alle aktive brukere, og det sjekkes om brukerne er innenfor det opplyste området. Brukere som er innenfor området vil få en melding om at de er sett. Brukeren som sendte lightUp vil få en liste over disse spillerne og poengene deres.

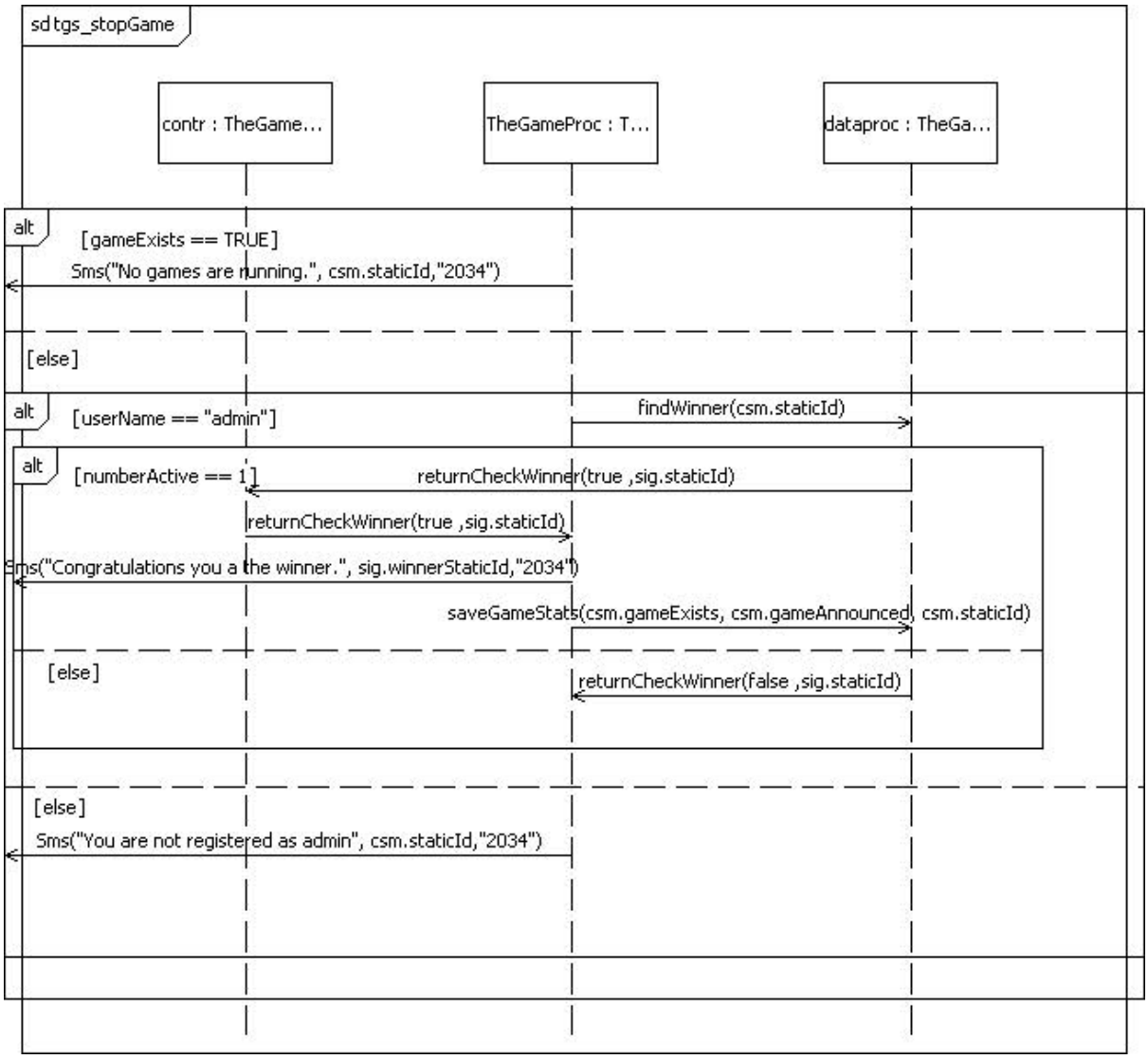
6.2.7 joinGame

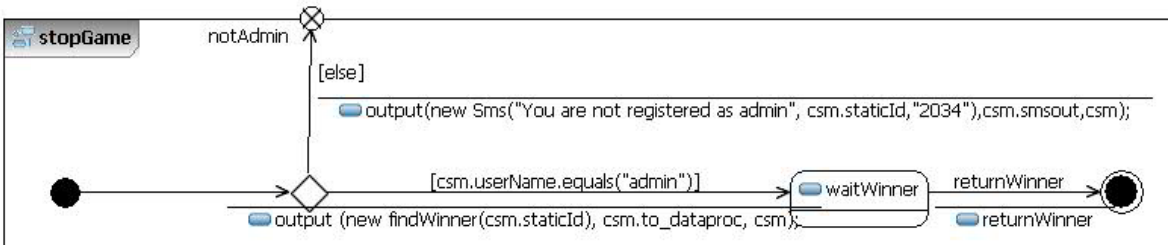




Spilleren har svart på annonsering av et nytt spill. Dersom spiller ønsker å delta, gjøres det en posisjonsbestemmelse. Dersom denne er vellykket, lagres denne i databasen. Hvis ikke, sendes det en SMS tilbake til brukeren som informerer om manglende posisjonering.

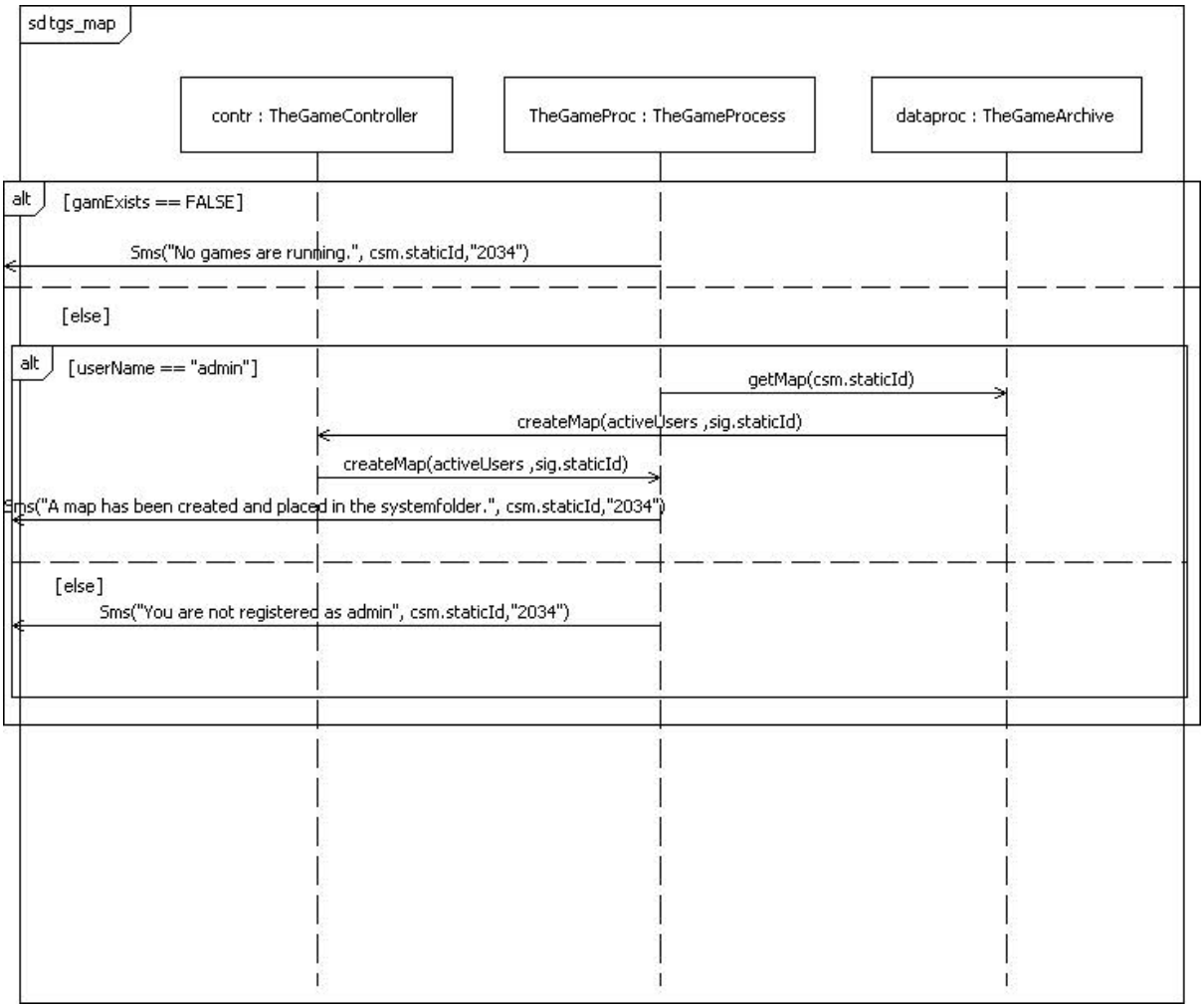
6.2.8 stopGame

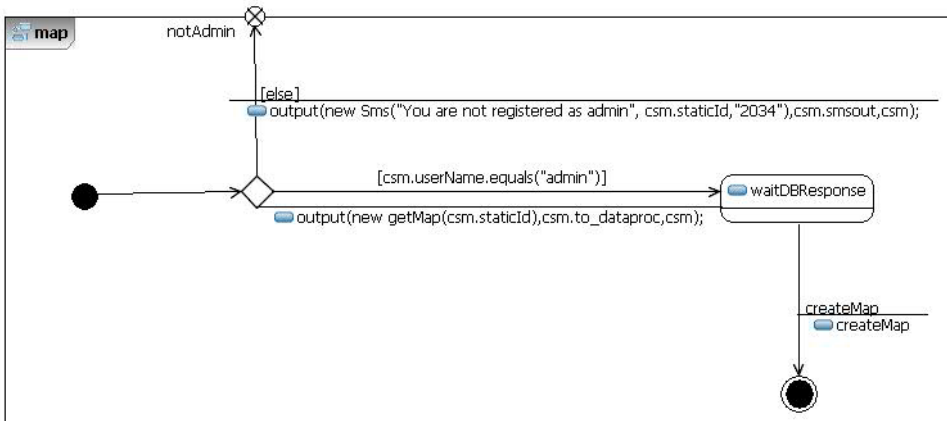




Dersom brukeren ikke er administrator, sendes feilmelding til brukeren om dette. Deretter identifiseres vinneren, som så får tilsendt en SMS med gratulasjoner.

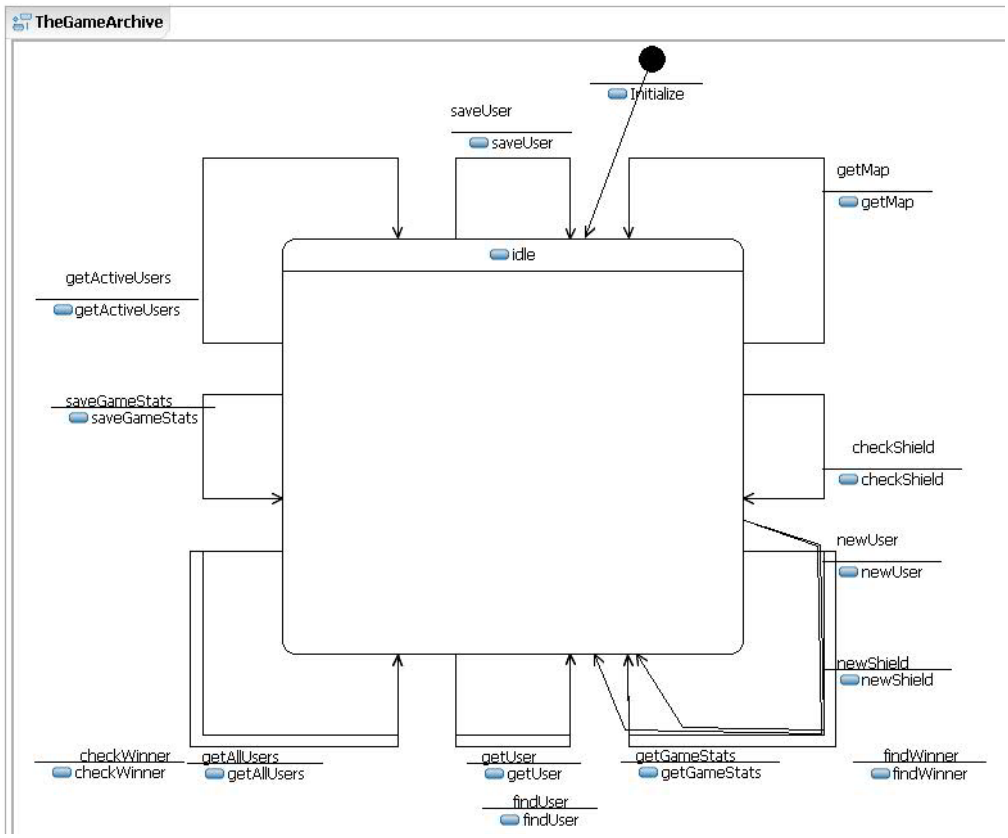
6.2.9 Map





Dette er en administratorfunksjon som genererer kartdata for alle aktive brukere.

6.3 Archive



Systemet har én instans av **Archive**. Arkivet holder rede på brukerdata, inklusive *shields*. Normalt ville arkivet benyttet en database for persistens, men i vårt system har vi valgt å kun benytte oss av *hashmaps* for unngå kompleksitet.

6.3.1 checkShield

Det sjekkes om brukeren har et *skjold*. Sender melding **hasNoShield** eller **hasShield**.

6.3.2 checkWinner ???

Iterer gjennom listen av brukere som deltar i spillet, og sender meldingen **returnCheckWinner**

6.3.3 findWinner

Identifiserer vinneren av spillet.

6.3.4 findUser

Finner bruker med gitt brukernavn. Sender meldingene **findUserOK** eller **findUserNOK**.

6.3.5 getActiveUsers

Itererer gjennom mengden brukere. Sender meldingen **sendStart** med en liste av aktive brukere.

6.3.6 getAllUsers

Sender meldingen **sendAnnounce** med en liste av alle brukere.

6.3.7 getGameStats

Sender meldingen **loadGameStats** med verdier for hvorvidt spillet eksisterer og om det er annonsert.

6.3.8 getMap

Genererer en liste av aktive brukere med tilhørende informasjon om poeng, skjold og siste kjente posisjon og returnerer denne i **createMap**.

6.3.9 getUser

Dersom brukeren finnes, returneres **loadUser** med brukernavn, poeng, skjold, siste posisjon, *aktivstatus*, og status *lightUp*. Hvis brukeren ikke finnes, sendes **loadUser_notFound**.

6.3.10 newShield

Et nytt **Shield** instansieres, og det settes utløpstidspunkt, styrke, og *staticId*. **Shield** legges til den tilstandsmaskinens liste av skjold. Så sendes **shieldUp** med styrke, varighet og *staticId*.

6.3.11 newUser

Sjekker om staticId allerede finnes. Hvis så, sendes **newUser_already_reg** med brukernavn. Hvis staticId ikke finnes, sjekkes det om det finnes en bruker med gitt brukernavn. Hvis så, sendes **newUser_nickTaken**. Ellers så initieres en ny **User** med defaultverdier for poeng og skjold. Denne legges til listen av brukere. Så sendes **newUser** som fanges opp av Controller.

6.3.12 saveGameStats

Lagrer en **GameStat** med status for anonnsert og aktivt spill.

6.3.13 saveUser

Lagrer brukerdata.

7. Sikkerhetsanalyse

I sikkerhetsanalysen har vi brukt CORAS-metoden, og illustrasjonene er gjort med CORAS' diagram-editor. I utgangspunktet har vi fulgt den fremgangsmåten som er beskrevet i Model-based security analysis in seven step (den Braber, et al. 2007).

7.1 Trinn 1-3: Avklaringer

CORAS-metoden beskriver syv trinn. De tre første er imidlertid innledende manøvre for å få etablert kontekst og felles utgangspunkt for arbeidet. Vi har valgt å behandle de tre første trinnene som ett.

7.1.1 Mål

Oppdraget er formulert som:

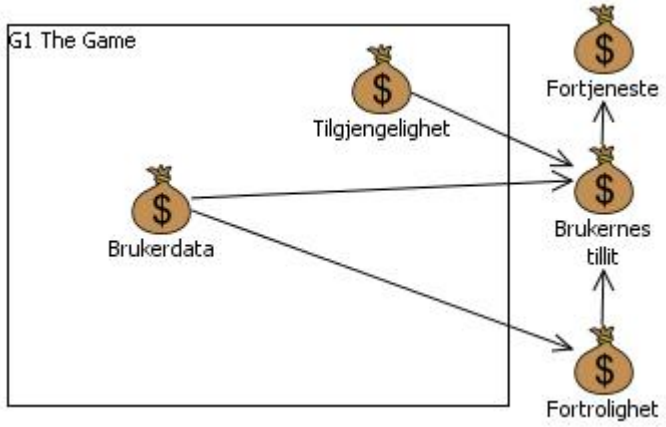
> *Assume that the players buy their points from the game administrator in real money (1 EURO per point + 20%). Assume also that the winner of the game may cash-in the points in real money (1 EURO per point). Conduct a security risk analysis on behalf of the game administrator according to the seven steps of the BT Technology Journal article.*

Oppdraget vektlegger betaling og fortjeneste. Systemet vårt implementerer ingen regnskapsrutiner. Vi velger derfor å tolke oppdraget slik at det er bruken av systemet (representert ved brukernes tillit) som er forutsetning for fortjeneste. Videre ser vi på korrekte brukerdata som forutsetning for poengregnskap som igjen kan generere utbetalinger til de enkelte brukerne. For å begrense kompleksiteten til analysen og diagrammer utelater vi direkte referanser til bruk og regnskap.

Det er en rekke sårbarheter og verdier som vi ville ha behandlet i en virkelig analyse, men oppdragsgiver har anmodet oss om å begrense oss til de elementer som impliserer fire risikoer som ikke møter evalueringskriteriene. Vi ser blant annet ikke på interne menneskelige trusler eller systemfeil knyttet til overbelastning (dårlig skalering).

7.1.2 Aktiva

Vi har identifisert fem aktiva som tilhører klienten (administrator):



Aktivum	**Viktighet**	**Type**
Tilgjengelighet	1	Direkte
Brukerdata	1	Direkte
Brukernes tillit til systemet	1	Indirekte
Fortrolighet	2	Indirekte
Fortjeneste	3	Indirekte

7.1.3 Trusler og sårbarheter

Oversikt over trusler og sårbarheter:

Hvem/hva	Hvordan	Hvorfor
Systemfeil	Systemet går ned	Feil/mangler ved systemet
PATS	Ustabil forbindelse	Nettverksfeil, feil/mangler ved systemet
Hacker	Bryter seg inn i systemet og kompromitterer brukerdata	Dårlig sikkerhet

7.1.4 Sannsynlighet/frekvens

Vi har prøvd å komme frem til en sannsynlighetstabell som er gyldig for alle uønskede hendelser. Dette gjør det mulig å finne kombinerte sannsynligheter.

Sannsynlighetsverdi	Beskrivelse
Sikker	>1000 ganger pr. år
Sannsynlig	100-999 ganger pr. år
Mulig	10-99 ganger pr. år
Sjelden	1-9 ganger pr. år
Aldri	0 ganger pr. år

7.1.5 Tilgjengelighet – konsekvenser og risikoevaluering

Konsekvenstabell for tilgjenglighet

Konsekvensverdi	Beskrivelse
Ubetydelig	<1 minutter responstid
Moderat	1-3 minutter responstid
Alvorlig	4-10 minutter responstid
Katastrofal	>10 minutter responstid

Risikoevalueringsmatrise for tilgjenglighet

		Konsekvens			
		Ubetydelig	**Moderat**	**Alvorlig**	**Katastrofal**
Sannsynlighet	**Aldri**	Akseptabel	Akseptabel	Akseptabel	Akseptabel
	Sjelden	Akseptabel	Akseptabel	Akseptabel	Må undersøkes
	Mulig	Akseptabel	Akseptabel	Må undersøkes	Må undersøkes
	Sannsynlig	Akseptabel	Må undersøkes	Må undersøkes	Må undersøkes
	Sikker	Må undersøkes	Må undersøkes	Må undersøkes	Må undersøkes

7.1.6 Brukerdata - konsekvenser og risikoevaluering

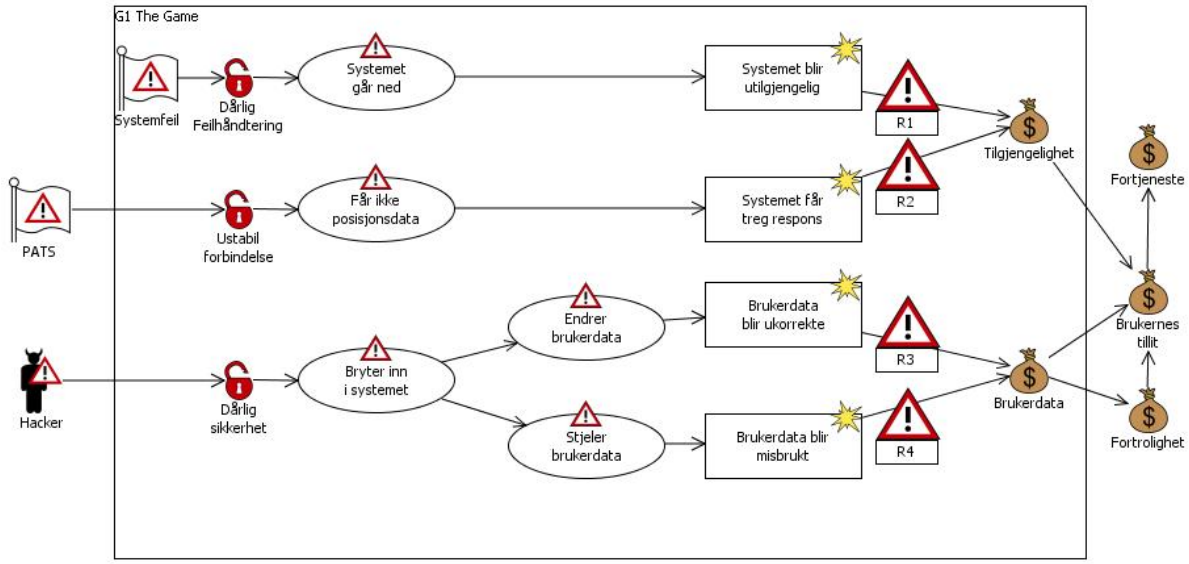
Konsekvenstabell for Brukerdata

Konsekvensverdi	Beskrivelse
Moderat	< 5 brukere berørt
Alvorlig	6- 50 brukere berørt
Katastrofal	Flere enn 50 brukere berørt

Risikoevalueringsmatrise for Brukerdata

		Konsekvens			
		Ubetydelig	**Moderat**	**Alvorlig**	**Katastrofal**
Sannsynlighet	**Aldri**	Akseptabel	Akseptabel	Akseptabel	Akseptabel
	Sjelden	Akseptabel	Akseptabel	Akseptabel	Må undersøkes
	Mulig	Akseptabel	Akseptabel	Må undersøkes	Må undersøkes
	Sannsynlig	Akseptabel	Må undersøkes	Må undersøkes	Må undersøkes
	Sikker	Må undersøkes	Må undersøkes	Må undersøkes	Må undersøkes

7.2 Trinn 4: Risikoidentifisering



Vi har avgrenset oss til å identifisere fire risikoer, to for hvert aktivum. Risikoene er navnet R1 … R4.

R1 er risikoen som oppstår som følge av en systemfeil som ikke blir fanget opp. Feilen vil kunne medføre at systemet går ned, og dermed blir utilgjengelig for brukerne. Dette går utover systemets generelle tilgjengelighet, og vil dermed kunne true brukerne tillit til systemet.

R2 er risikoen for at en ikke får nødvendig posisjonsdata fra PATS. Systemet vil bruke tid på å vente på data (som ikke kommer), og systemet blir dermed tregt. Dette går ut over tilgjengeligheten som brukerne.

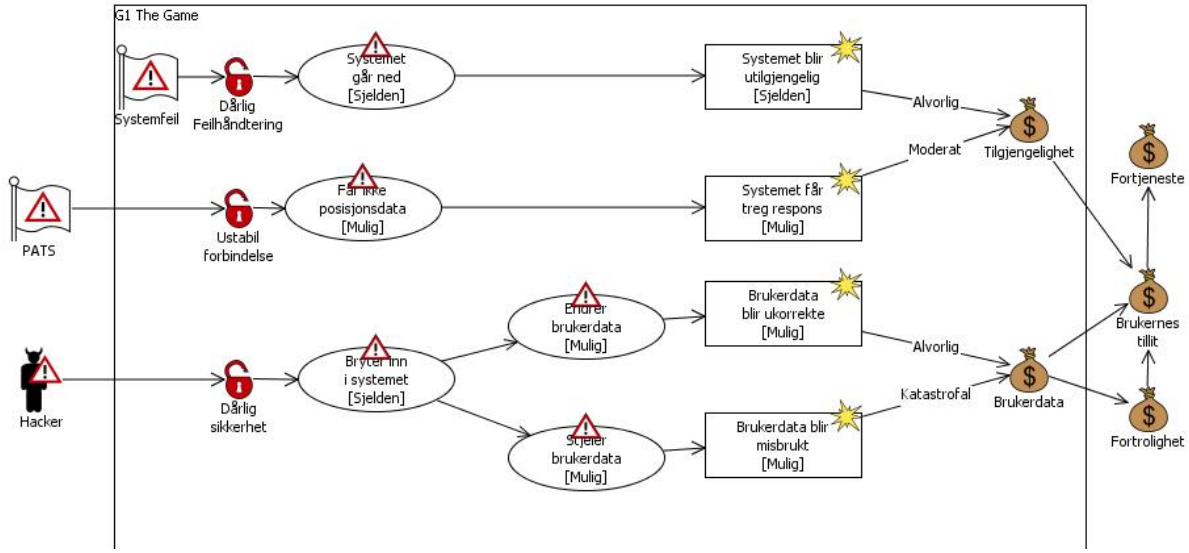
R3 oppstår som følge av en hacker som utnytter en sikkerhetssvakhet for å skaffe seg tilgang til systemet, som så blir brukt for å manipulere brukerdataene. Disse blir kompromittert, noe som påvirker riktigheten til brukerdataene. Dette truer konfidensialiteten til brukerdataene, og truer vil kunne redusere brukernes tillit til systemet.

R4 er også en hacker som bryter seg inn i systemet. Denne gangen stjeles brukerdata som så blir misbrukt.

7.3 Trinn 5: Risikoestimering

I dette trinnet prøvde vi å bruke vår erfaring til å estimere sannsynligheter for alle trusselscenarioer og deretter alle uønskede hendelser. Vi har ikke identifisert noen hendelse som kan være resultater av kombinerte scenarioer. Dermed var det ikke noe behov for å kalkulere kombinerte sannsynligheter.

Vi anga også konsekvensen for de uønskede hendelsene for de berørte aktiva.



7.4 Trinn 6: Risikoevaluering

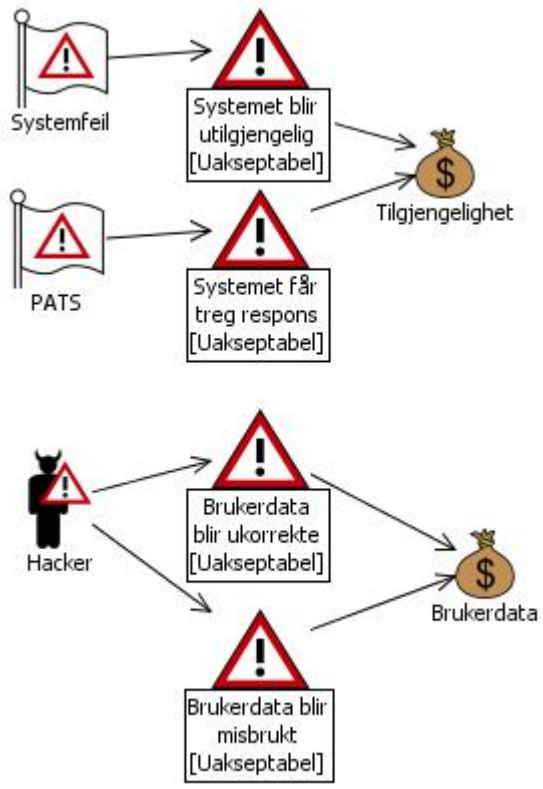
I dette trinnet kombinerte vi sannsynligheten for en hendelse med konsekvensen den har for det aktuelle aktivum. De fire hendelsene er

- Systemet blir utilgjengelig (SU)
- Systemet får treg respons (ST)
- Brukerdata blir ukorrekte (BU)
- Brukerdata blir misbruk (BM)

Hendelsene ble så plassert i evalueringsmatrisen i forhold til sannsynlighet og konsekvens.

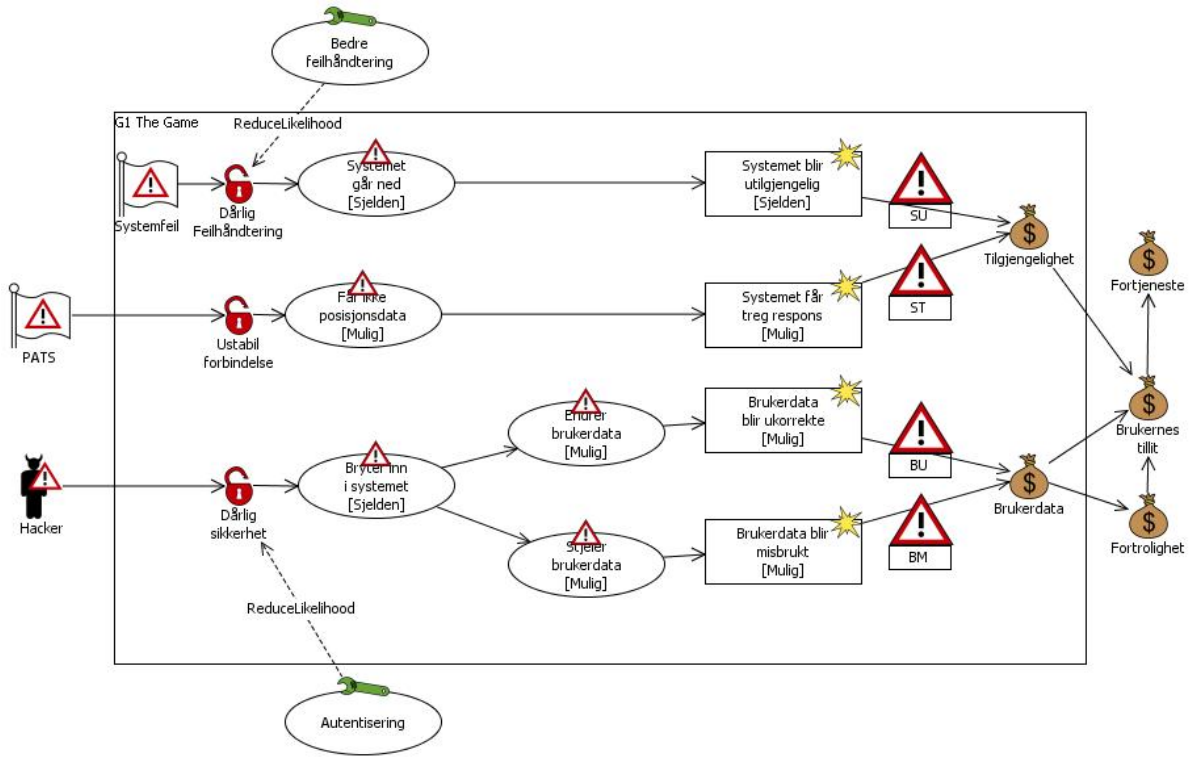
		Konsekvens			
		Ubetydelig	**Moderat**	**Alvorlig**	**Katastrofal**
Sannsynlighet	**Aldri**				
	Sjelden				SU
	Mulig			ST , BU	BM
	Sannsynlig				
	Sikker				

I denne prosessen ble vi nødt til å gå noen runder på sannsynligheter og konsekvenser for å få frem et risikobilde vi syntes var korrekt etter vår oppfatning. Vi kom da frem til følgende risikodiagram:



7.5 Trinn 7: Risikobehandling

Vi har identifisert fire uakseptable risikoer. Vi har forslag til behandling av tre av dem.



- Systemet blir utilgjengelig: Denne risikoen utnytter dårlig feilhåndtering. Sårbarheten kan behandles ved å implementere bedre feilhåndtering.
- Systemet får treg respons: Dette skjer som følge av sårbarheten vedrørende forbindelsen til PATS. Vi kan ikke se at det er noen enkel sak å selv behandle dette. Vi tror at det er PATS selv som må sikre at forbindelsen/leveransen av data er så stabil som mulig. *Vi outsourcer problemet.*
- Brukerdata blir ukorrekte/misbrukt: Systemet har i dag en svært begrenset form for autentisering. Autentisiteten er kun gitt av identiteten som ligger i telefonen (SIM-kortet). En kan tenke seg å etablere en egen mekanisme for autentisering som for eksempel støtter bruk av passord.