Radboud Universiteit Nijmegen

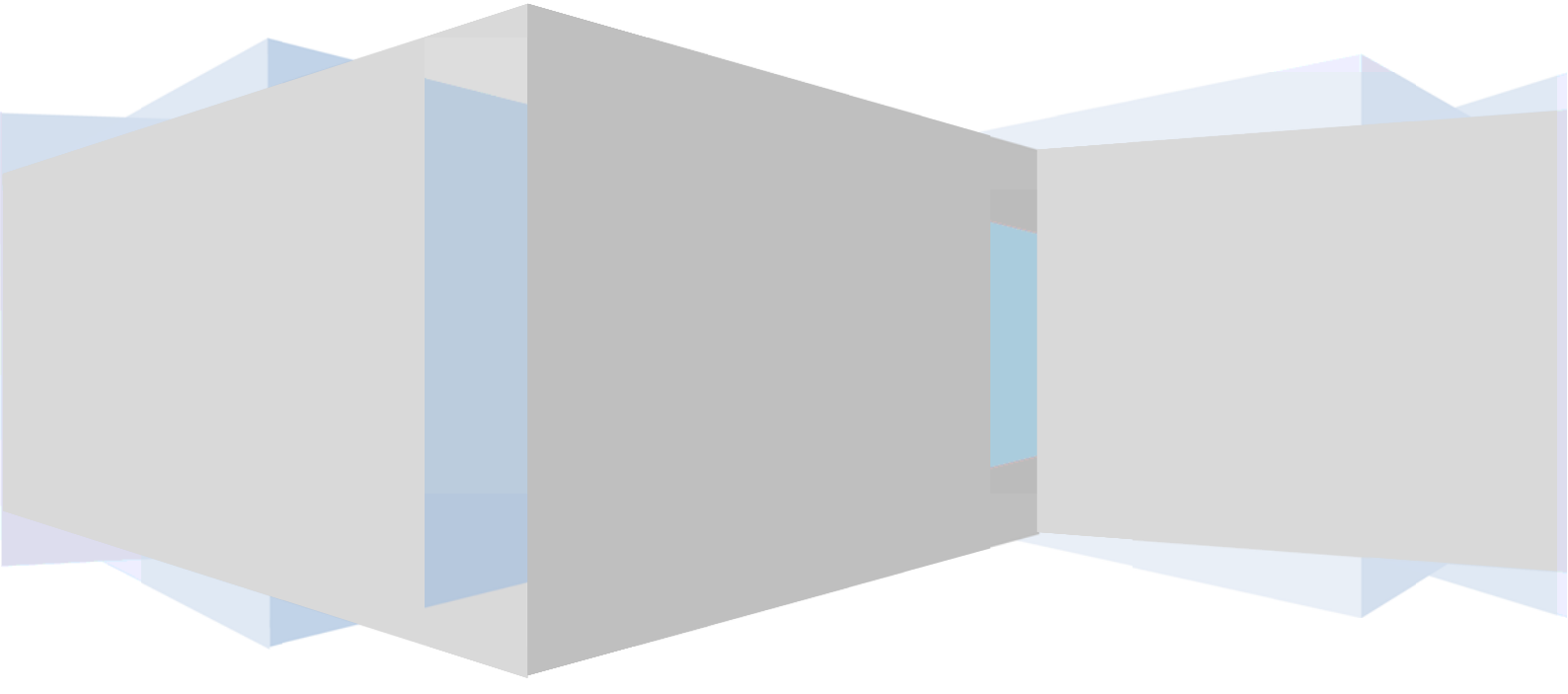
Serious Gaming in Requirements Engineering

Van spel tot Use Case

Jeffrey Kwee, Informatiekunde

jeffrey@zoja.nl

Begeleid door dr. S.J.B.A. (Stijn) Hoppenbrouwers



Abstract

Serious Gaming is een relatief nieuw begrip, al bestaat het concept al honderden jaren; een spel waarin het voornaamste doel niet vermaak is, maar functionaliteit of kennisoverdracht. Serious Gaming heeft veel voordelen, waaronder serieuze onderwerpen op speelse wijze aan een gebruiker brengen en die gebruiker motiveren.

Requirements Engineering, het proces van het vaststellen van formele eisen aan een informatiesysteem in ontwerp en alle activiteiten die daarmee te maken hebben, is een veelgebruikt proces binnen Systeem- en Software Engineering. Bij het proces van Requirements Engineering gaat veel tijd verloren en kampt men vaak met ongemotiveerde deelnemers aan het proces.

In dit artikel worden theoretische kaders gevormd omtrent het gebruik van Serious Gaming als middel om Requirements Engineering te vergemakkelijken. Dit theoretisch kader leidt tot een testapplicatie in een proof of concept-setting; de principes worden getest op werking en functionaliteit. De uitkomsten van de experimenten met de testapplicatie zijn positief; Serious Gaming als middel om Requirements Engineering te vergemakkelijken werkt.

Voorwoord

Deze scriptie is geschreven in het kader van de Bachelorscriptie van de opleiding Informatiekunde aan de Radboud Universiteit Nijmegen door Jeffrey Kwee.

De scriptie is tot stand gekomen onder begeleiding van Stijn Hoppenbrouwers. Hierbij wil ik Stijn graag bedanken voor zijn constructieve kritiek, zijn verfrissende ideeën omtrent het onderwerp en het engelengeduld dat hij toonde als ik weer een nieuw idee wilde uitwerken.

Ook gaat mijn dank uit naar de Bachelorscriptiecoördinator Engelbert Hubbers, zonder wiens goedkeuring deze scriptie uiteraard niet had bestaan.

Als laatste gaat mijn dank uit naar Douwe Egberts, Phillips, Illy en Peeze, zonder hen was ik niet in staat geweest om nachtenlang aan de scriptie te werken, mijn familie die me volstopte met lekker eten, mijn vrienden die me volstopten met lekker drinken, en aan Stéfanie, mijn persoonlijke muze die me steeds weer motiveerde om door te gaan.

Jeffrey A. Kwee,
22 Juni 2010

Informatiekundestudent
#0313270
Cohort 2003

Inhoudsopgave

Hoofdstuk 1: Inleiding

"The computer was born to solve problems that did not exist before."
- Bill Gates

Inleiding

We leven in de 21e eeuw, in een informatiemaatschappij die tot elk aspect van ons leven doordringt. Informatie wordt overal om ons heen uitgewisseld en informatie-uitwisseling maakt een steeds groter deel uit van ons leven. Een gevolg hiervan is dat men in het dagelijks leven steeds meer en meer in contact komt met informatiesystemen.

Het Internet verbindt mensen over de digitale snelweg die duizenden kilometers van elkaar verwijderd zijn, de huisartsenpost heeft een informatiesysteem waarmee symptomen van ziekten kunnen worden nagetrokken, autofabrikanten experimenteren met drive-by-wire, waarbij een computersysteem de taak van een autobestuurder overneemt.

Deze voorbeelden maken duidelijk dat we steeds afhankelijker worden van informatiesystemen, en dat het belangrijk is dat deze goed ontworpen zijn en goed werken om ons tot dienst te kunnen zijn.

Bouw het *juiste* systeem

Vaak is dat echter niet het geval. Een veel voorkomend geval is dat van een systeemontwikkelaar die een product aflevert na maandenlange ontwikkeling, dat bij oplevering ineens blijkt te stuiten op weerstand van de klant. De reden ervoor: het product werkt niet precies zoals de klant het had gewild. De systeemontwikkelaar had van tevoren interviews gehouden met belanghebbenden en op basis van die interviews een ontwerp gemaakt, dat geïmplementeerd, getest en opgeleverd. Het systeem ziet er goed uit en biedt veel functionaliteit. Waar is er dan iets mis gegaan?

Systemen die niet voldoen aan eisen van de klant na oplevering hebben vaak meerdere oorzaken als grondslag voor het falen. Een beperkt budget, deadlines die niet gehaald kunnen worden, een lijst van eisen die niet binnen dat budget en met die tijdslimieten kunnen worden gerealiseerd en het belangrijkste punt: de *juiste* eisen voor de oplossing van het probleem ontbreken. Een mooi, goed functionerend systeem is niet per definitie een goed systeem als het gebouwd is om een ander probleem op te lossen dan waar het voor ontworpen is.

Requirements Engineering en Requirements Modeling

Requirements Engineering is het proces van het boven water krijgen en formeel vastleggen van eisen (vanaf nu: **requirements**) aan een bepaald informatiesysteem, alvorens aan de implementatie ervan te beginnen. Requirements Engineering wordt toegepast in verschillende bedrijfssectoren, maar vooral op gebied van software engineering. Goede toepassing van het Requirements Engineeringproces bevordert de bouw van het *juiste*

informatiesysteem als oplossing van het *juiste* probleem. Het eindproduct van Requirements Engineering is een requirements specification, waarin de requirements van de klant en stakeholders nauwkeurig zijn vastgelegd.

Requirements modeling is een essentieel onderdeel van Requirements Engineering, wat het in kaart brengen van requirements op grafische of tekstuele wijze omvat. Door goed gebruik van requirements modeling binnen het proces van Requirements Engineering kan door de belanghebbenden en de systeemontwikkelaar goed, duidelijk en eenduidig gecommuniceerd worden over de eisen aan het systeem.
Gedurende het gehele Requirements Engineeringproces worden de requirementsmodellen bijgewerkt, gevalideerd en geverifieerd, zodat aan het einde van het proces een eenduidige requirements specification met duidelijke requirementsmodellen en de daarbij behorende beschrijvingen wordt opgeleverd.

Verminder de kosten, bevorder efficiëntie

Het bouwen van informatiesystemen kost tijd, geld en mankracht, middelen die elke onderneming zo efficiënt mogelijk wil spenderen. Men is in de IT-sector, waarin gewoonlijk grote geldbedragen omgaan en gehouden moet worden aan strakke tijdsschema's, steeds op zoek naar zaken om met kosten- en tijdsbesparing de eerdergenoemde efficiëntie te bewerkstelligen.

Tijd

Sommige fasen in het systeemontwikkelingsproces vergen menselijke interactie, wat weer verbonden is met de middelen tijd, geld en mankracht. Deze interactie met belanghebbenden, of **stakeholders**, moet zoveel mogelijk resultaat opleveren in zo min mogelijk tijd. Het ligt echter in de menselijke aard om informatie en kennis soms achter te houden voor ondervragers met wie er geen of nog geen vertrouwensband bestaat. Ook kan het zijn dat men niet precies de informatie verstrekt die gewenst is door incompleetheid of onjuistheid van de informatie. Het kan dan lastig zijn om deze essentiële zaken te achterhalen, wat weer vertraging oplevert in het systeemontwikkelingsproces.
Een probleem binnen Requirements Engineering is dat de stakeholders gemotiveerd moeten worden om informatie en kennis over te dragen, wat tijd kost. Ook is het vaak lastig om het proces van Requirements Engineering met meerdere personen tegelijkertijd te kunnen doorlopen, aangezien alle deelnemers tijd moeten vrij maken om tegelijkertijd aanwezig te kunnen zijn.

Kosten

Informatiesystemen worden in eerste instantie gebouwd om problemen op te lossen. De ontwikkeling ervan brengt kosten met zich mee, die zo laag mogelijk moeten worden gehouden. De kosten van de bouw van software kunnen redelijk in de hand worden gehouden. Er zijn echter situaties waarin er middels virtualisatie, grote besparingen kunnen worden gerealiseerd.

Een groot voordeel van een informatiesysteem is het *virtuele karakter* ervan. De training van een grote groep mensen in een vluchtsimulator, de uitvoer van een wijziging in een wegennet en daarop volgend de evolutie volgen van het verkeer, en de aerodynamica testen van een verandering in de structuur van een spaceshuttle, zijn voorbeelden van situaties waarin een virtuele oplossing enorme kostenbesparing met zich meebrengt. Men kan op basis van modellen en theorieën processen simuleren, zonder dat de fysieke componenten als een vliegtuig of wegennet daadwerkelijk gebruikt hoeven te worden.

Serious Gaming

Serious Gaming is een potentieel hulpmiddel als oplossing voor tijds- en kostenproblemen. Het is een relatief nieuw concept, waarbij trainingen worden gegeven, bedrijfsprocessen gesimuleerd en informatie en kennis uitgewisseld door gebruik van een spelgeoriënteerde omgeving.
Deze spelgeoriënteerde omgeving bevat een virtuele speelwereld binnen een applicatie, waarin de gebruiker van de applicatie een hoofdrol vervult. Door acties uit te voeren in de speelwereld, die aan bepaalde spelregels gebonden zijn, ondergaat de gebruiker een interactieve ervaring van een leerproces. Doordat er binnen de speelwereld interactie plaatsvindt tussen de gebruiker en de omgeving, worden de gevolgen van zijn of haar acties duidelijk.

Serious Gaming levert tijdsbesparing op door deelnemers intensief te betrekken in een proces en zo gemakkelijker informatie en kennis overdragen. Ook kunnen processen worden versneld om te kijken wat het effect van een actie is, zonder dat daarop in de 'werkelijke' tijd te hoeven wachten.
Kosten worden door Serious Gaming bespaard door de eerder genoemde tijdsbesparing en doordat processen op virtuele wijze snel kunnen worden uitgevoerd en dure apparatuur kan worden gesimuleerd.

Daarnaast zorgt Serious Gaming ervoor, dat deelnemers door simulatie van een echt bestaande omgeving, op veilige, gecontroleerde en herhaalbare manier een virtuele ervaring beleven van iets wat gewoonlijk veel geld en tijd zou kosten [**Verbraeck**].

Kernonderwerp
Het kernonderwerp van dit artikel is de beantwoording van de vraag:

"Hoe kan de essentie van requirements modeling worden ingekaderd in een spelachtige procedure?"

De intentie van dit artikel is om Serious Gaming toe te passen als instrument om de Requirements Engineering fase bij software engineering te vergemakkelijken. Hierbij wordt voornamelijk gekeken naar het aspect van requirements modeling. Hiertoe zal er eerst een theoretisch onderzoek plaatsvinden naar de *best practices* of beste ontwerpbeslissingen bij Serious Gaming, waarna er een testapplicatie zal worden gebouwd en te getest om te kijken hoe Serious Gaming in Requirements Engineering kan worden gebruikt.

Opbouw
De opbouw van dit artikel is als volgt:

Hoofdstuk 2 biedt een theoretisch kader betreffende het begrip Requirements Engineering, het vertelt hoe het proces van Requirements Engineering werkt, welke stappen er in dat proces zijn en welke rol requirements modeling daarbij speelt.
Hoofdstuk 3 legt het principe van Serious Gaming uit en biedt een theoretisch kader, beschrijft de opbouw en toepassing van Serious Gaming en de best practices die worden gebruikt bij het ontwerpen van een Serious Game.
Hoofdstuk 4 beschrijft de testapplicatie die geïmplementeerd gaat worden met de beargumentering achter de keuzes die gemaakt worden tijdens het ontwerpen ervan.
Hoofdstuk 5 bevat de resultaten, analyse en conclusies van de experimenten met de testapplicatie.
Hoofdstuk 6 beschrijft alternatieven op de gebruikte methode.
Hoofdstuk 7 biedt tot slot een conclusie en beantwoording van de vraag uit het kernonderwerp.

Hoofdstuk 2: Requirements Engineering

"Software is like entropy: It is difficult to grasp, weighs nothing, and obeys the Second Law of Thermodynamics; i.e., it always increases."

- Norman Augustine

Definities

Requirements

Een *requirement* wordt door Kulak en Guiney [**Kulak**] omschreven als '*iets dat een computerapplicatie moet doen voor zijn gebruikers*'. Het is een specifieke functie, eigenschap, kwaliteit of principe dat het systeem moet aanbieden om zijn bestaan te rechtvaardigen.

Requirements beschrijven *wat* een systeem moet doen, en *waarom*. Methoden die beschrijven *hoe* een systeem iets moet doen, zijn invullingen van de implementatie van requirements en behoren niet tot Requirements Engineering, maar tot de ontwerpfase.
Op basis van requirements wordt een functioneel ontwerp gemaakt, wat aan de basis staat van de implementatie van een informatiesysteem.

Functional en non-functional requirements

Requirements in Requirements Engineering kunnen grofweg worden ingedeeld in twee categorieën, de functional requirements en non-functional requirements.
Functional requirements zijn de eisen aan het systeem die de functies en eigenschappen beschrijven van het systeem, of het gedrag ervan, die gebruikers nodig hebben om het systeem te laten werken.
Non-functional requirements omvat de eisen aan het systeem die niet direct met functionaliteit te maken hebben, maar welke kunnen worden gebruikt om een goede werking ervan te eisen. Voorbeelden van non-functional requirements zijn robuustheid van het systeem, schaalbaarheid en beveiliging van gegevens.

Requirements Engineering

Requirements beschrijven kort gezegd de formele eisen waaraan een systeem moet voldoen, specifiek de eisen van de belanghebbenden van het te implementeren systeem, de *stakeholders*.
Het proces van het vaststellen van de requirements aan een systeem, het valideren, verfijnen en documenteren van die requirements heet *Requirements Engineering* (**RE**).

RE zorgt voor een solide basis voor ontwerp en constructie van een informatiesysteem. Zonder duidelijke, formeel vastgelegde requirements is er een hoge kans dat het op te leveren systeem niet voldoet aan de wensen van de klant. Net zoals een architect in de bouw geen gebouw kan ontwerpen als hij niet weet wat het doel van het gebouw is en wat de

randvoorwaarden voor de bouw zijn, weet een systeemontwikkelaar niet wat hij voor systeem zal ontwerpen als de requirements ervoor niet nadrukkelijk kenbaar zijn gemaakt.

Hoewel er veel verschillende methoden van systeemontwikkeling zijn, of het nu de watervalmethode, het V-model, het spiraalmodel of incrementele methode betreft, zijn de verschillende fasen binnen die methoden vaak hetzelfde. Het proces van systeemontwikkeling wordt dan in deze volgorde doorlopen:

- **Communicatie**, initiatie van een project, verzamelen van requirements
- **Planning**, tijd en kosten schatten, schema's opstellen
- **Modelleren**, analyse en ontwerp
- **Constructie**, implementatie van code en testen
- **Uitrol en onderhoud**, afleveren van het systeem, leveren van feedback en support

Vanuit het perspectief van systeemontwikkeling is RE te zien als een systeemontwikkeling die begint met de activiteit van communiceren en eindigt in de activiteit van het modelleren [**Pressman**] . In de eerste drie fasen van het proces vormt RE de exacte blauwdrukken van eisen waarop het ontwerp wordt gebaseerd, wat op zijn beurt de basis vormt voor de uiteindelijke implementatie. In de fase van uitrol en onderhoud speelt RE ook een rol bij het bijhouden van veranderende requirements aan het systeem en de daarop volgende aanpassingen.
Feitelijk maakt RE deel uit van het softwareontwikkelingsproces, maar omdat software binnen een informatiesysteem niet op zich kan functioneren zonder interactie met de rest van dat systeem, wordt RE ook wel gezien als deel van het systeemontwikkelingsproces [**Nuseibeh**] .

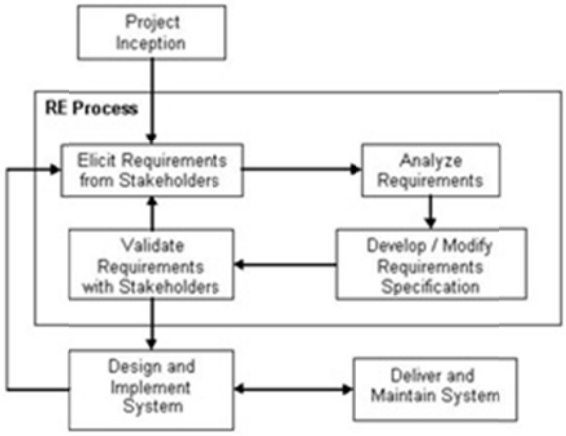
Het Requirements Engineeringproces

Het Requirements Engineeringproces bestaat uit verschillende fasen, die elk tot doel hebben te definiëren wat de klant wil en om een sterke fundering te realiseren voor het ontwerp en de constructie van het eindproduct. Omdat het proces op meerdere manieren kan worden doorlopen, bestaan er verschillende opvattingen over welke fasen het proces inhoudt.

Zo definieert Pressman [**Pressman**] zeven fasen van RE, die elkaar lineair opvolgen. Deze zijn inception (aanvang), elicitation (boven water krijgen), elaboration (verfijnen), negotiation (onderhandeling), specification (specificeren), validation (goedkeuring) en management. Loucopoulos [**Loucopoulos**] definieert alleen de fasen elicitation, specification en validation. Sommerville en Kotonya [**Sommerville**] definiëren de fasen elicitation, analyzation/elaboration, modification en validation (Figuur 1).

In principe werken al deze methoden hetzelfde, met als nuances dat de verschillende opvattingen van het Requirements Engineeringproces ervoor zorgen dat in bepaalde methoden meer nadruk wordt gelegd op een bepaalde fase in het proces. Binnen de

methoden kunnen bepaalde fasen worden overgeslagen of toegevoegd, wat van het project en de projectontwikkelaars afhangt. Ze worden vaak simultaan uitgevoerd om betere communicatie en tijdswinst tot stand te brengen.



Figuur 1. Het Requirements Engineeringproces volgens Sommerville en Kotonya. Uit [Sommerville].

Requirements Inception

Inception is de fase waarin aan de ontwikkeling van een informatiesysteem wordt begonnen. Stakeholders zien een business case die moet worden uitgevoerd en proberen de markt af te tasten qua diepte en breedte en stellen een werkomschrijving vast voor de scope van het project. Al deze informatie zal later in het proces nog veranderen, maar voor het moment volstaat als stof om over te discussiëren met de ontwikkelaar die het informatiesysteem gaat ontwikkelen.

Requirements Elicitation

Elicitation is de fase waarin de belanghebbenden zoals de klant, de gebruikers en anderen, zoals de databasedesigner, back-end specialist et cetera, wordt gevraagd wat de doelen van het systeem of product zijn. Deze doelen worden dan gebruikt om om te zetten in requirements. Een ander belangrijk onderwerp is de vraag hoe het product bij de eisen van de dagelijkse gang van zaken in de businesskant van het verhaal past. Verder wordt er gekeken hoe het systeem of product zal worden gebruikt in het dagelijkse leven.

Requirements elicitation is een erg lastige fase, waarin stakeholders hun wensen op tafel leggen. Christel en Kang [**Christel**] noemen enkele problemen op die de kop op kunnen steken: Vaak zijn het meerdere stakeholders met verschillende eisen, weten stakeholders niet wat ze precies willen dat het systeem doet, stellen ze eisen die niet te behalen zijn met het gestelde budget en tijdschema, hebben stakeholders conflicterende eisen en veranderen de requirements over tijd.

Er bestaan veel verschillende technieken waarmee requirements elicitation kan worden uitgevoerd. Elke techniek heeft situaties waarin die techniek het beste worden aangepast. Enkele daarvan zijn:

- *Vraaggesprekken*
De gebruiker wordt gevraagd wat hij wil dat het systeem gaat doen. Dit kan gebeuren middels een interview, brainstormsessie of vragenlijst. Het is een gemakkelijk uit te voeren techniek, die als nadeel heeft dat in groepsgesprekken enkele dominante personen de boventoon kunnen krijgen en dat de ondervraagden vaak in jargon praten en denken, wat lastig kan zijn om concreet om te zetten in requirements.
- *Taakanalyse*
Een analyse van taken die de gebruikers in de dagelijkse gang van zaken uitvoeren, wordt uitgevoerd. Hierbij worden de taken en subtaken in een hiërarchie onderverdeeld.
- *Prototyping*
Prototyping wordt gebruikt als de stakeholders onzeker zijn over de requirements. Er kan dan een prototype worden gebouwd, dat gebruikt kan worden om ervaring op te doen en naderhand nieuwe of betere requirements op te stellen. Een nadeel van prototyping is dat stakeholders het prototype kunnen zien als het definitieve eindproduct en verdere medewerking overbodig kunnen gaan vinden [**Kulak**] .
- *Scenario gebaseerde analyse*
Bij scenario gebaseerde technieken wordt gekeken naar concrete voorbeelden van handelingen die gebruikers uitvoeren. Scenario gebaseerde analyse wordt ook wel *Use Case analyse* genoemd, waarin Use Cases worden gebruikt en is een populaire techniek binnen RE. *Use Cases* of *user scenario's* zijn tekstuele requirementsmodellen waarin de handelingen van gebruikers gedetailleerd zijn vastgelegd.

Requirements Elaboration
Tijdens het elaborationproces wordt er gefocust op het ontwikkelen van een verfijnd technisch model van de functies, functionaliteiten en randvoorwaarden van het systeem. Het elaborationproces zelf bestaat uit een aantal modelleer- en verfijntaken. Drijfveer achter het geheel is de verfijning van de Use Cases, die beschrijven hoe de eindgebruikers interacteren met het systeem. Elke Use Case wordt geparseerd en geanalyseerd om daaruit *analysis classes* te verkrijgen, klassen van entiteiten binnen het systeem die zichtbaar zullen zijn bij de eindgebruiker. De attributen van elke analysis class worden gedefinieerd en de operaties die elke klasse nodig heeft, worden geïdentificeerd. De (hiërarchische) relaties en samenwerkingsverbanden tussen klassen worden geïdentificeerd. Dit alles leidt tot een reeks UML-diagrammen. Het eindresultaat van de elaborationfase is een analysemodel dat het informationele en functionele domein van het systeem beschrijft.

Requirements Negotiation
Gedurende het negotiationproces overleggen klanten, gebruikers en andere stakeholders over de prioriteiten van de requirements en proberen eventuele conflicten tussen de

verschillende prioriteiten op te lossen. Ruwe schattingen over ontwikkelkosten in termen van projectkosten en tijd worden geraamd, waarna de requirements op iteratieve wijze worden gecombineerd, verwijderd of aangepast zodat elke partij tevreden is.

Requirements Specification
Tijdens het specificationproces worden beschrijvingen gecombineerd tot een geschreven document, de requirements specification. Deze beschrijvingen kunnen variëren van informeel (natuurlijke taal en grafische modellen) tot zeer formeel (wiskundige modellen).

Requirements Validation
Het validationproces is een vitaal onderdeel van Requirements Engineering. Requirements worden gevalideerd en beoordeeld op consistentie en kwaliteit. Inconsistenties, fouten en weglatingen worden gecorrigeerd.

Dit valideren gebeurt in een formeel technisch reviewteam. Het reviewteam bestaat uit systeemontwikkelaars, klanten, gebruikers en andere stakeholders, die de requirements specification doorspitten en zoeken naar inhoudelijke fouten of interpretatiefouten. Naast het vaststellen dat de *juiste eisen* zijn vastgelegd (validatie), moet ook duidelijk worden of deze eisen *juist zijn vastgelegd* (verificatie).

Requirements Management
Bij het managementproces worden de requirements geïdentificeerd. Elke requirement krijgt een unieke code, waarmee *traceability tables* worden gegenereerd. Traceability zorgt voor transparantie in de verzameling requirements en houdt bij wie welke requirement heeft opgesteld, welke requirements van elkaar afhankelijk zijn en welke requirement bij welke business requirement hoort. Deze traceability tables worden opgeslagen in een requirementsdatabase als naslagwerk. Verder omvat requirements management het bijhouden welke functies van het systeem bij welke eis horen, aan welke bronnen een requirement ontsproten is, hoe bepaalde requirements van andere requirements af hangen, het onderschikken van requirements in subsystemen en het aan kunnen tonen hoe requirements relateren aan interne en externe systeeminterfaces.

Requirements modeling

RE is niet alleen het proces van blootleggen en specificeren van requirements, maar dient ook om effectieve communicatie van deze requirements tussen de diverse stakeholders mogelijk te maken [**Nuseibeh**]. Om gemakkelijk informatie over te kunnen dragen, wordt tijdens het proces van RE gebruik gemaakt van modellen. Het proces waarbij die modellen worden gegenereerd heet *requirements modeling*. Nuseibeh en Easterbrook [**Nuseibeh**] omschrijven requirements modeling als ‘de constructie van abstracte omschrijvingen die helpen interpreteren [bij overdracht van informatie]’.

De requirementsmodellen die requirements modeling oplevert, beschrijven de essentiële entiteiten in het gemodelleerde proces, hun eigenschappen en hun onderlinge relaties in tekstuele en grafische vorm. De modellen veranderen in de loop der tijd omdat systeemontwikkelaars steeds meer leren over het te bouwen informatiesysteem en stakeholders steeds meer begrijpen wat ze nodig hebben. Daarom is een requirementsmodel feitelijk een momentopname van de requirements op een bepaald moment. Met de evolutie van het requirementsmodel worden bepaalde elementen relatief stabiel, zodat ze uiteindelijk een robuuste fundering bieden voor de ontwerpproces, dat het proces van RE opvolgt.

Er bestaan verschillende soorten requirementsmodellen, elk met hun eigen functies en eigenschappen. Vaak leggen ze vast welke acties er moeten worden uitgevoerd door de *actors*, de personen of entiteiten die een rol spelen binnen het systeem.

Soorten modellen
Requirementsmodellen bestaan in vele vormen. De modellen of diagrammen worden opgebouwd door entiteiten binnen een systeem vast te stellen, de onderlinge banden en eigenschappen van die entiteiten te modelleren. De manier waarop ze met elkaar zijn verbonden hangt af van het soort model.
Een veelgebruikte objectgeorienteerde modelleertaal is de Unified Modeling Language, of UML. UML is in de jaren '90 van de vorige eeuw ontwikkeld door Grady Booch, James Rumbaugh en Ivar Jacobson en geldt sinds 1997 als internationale standaard. UML omvat een grote collectie modellen, die veelal worden gebruikt in de Requirements Engineering. Deze modellen kunnen grofweg worden ingedeeld in twee categorieën, modellen die de structuur van een systeem in kaart brengen, en modellen die het gedrag van een systeem weergeven.

Hieronder volgt een opsomming van de belangrijkste soorten modellen die in requirements modeling worden gebruikt.

- Scenariogebaseerde modellen
Deze modellen, eerder beschreven in de methoden van requirements elicitation, dienen vaak als input voor de creatie van andere requirementsmodellen. Ze beschrijven het systeem vanuit het oogpunt van de gebruiker door een scenariogebaseerde aanpak. Deze scenario's, die stapsgewijze interacties tussen de gebruiker en het systeem beschrijven, kunnen op hun beurt weer op worden gedeeld in subscenario's. Een bekende implementatie van scenariogebaseerde modellen zijn Use Cases en Use Case diagrammen (Figuur 2).

- Klassengebaseerde modellen
Elk scenario bevat een aantal entiteiten die worden gemanipuleerd wanneer een actor met het systeem interacteert. Klassengebaseerde modellen brengen de 'fysieke' structuur van de entiteiten binnen het systeem in beeld, inclusief verbindingen met andere entiteiten en een eventuele onderlinge hiërarchie. Deze entiteiten zijn gecategoriseerd in de analysis classes uit

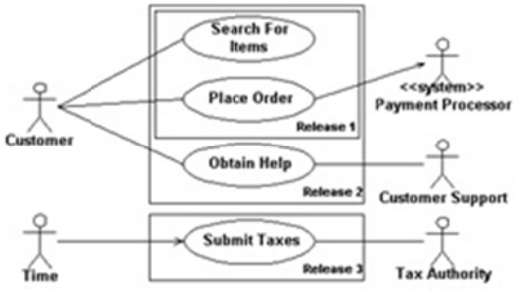
de requirements elaboration fase. Deze analysis classes beschrijven welke eigenschappen de instanties van een klasse hebben en welke functies ze bezitten. Een voorbeeld van een analysis class kan 'auto' zijn. In een klassengebaseerd model kan een auto worden ondergeschikt aan de superklasse 'voertuig' en een band hebben met een klasse 'kentekenregistratie'. Een veelgebruikte toepassing van klassengebaseerde modellen is een UML Class diagram (Figuur 3).

- Gedragsmodellen

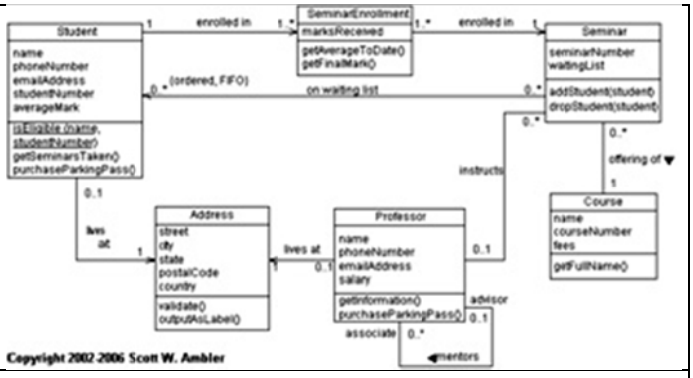
Gedragsmodellen brengen het gedrag van een systeem in beeld. Het gedrag van een informatiesysteem kan een grote invloed hebben op de keuze van ontwerp en de manier van implementatie. Om deze reden is het van belang om naast de functionele en structurele kant, ook het gedrag van het systeem zichtbaar te maken gedurende bepaalde stappen in het uitvoeringsproces. Men kan daarbij denken aan een UML State diagram (Figuur 4) dat de verschillende staten van gedrag van het systeem uitbeeldt. In het model worden verschillende staten weergegeven, met elk een invulling van systeemvariabelen en activiteiten die zich binnen die staat bevinden.

- Stromingsgebaseerde modellen

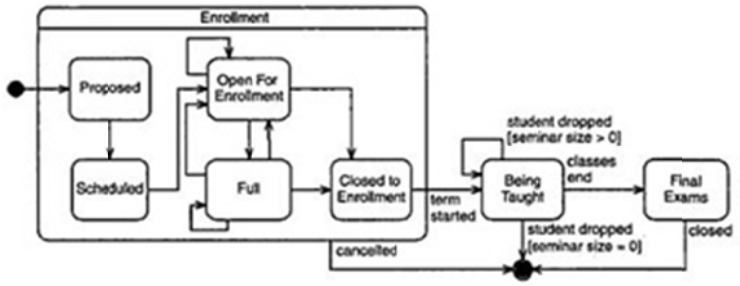
Informatie wordt getransformeerd terwijl het door het informatiesysteem stroomt. De stromingsgebaseerde modellen beschrijven welke input van informatie er is, welke transformaties er op welke plek in het systeem plaatsvinden en van welke output van informatie het systeem levert. Een voorbeeld van toepassing van stromingsgebaseerde elementen is een swim lane diagram (Figuur 5).



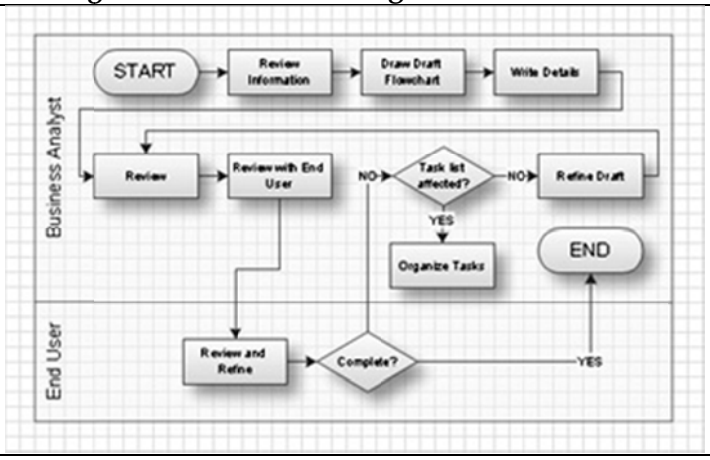
Figuur 2. Use Case Diagram. Uit [Ambler].



Figuur 3. UML Class Diagram. Uit [Ambler].



Figuur 4. UML State Diagram. Uit [Ambler].



Figuur 5. Swim Lane Diagram. Uit [Concept].

Ontwikkeling van RE

In de jaren 70 van de twintigste eeuw bestond er nog geen expliciete vraag naar het creëren, begrijpen en gezamenlijk opstellen van requirementsspecificaties. Het proces bestond toen niet als aparte entiteit, maar werd geschaard onder de algemene noemer 'engineering'. Softwareontwikkelaars werden geleerd specificaties te produceren in elke vorm van presentatie die in dat geval nodig was. Soms was dit in tekstuele vorm, soms als tabellen data en vaak in de vorm van afbeeldingen.

Deze specificaties en documentatiemethoden produceerden systeemmodellen die gebruikt werden als basis voor tests nadat het product was gecreëerd. In de jaren '70 werden de requirements van de klant gedocumenteerd in een zogenaamde *customer requirements specification*.

In de jaren '80 werd bedacht dat het woord 'customer', 'klant' betekenend, te beperkend was. Het zou immers kunnen impliceren dat het alleen de requirements representeerde van de persoon die het systeem betaalde. Om dit tegen te gaan werd het woord 'user' of 'gebruiker' gebruikt. Customer requirements specification werd *user requirements specification*.

In de jaren '90 bedacht men, dat het woord 'user' ook te beperkend was. Het gaf de indruk dat alleen op de eindgebruiker van het systeem werd gefocust, in plaats van ook op onderdelen als testen, onderhoud, verkoop en transport, waarvoor ook requirements

opgesteld moeten worden. In de jaren '90 werd het woord 'stakeholder' steeds vaker gebruikt. User requirements specification werd *stakeholder requirements specification*.

In deze periode werden de processen die gebruikt werden om de kwaliteit van requirementsspecificaties te waarborgen, niet meer aangeduid als engineering, maar als *requirements management*.

Rond de millenniumwisseling begon men de termen requirements management en *Requirements Engineering* beide te gebruiken om hetzelfde aan te duiden. In deze periode leidde de strikte betekenis van het woord 'stakeholder' ertoe, dat de stakeholder requirements specification de requirements van alle stakeholders ging bevatten, zelfs die van stakeholders die niet direct met het op te leveren systeem zouden werken. Hierdoor werd het document dat de requirements representeerde, uitgebreid met zowel de oplossing voor die requirements en vele andere requirements van gebruikers die niets te maken hadden met de oorspronkelijke, betalende klant.

In de jaren '2000 veranderde de specificatie van requirements die een klant stelde, weer in een *customer requirements specification*, net zoals dertig jaar daarvoor. Men was weer terug bij af [**Hood**] .

Tijdperk	**Term voor specificeren eisen (requirements)**	**Term voor het document van gespecificeerde eisen**	**Term voor het document van eindproduct**
Jaren '70	Engineering	Customer requirements specification	Product specification
Jaren '80	Engineering	User requirements specification	Functional requirements specification
Jaren '90	Requirements Management	Stakeholder requirements specification	System requirements specification
Jaren '2000	Requirements Management & Engineering	Customer requirements specification	System requirements specification

Tabel 1. Evolutie van naamgeving van Requirements Engineering.

Voor- en nadelen van RE

Voordelen

Hoewel niet iedere softwareontwikkelaar aan RE doet, is het verstandig dat wel te doen. Naast de tijd- en kostenbesparing door het voorkomen van het bouwen van het verkeerde systeem, zijn er nog andere voordelen die juist gebruik van Requirements Engineering opleveren.

Scope creep

Goed gebruik van RE zorgt er vroeg in het softwareontwikkelproces voor, dat problemen

later in het proces minder snel de kop op duiken. Zo wordt een fenomeen dat *scope creep* heet, voorkomen door goede Requirements Engineering. Scope creep is het fenomeen van ongecontroleerde veranderingen in de scope van het project. Dit houdt in, dat er nieuwe eisen worden gesteld aan het op te leveren systeem of project, zonder additionele financiële middelen, mankracht of tijd. Door een goede requirements specification te maken met daarin precies alle requirements, op prioriteit gesorteerd, heeft men een middel om scope creep tegen te gaan. Er is daarbij van tevoren vastgesteld welke functionaliteiten er definitief in het nieuwe systeem komen en welke functionaliteiten eventueel weg kunnen worden gelaten.

Contractuele eisen
Omdat tijdens RE gezamenlijk overleg plaatsvindt tussen de stakeholders en de systeemontwikkelaar die het systeem gaat implementeren, nemen de partijen zichzelf met een goede requirements specification in bescherming tegen elkaar. De klant krijgt precies waar hij voor betaalt, de ontwikkelaar hoeft niets extra's op te leveren buiten de functionaliteiten die vastgesteld zijn. Dit zorgt ervoor dat contracten minder snel ontbonden kunnen worden op basis van niet geleverde producten of de claim daarop. Door te communiceren via een gezamenlijk opgesteld formeel document wordt het onderlinge vertrouwen vergroot.

Kwaliteit eindproduct
Door naast de functional requirements, ook de non-functional requirements vast te stellen en te documenteren, kan niet alleen een correct systeem worden gegarandeerd, maar ook een correcte werking ervan. Dit komt niet alleen de kwaliteit van het eindproduct ten goede, maar het vergemakkelijkt ook onderhoud na oplevering.

Nadelen
Helaas heeft Requirements Engineering ook zijn nadelen. De uitvoer van een correct Requirements Engineering proces kost tijd en geld. Er is veel vooronderzoek nodig, waarbij stakeholders vaak meerdere keren worden benaderd voor bijeenkomsten, validatie en verificatie van requirements. Daarnaast moeten veel documenten worden gecreëerd en onderhouden [**McPhee**].
Bij de uitvoer van requirements management, waarbij veranderingen in de requirements worden bijgehouden, komt het voor dat de bestaande software moet worden aangepast doordat requirements veranderen, wat ook weer tijd en geld kost. Een nare bijkomstigheid hiervan is dat dit vaak moeizaam verloopt en gevoelig is voor fouten die optreden bij het implementatieproces.

Hoofdstuk 3: Serious Gaming

"Tell me and I forget. Teach me and I remember. Involve me and I learn."
- Benjamin Franklin

Doordat de huidige generatie professionals steeds meer en meer met communicatie- en informatietechnologie werken, is het een logische ontwikkeling dat leermethoden zich ook verplaatsen richting de virtuele wereld. Serious Gaming is een zich ontwikkelende niche in de wereld van computersoftware. Het speelt daarop in door naast een aantrekkelijke en betrekkende, spelgeoriënteerde omgeving, ook een ingewikkelde simulatie te brengen met een serieuze boodschap. De speler krijgt een complex probleem voor de kiezen, dat hij of zij op actieve manier moet oplossen. De doelen van Serious Gaming zijn vooral om betrokkenheid bij het onderwerp te stimuleren, samenwerking op vakgebied te stimuleren en kennis over te dragen die behouden blijft, doordat het vergaard is op een vermakelijke manier. Niemand wilt een saaie test doen, maar iedereen wilt aan een spelshow meedoen!

Definitie

Serious Gaming heeft diverse definities, waarvan die van Mike Zyda [**Zyda**] vaak wordt gebruikt.

***Game**: Een fysieke of mentale wedstrijd, gespeeld volgens specifieke regels, met als doel het amuseren of belonen van de deelnemer.*

***Video Game**: Een mentale wedstrijd, gespeeld met een computer volgens specifieke regels voor amusement, recreatie of winnen van een inzet.*

***Serious Game**: Een mentale wedstrijd, gespeeld met een computer volgens specifieke regels waarbij entertainment wordt gebruikt om doelen met betrekking tot bestuurlijke of commerciële training, educatie, gezondheid, publiek belang en strategische communicatie te bewerkstelligen.*

Ontwikkeling

Al gedurende lange tijd zijn Serious Games ontwikkeld en gebruikt in de softwarewereld. Al in 1980, voordat de personal computer zijn toetreden tot de huiskamer deed, werd 'Battlezone' uitgebracht, een computerspel voor de Atari, waarin men in een 3d model van een tank rondreed. Een jaar later werd het spel in de vorm van Serious Game uitgebracht onder de naam The Bradley Trainer, waarmee het Amerikaanse leger zijn troepen liet trainen in het Bradley militaire voertuig [**Stone**] .

In 2002 verkreeg Serious Gaming meer bekendheid, doordat het Serious Gaming Initiative werd opgezet door het Woodrow Wilson International Center for Scholars. Dit Amerikaanse initiatief stimuleert de ontwikkeling van spellen die politieke en managementzaken adresseren [**WikiSG**] . Het SGI heeft sinds de oprichting meerdere artikelen gepubliceerd en

een workshop gehouden in 2003 wat resulteerde in middelen voor de ontwikkeling van Serious Games voor parken, ziekenhuizen en middelbare scholen.

De historie van Serious Gaming gaat echter veel verder terug, hoe lang precies weet niemand. Lang geleden was Serious Gaming al in de praktijk gebracht. Het immer populaire Backgammon, het Japanse strategiespel Go en het moderne Kolonisten van Catan zijn slechts voorbeelden van spellen die de deelnemers principes van gokken, strategie en handelsgeest bijbrachten.

Toepassing

Serious Games worden in veel verschillende sectoren gebruikt, in vele soorten en maten. Er zijn diverse genres binnen de overkoepelende term Serious Games, zoals
- *Advergames*, games met een primaire focus op commerciële productaanbieding,
- *Edutainment,* games met een focus op aanleren van vaardigheden of trainen van de speler,
- *Infotainment*, games met een focus op het overbrengen van informatie en kennis op de speler,
- *Therapeutische games*, games die worden gebruikt als alternatieve therapie bij de behandeling van ziekte of aandoeningen,
- *Propaganda*, games met een focus op het aanleren van een bepaald denkpatroon of gedrag aan de speler.

Serious Games kunnen meerdere genres bevatten. Hieronder volgen een paar toepassingen van Serious Games.

Educatieve sector

Sportcoaches gebruiken aangepaste versies van sportmanagerspellen als FIFA Manager (voetbal) en NFL Head Coach (American Football) om tactieken uit te denken, visualiseren en aan hun spelers te presenteren in de vorm van simulatie.

Commerciële sector

Bezoekers van Intel's website met zijn IT Manager 3 applicatie worden aangesteld als virtuele IT-manager binnen een fictief bedrijf. De gebruiker draagt als IT-manager verantwoordelijkheid over beslissingen met betrekking tot werknemers, cashflows en inzet van Intel technologie. [**Intel**]

Publieke sector

Het Nederlandse bedrijf Tygron heeft in samenwerking met enkele bedrijven Watergame ontwikkeld, waarin meerdere spelers een waterontwikkelingsproject uitvoeren. Het probeert de deelnemers kennis op gebied van gebiedsontwikkeling en watermanagement bij te brengen. [**Watergame**]

Medische sector

TruSim, een divisie van de Engelse Blitz Games Studios, heeft Interactive Triage Trainer ontwikkeld, een driedimensionale virtuele representatie van een werkelijke situatie waarin

professionals worden getraind hoe ze de behandeling van slachtoffers na een ramp moeten prioritiseren. [**Keegan**] [**Trusim**]

Militaire sector
Een met aangepaste mappen en scenario's uitgeruste versie van Tom Clancy's Rainbow Six, een spel dat voor het eerst in 1998 werd uitgebracht als strategisch schietspel voor de pc, wordt door het Amerikaanse leger gebruikt als trainingsmiddel voor antiterrorisme-troepen. [**Stone**]

Kenmerken

Omdat Serious Gaming een breed begrip is, is er geen bundel regels waaraan een ontwerper van een Serious Game zich moet houden. Er zijn wel enkele typische kenmerken aan een Serious Game;

De speler
De speler heeft een actieve actorrol in het spel en heeft de mogelijkheid om acties te ondernemen.

De speelwereld
Een unieke speelwereld waarin een probleem wordt voorgeschoteld aan de speler. De speelwereld heeft definities van spelelementen, vooraf vastgestelde acties die de speler kan ondernemen en regels waaraan de speler zich moet houden.

User interface
De user interface is optioneel, maar heeft vaak prestatie-indicatoren die aangeven hoe goed de speler het spel speelt. Deze indicatoren zijn per spel verschillend, maar gedacht kan worden aan kwaliteit van financieel beleid, morele beslissingen, visueel probleemoplossingvermogen et cetera.

Naast deze elementen is er ook begeleiding aanwezig, vaak in tekstuele of grafische vorm. Door deze begeleiding wordt de speler duidelijk gemaakt hoe de speelwereld in elkaar zit, welke regels er gelden en hoe de speler in het spel kan ingrijpen.
Vaak is er na afloop van het spel een evaluatie, waarbij duidelijk wordt of de speler iets heeft opgestoken van het spel. Dit kan ook een opzetje voor de ontwikkelaars van het spel zijn om leerdoelen van de Serious Game aan te passen of processen in het spel fijn te stellen.

Evolutie van het leerproces

Het succes van Serious Gaming gaat hand in hand met de verandering in leer- en denkprocessen bij de generatie die opgegroeid is met technologie. Deze 'Gaming Generation', beschreven in 2001 door Marc Prensky in zijn boek *Digital Game-based Learning* [**Prensky**], is opgegroeid in een wereld waarin media een grote rol speelt in het dagelijks leven. Prensky schrijft dat de groep mensen die na 1961 zijn geboren, en veelal geen analoge telefoon hebben gebruikt, geen tijd hebben gekend waarin muziek niet draagbaar was en

nooit hebben geleefd zonder honderdduizenden visuele beelden per dag voorbij te zien komen, een groot deel van de werkende wereldbevolking omvat. De Gaming Generation kan zich geen wereld voorstellen waarin digitale media, computers, computerspellen en internet niet bestaan. Deze nieuwe generatie heeft een totaal andere verzameling ervaringen ondergaan dan de voorgaande generatie, met nadruk een *digitale* verzameling ervaringen. De veranderingen van denkpatronen of cognitieve veranderingen door de komst van moderne digitale technologieën en media hebben geleid tot een variëteit van nieuwe eisen en voorkeuren bij deze relatieve generatie, met in bijzonder op gebied van leren.

De huidige werknemers van bedrijven hebben van jongs af aan dagelijks mysteries en puzzels opgelost, steden gebouwd en onderhouden, themaparken, bedrijven gerund, samenlevingen opgebouwd, vliegtuigen, helikopters en tanks bestuurd, vuurgevechten uitgevochten en strategische oorlogsplannen gesmeed, allemaal in de vorm van computerspellen. Deze gebeurtenissen vonden niet incidenteel plaats, maar steeds opnieuw en opnieuw, tot de spelers hun speelwijze perfectioneerden.

Prensky stelt dat er tien belangrijke veranderingen hebben opgetreden in het denkproces van de Games Generation, welke allen bijdragen aan het feit dat de Games Generation een grotere stroom van informatie kan verwerken en meer aangepast is op digitale leerwijzen.

Twitch speed vs. Conventional speed
De Games Generation is opgegroeid met informatie die op grote snelheid voorbij komt. Met de komst van televisiezenders die videoclips uitzenden met veel beelden per seconde zoals MTV en TMF en computerspellen waarin snel informatie moet worden opgenomen, verwerkt en actie moet worden ondernomen, is de huidige generatie gewend om meer informatie te verwerken dan de generatie ervoor, en vooral sneller.

Parallel processing vs. Linear processing
Het gebruik van computers voor verschillende taken tegelijkertijd zoals het luisteren naar muziek en het genereren van complexe digitale illustraties of het bekijken van webpagina's, en het gebruik van mobiele telefoons terwijl er televisie wordt gekeken, zijn voorbeelden van zaken waardoor de Games Generation gewend is om verschillende soorten informatie tegelijkertijd te verwerken. De zogenaamde 'tickers' onderin beeld bij zakelijke televisieprogramma's maken hier gebruik van; zij bieden de kijker naast de gebruikelijke nieuwsuitzending nog extra informatie over wereldwijde gebeurtenissen of de waarde van hun aandelen.

Random Access vs Step-by-step
De Games Generation was de eerste generatie die hypertext en 'rondklikken' op links op webpagina's ervoer. Door gebruik van edutainment in CD-ROMs en het internet, is de Games Generation gewend geraakt om informatie te vinden op meerdere locaties en in een minder sequentiële manier. Deze nieuwe manier van informatie zoeken heeft de Games

Generation bewust gemaakt van het feit dat er meerdere manieren zijn om aan informatie te komen dan een enkel pad dat lineair gevolgd wordt. De Games Generation is zich hiervan bewust en is in staat om meerdere, minder gestructureerde informatiebronnen te vinden en aan elkaar te koppelen. Dit heeft er ook toe geleid dat de huidige generatie gemakkelijker in structuren en patronen kan denken.

Graphics First vs. Text First

In de generaties voorgaand aan de Games Generation bestond grafisch materiaal voornamelijk uit illustraties, om tekst te begeleiden en tastbaarder te maken. Vandaag de dag is die relatie juist omgekeerd; de rol van tekst is om datgene tastbaar te maken dat als eerst wordt ervaren als beeld. De huidige generatie is van jongs af aan continue blootgesteld aan televisie, video's en computerspellen die zeer expressieve beelden in hoge kwaliteit, met geen of weinig bijgaande tekst. Patricia Marks Greenfield, professor in psychologie aan de University of California, Los Angeles, heeft dit fenomeen uitvoerig bestudeerd en concludeert in meerdere onderzoeken dat het spelen van computerspellen de representationele competentie aanscherpt, de vaardigheid om visuele beelden te interpreteren als representaties van driedimensionale ruimte. Deze steeds grotere voorkeur voor grafisch materiaal als leerbron brengt nieuwe taken met zich mee, waaronder het behoud van tekstueel alfabetisme en het behoud van diepte van informatie. De grote uitdaging is om deze focus op grafische informatie te gebruiken om begrip van de informatie te vergroten terwijl hetzelfde of zelfs een hoger niveau van informatiebeleving en –interpretatie wordt gehandhaafd in de nieuwe visuele context.

Connected vs. Standalone

De Games Generation is opgegroeid met mogelijkheden tot wereldwijde verbintenis in de vorm van email, bulletin boards, nieuwsgroepen, chat, multiplayer spellen en instant messaging. Het resultaat hiervan is dat de Games Generation anders denken over manieren om aan informatie te komen en problemen op te lossen dan hun voorgangers. In plaats van directe communicatie te gebruiken zoals een telefoongesprek om een vraag te stellen, zetten ze bijvoorbeeld een vraag op een bulletin board op internet, waar het gelezen wordt door duizenden mensen en een enorme bron van gemeenschappelijke informatie kan worden geraadpleegd. De verbondenheid van de Games Generation heeft ze ook minder begrensd door hun fysieke locatie en ze zijn meer bereid om te werken in virtuele teams die steeds meer hun intrede doen in de commercie en industrie, waarbij teamleden verspreid over de hele wereld samenwerken via digitale communicatiemiddelen.

Active vs. Passive

Een van de belangrijkste verschillen tussen een oudere generatie en de Games Generation is de ontdekkingslustigheid van de laatstgenoemde. Leden van oudere generaties willen een handleiding van een product lezen alvorens ermee aan het werk te gaan. De Games Generation is gewend zonder angst ergens op te klikken, iets in te drukken, iets uit te voeren en te kijken wat het effect is. Leden van de Games Generation denken nauwelijks aan een

handleiding bij het uitproberen van neen nieuw product. Ze spelen met software, slaan elke toets aan indien nodig, tot ze snappen hoe het werkt. Als het niet lukt, nemen ze aan dat het probleem bij de software ligt en niet bij hen – software *hoort* je te leren hoe je het moet gebruiken. Prensky geeft aan dat er minder draagvlak is bij de Games Generation voor passieve situaties als colleges en traditionele vergaderingen, ze geven de voorkeur aan actievere ervaringen zoals chat, berichten posten op internet en digitale leerwijzen. Hierbij zijn ze niet alleen actiever bezig, maar hebben ze ook meer controle over wat er gebeurt.

Play vs. Work

Door het gebruik van moderne digitale media is de Games Generation gewend bijna alles te zien als een spel. Voor hen is spelen te zien als werk. Het feit dat de spellen die gespeeld worden in de echte wereld een grotere en serieuzere impact hebben dan de virtuele spellen maakt voor hen niets uit. Het bereiken van doelen, winnen en de tegenstanders verslaan horen bij de ethiek en het proces ervan. Een moeilijke uitdaging die Prensky stelt is dat de managers en trainers een manier moeten vinden om de speelse instelling van de jongere generatie te incorporeren in de 'echte' wereld van zakendoen.

Payoff vs. Patience

Een van de grootste lessen die de Games Generation heeft geleerd van het opgroeien met computerspellen is dat als je voldoende tijd besteedt aan een spel en alle vaardigheden onder de knie krijgt, je zult worden beloond, zij het met een volgend level om te spelen, een overwinning of een plaats op de lijst van high scores. Wat je doet in het spel bepaalt wat je krijgt, en wat je krijgt is de moeite die je erin gestopt hebt, waard.
Het gevolg van deze ontwikkeling is een intolerantie bij de Games Generation voor zaken waarvoor de beloning onvoldoende bevredigt. Waarom zou je je studie afmaken als middelbare scholieren professionele websites kunnen maken, jongeren multimiljonairs kunnen worden met zelf opgerichte bedrijven en Bill Gates, die aan Harvard studeerde, weggegaan is om iets te doen met een betere 'payoff', het worden van de rijkste man ter wereld?
De uitdaging voor managers en trainers hierin ligt in het vinden van manieren om medewerkers van de Games Generation te belonen in het hier en nu, in plaats van op de langere termijn. Deze trend toont zich in de neiging naar beloning naar prestaties, het toenemende gebruik van vermogen van een bedrijf als onderdeel van compensatie voor de werknemers en het vrijmaken van budget binnen bedrijven voor interne startups, waarbij werkers in staat worden gesteld hun beloning eerder binnen te halen door een startup te leiden en waarbij het bedrijf dat de startups financiert en er aandeelhouder in is, ook profiteert door de groei in eigen vermogen.

Fantasy vs. Reality

De Games Generation is opgegroeid met populaire cultuur in de genres fantasy en science fiction. Hoewel fantasie jonge mensen altijd al heeft geprikkeld, heeft de computer het met zijn virtuele wereld gemakkelijker en realistischer gemaakt om fantasie tot leven te brengen.

Een gemeenschappelijke fantasiewereld als een manier wordt gezien om te binden met anderen, zoals bij het spel World of Warcraft met een online community waar in december 2008 11,5 miljoen met elkaar interacterende gebruikers waren geregistreerd. Trainers en managers staan voor de uitdaging om fantasie met werkelijkheid te combineren voor hun werknemers. Dit is terug te zien in de werkomgevingen die bepaalde technologiebedrijven als Google en Microsoft hebben, waar werknemers zich terug kunnen trekken in een ontspannende activiteit als indoor bergbeklimmen of computerspellen kunnen spelen als onderbreking van hun werk. Fantasiegebaseerde Serious Games zouden een geaccepteerde en boeiende methode zijn om werknemers op te leiden en begeleiden.

Technology as Friend vs. Technology as Foe

De generaties die aan de Games Generation voorgaan, zijn veelal niet gewend aan moderne technologie. Het zit in hun gedachtegang om nieuwe dingen te schuwen, wat voor een belangrijk deel komt door de introductie van nieuwe technologie op latere leeftijd dan de Games Generation, die technologie van jongs af aan heeft gebruikt en er bekend mee is.
De Games Generation ziet technologie als een vriend, die er altijd is als er behoefte is aan spelen, ontspanning en plezier. Voor velen uit deze generatie voelt het bezitten van of toegang hebben tot een computer als geboorterecht, iets dat iedereen zou moeten bezitten. Deze generatie staat ook bekend om de generatieomslag in technische kennis, waarbij ouders advies inwinnen bij hun kinderen over het gebruik van hun eigen dure apparatuur. Trainers en managers kunnen werknemers uit oudere generaties stimuleren om hun eigen elementen van technologie te maken zoals computerapplicaties, webpagina's, modellen en structuren en te gebruiken voor hun eigen, voor hun generatie zinnige doeleinden, of ze te laten werken in een team dat eraan bezig is. Dit bevordert de toegankelijkheid van technologie bij de oudere generaties.

"Attitude"

Als toevoeging tot al de bovenstaande veranderingen in denkwijzen met betrekking tot werken en gebruik van moderne media, heeft de Games Generation volgens Prensky een hoop 'attitude', een directe, vaak sarcastische kijk op dingen. Ze rekenen mensen direct af op hun tekortkomingen en zijn mondiger dan voorgaande generaties. Om met deze generatie te communiceren moet men direct zijn en geen blad voor de mond nemen.

Al deze cognitieve verschillen, die gevolgen zijn van jaren van 'nieuwe media socialisatie', zoals Prensky het beschrijft, hebben veel effect op de manier van leren en vaardigheden van de Games Generation. Een logisch gevolg hiervan is, dat men betere manieren zoekt om leden van deze generatie te benaderen. Hoewel het niet de enige manier is, zijn computerspellen en videospellen veelgebruikte informatiestructuren waar de Games Generation bekend mee zijn en die in staat zijn om de veranderde leerbenodigdheden en eisen aan het leerproces te ondervangen. Dit is een van de redenen dat digitale leerwijzen, en Serious Gaming in het bijzonder, steeds vaker wordt toegepast en tot bloei komt.

Hoofdstuk 4: De testapplicatie

"It is easier to change the specification to fit the program than vice versa."
- Alan Perlis

De essentie van requirements modeling

De hoofdvraag van dit artikel luidt: "Hoe kan de essentie van requirements modeling worden ingekaderd in een spelachtige procedure?". Wat is die essentie dan precies?
Van alle toepassingen van Serious Gaming binnen Requirements Engineering wordt er alleen gekeken naar de requirements modelingfase. In deze fase staat, zoals in hoofdstuk 2 vermeld, het beschrijven van de essentiële entiteiten in het gemodelleerde proces, hun eigenschappen en hun onderlinge relaties in tekstuele en grafische vorm centraal.

Requirements Engineering en Serious Gaming

De voornaamste doelen van Serious Gaming bij Requirements Engineering zijn het motiveren van de deelnemers en het op spelenderwijs boven water krijgen van requirements. Als men kijkt naar de stappen van het Requirements Engineeringproces zoals beschreven in hoofdstuk 3, zou Serious Gaming het meest van pas komen in de fasen elicitation en elaboration, waarbij requirements worden blootgelegd, verfijnd en gedocumenteerd.
Er zijn verschillende redenen waarom het proces van Requirements Engineering goed te vatten is in een Serious Game. Hierbij kan onderscheid worden gemaakt tussen voordelen bij het aanleren van vaardigheden binnen het vakgebied van Requirements Engineering en voordelen bij het uitvoeren van het Requirements Engineeringproces zelf.

Voordelen bij het aanleren van Requirements Engineering

Behoefte aan training in complexere situaties

RE is een complex vakgebied waarin 'simpele' manieren van methoden aanleren, zoals het vergaren van kennis van een PowerPoint presentatie of frontaal leren, zoals van een flipover, niet afdoende zijn. De theoretische kennis moet in de praktijk worden toegepast om het proces te begrijpen en goed toe te kunnen passen, wat met een Serious Game zeer goed kan.

Daarnaast verlopen trainingen van gangbare leermethoden tot nu toe vooral sequentieel volgens bepaalde stappen. Bij zulke leermethoden zijn situaties waarbij uitzonderingen optreden, moeilijk te oefenen. Serious Gaming biedt een medium om dat soort uitzonderingen op interactieve wijze te analyseren en erop in te grijpen. [**Verbraeck**]

Behoefte aan training op de werkplaats

In het huidige economische klimaat wordt gekeken naar elke uitgave en wordt er op elke mogelijke uitgavenpost gekort, waaronder reistijd en -kosten. Het trainen van requirements engineers en het uitvoeren van het Requirements Engineeringproces vergt vaak veel tijd, omdat alle betrokkenen samen moeten komen om een activiteit uit te kunnen voeren.

Met een Serious Game voor het trainen van requirements engineers kunnen ze vaardigheden op interactieve wijze op hun werkplaats aanleren, zonder daarvoor de deur uit te moeten. Een bijkomend voordeel is dat ze zichzelf zo vaak kunnen trainen als ze willen in tegenstelling tot de incidentele keren dat de requirements engineers deelnemen aan een workshop of seminar.
Een Serious Game die het Requirements Engineeringproces faciliteert, neemt de reistijd weg die de gebruikers van het proces zouden moeten spenderen, omdat ook zij de Serious Game op hun eigen locatie kunnen uitvoeren. [**Verbraeck**]

Voordelen bij het uitvoeren van het Requirements Engineeringproces
Interactie brengt gebruikers nieuwe inzichten
Bij het uitvoeren van het RE-proces worden veel vraag- en antwoordgesprekken gehouden, waarbij de gebruikers simpelweg antwoord geven op de gestelde vragen. Deze statische manier van informatie vergaren verkleint de hoeveelheid informatie die de requirements engineer uiteindelijk tot zijn beschikking zal hebben omdat de gebruikers proberen specifiek antwoord te geven op de gestelde vraag en alleen op dat antwoord gefocust zijn. Door ze op interactieve wijze deel uit te laten maken van het RE-proces zelf, laat men de gebruikers actief meedenken, wat tot nieuwe inzichten in het bedrijfsproces dat geautomatiseerd moet worden kan leiden. Gedurende het spelen van een Serious Game doen gebruikers abstracte kennis op van het te doorlopen proces en door trial-and-error bouwt de gebruiker reflectievaardigheden op. De ontwikkelde cognitieve strategieën helpen de gebruiker verbanden te leggen tussen reeds bekende en nieuwe informatie en zo tot een completer overzicht van requirements te leiden. [**Chen**]

Motivatie van gebruikers
Een van de grootste uitdagingen bij het uitvoeren van het RE-proces is het motiveren van de gebruikers om zo compleet mogelijke requirements te formuleren. De gebruikers, vaak domeinexperts, middelmanagement of eindgebruikers, werken meestal samen met de requirements engineer in opdracht van hun leidinggevenden. De motivatie van deze gebruikers is niet altijd even hoog omdat ze tijd vrij moeten maken om met de requirements engineer samen te komen om een complex proces te doorlopen dat voor een buitenstaander onbegrijpelijk kan lijken. Ook het opbouwen van een vertrouwensband tussen requirements engineer en gebruiker kan moeizaam verlopen.

Deze factoren maken het bedenkelijk dat de gebruikers beknopte informatie verstrekken aan de requirements engineer, om zo snel mogelijk weer hun eigen gang te kunnen gaan. Dit kan leiden tot incomplete of verkeerde requirements.
Het speelse en toegankelijke karakter van een Serious Gaming methode kan hierin verandering aanbrengen. Door het proces leuk en aantrekkelijk vorm te geven kan men de gebruikers motiveren zo veel mogelijk mee te werken. Doordat de Serious Game leuk is om te spelen, motiveren gebruikers zichzelf om de doelen van het spel zo goed mogelijk te voltooien.

Het RE-proces op de werkplaats
Zoals eerder werd beschreven, vergt het RE-proces veel tijd en zijn er meerdere bijeenkomsten met gebruikers benodigd. Een Serious Game kan door de gebruikers worden gespeeld op locatie, waarna de verzamelde informatie aan de requirements engineer kan worden geleverd. Bij gebruik van een efficiënt ontworpen Serious Game krijgt de requirements engineer dezelfde informatie als bij een fysieke bijeenkomst, terwijl zowel gebruiker als requirements engineer op zijn eigen werkplek kan blijven.

Eenvoudig functies toepassen op het RE-proces
Doordat een Serious Game een digitale omgeving is, kunnen functies als logboeken bijhouden, statistieken berekenen over antwoorden en vergelijkingen tussen iteraties van de Serious Game maken, gemakkelijk worden uitgevoerd en eventueel zelfs in de applicatie worden ingebouwd. Dit biedt de requirements engineer een extra dimensie aan informatie die kan worden gebruikt om tot een zo compleet mogelijk systeemontwerp te komen.

Het ontwerpen van een Serious Game
Bij het ontwerpen van een Serious Game moet worden gekeken naar het doel van de Serious Game. Er zijn veel doelen die kunnen worden bereikt door middel van een Serious Game, waaronder performanceverbetering, het testen van competenties bij gebruikers, het verhogen van bewustzijn dat bepaalde rollen in een organisatie belangrijk zijn, het selectieproces bij sollicitaties, verhogen van begrip bij klanten en partners van een organisatie, promotie, motivatie, best practice overdracht van kennis en het laten samenwerken van gebruikers in een bepaald proces. [**Corti**]

Bij het ontwerpproces kan men rekening houden met bepaalde aspecten van Serious Gaming, die het bereiken van deze doelen vergemakkelijken. Veel van deze aspecten hebben te maken met het karakter van games en hoe games in elkaar zitten.

Doelen
Zowel in games als in Serious Games zijn de doelen van elke iteratie van de game duidelijk, er is niet wezenlijk veel verschil tussen ‘red de prinses van de grote aap’ en ‘verdubbel de omzet over 3 jaren’. Deze doelen stellen de gebruiker in staat om na te denken over de aanpak van een probleem.

Regels
In games zijn er duidelijke regels die aan de gebruiker worden voorgelegd. Vaak zijn deze in de vorm van deductieregels of ‘als…, dan…’ regels. Deze regels kunnen inherent zijn aan het gesimuleerde systeem, of ingebouwde regels om de gebruiker de juiste richting op te wijzen.

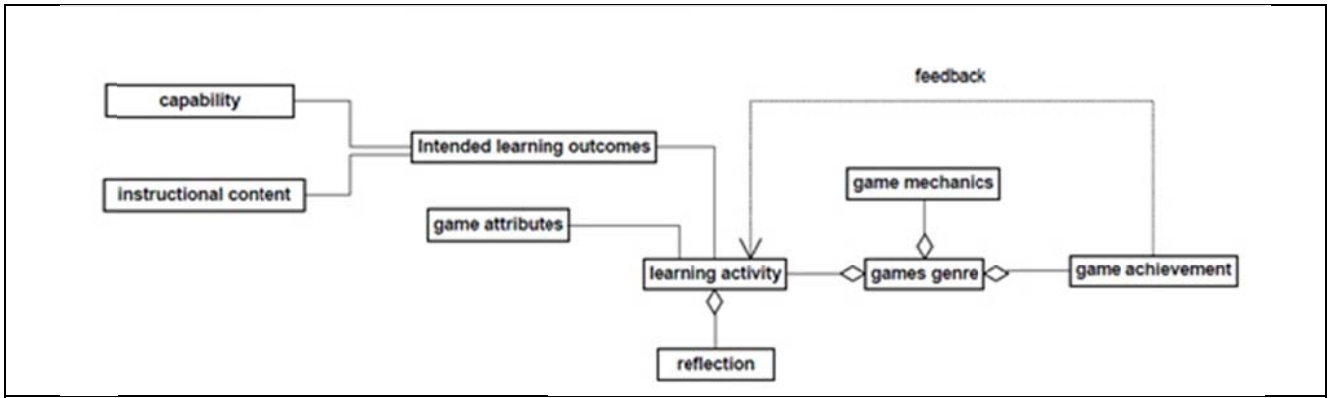
Herhaalbaarheid
De acties die een gebruiker van de Serious Game kan ondernemen, kunnen deel uitmaken van een bepaalde strategie of benadering van het probleem. Bij mislukking kan de gebruiker

zijn strategie aanpassen en de nieuwe strategie opnieuw toepassen op het probleem. Hierdoor krijgt de gebruiker feedback over zijn werkwijze, wat weer leidt tot nieuwe kennis.

Omgevingen
Door gebruik van moderne systemen kan een Serious Game worden gegenereerd die grafisch zeer realistisch is en complexe werking kent. Door een realistische simulatie van een bestaande omgeving kunnen gebruikers grenzen, mogelijkheden en de probleemruimte zelf verkennen.

Uitdaging
Games eisen van de gebruiker een niveau van cognitieve toepassing die verder gaat dan alleen het lezen van tekst en het opnemen van feiten. De gebruiker wordt aangemoedigd om creatief te denken, problemen op te lossen, op onderzoek uit te gaan en trial and error toe te passen, allemaal zaken die in een Serious Game tot zijn recht komen en bijdragen aan de oplossing van een probleem.

Framework voor Serious Games
Bij het vergelijken van Serious Games met reguliere computergames, waarbij beiden beschikken over een verhaal, vormgeving en software, is het de toevoeging van *pedagogiek* die een game een Serious Game maakt. Pedagogiek omvat de activiteiten van instrueren of aanleren, met als gevolg het verkrijgen van kennis en het aanleren van vaardigheden. [**Susi**] Amri Yusoff et al. hebben een conceptueel framework ontworpen dat elementen bevat die in een Serious Game zouden moeten zijn verwerkt. Dit framework, dat ontworpen is als effectief model voor het leerproces in Serious Games, combineert pedagogiek met de eerder genoemde aspecten van Serious Gaming. [**Yusoff**]



Figuur 6. Serious Game Framework. Uit [Yusoff].

Capability staat voor de cognitieve en motorische capaciteit van de gebruiker van de Serious Game, die hij ontwikkelt door de game te spelen.
Instructional content zijn de feiten, procedures, concepten en principes van het onderwerp die de gebruiker als doel heeft te leren.
Intended learning outcomes zijn de doelen die men bereikt door de Serious Game te spelen, wat een combinatie is van de capaciteit van de gebruiker samen met het onderwerp van de game.

In dit geval is het 'leer een requirementsspecificatie in de vorm van een Use Case op te stellen'.
Game attributes omvatten het geheel van aspecten van de Serious Game die de kennisoverdracht en het aanleren van vaardigheden moet ondersteunen. Hierbij kan men denken aan op incrementele wijze uitleggen van de onderwerpstof aan de gebruiker, oefeningen voorschotelen die de gebruiker in staat stelt om zijn nieuwe kennis toe te passen en directe feedback vanuit de Serious Game naar de gebruiker.
Learning activity is de activiteit die de gebruiker uitvoert om kennis op te doen of een vaardigheid aan te leren. Hoe effectiever deze learning activities zijn, hoe betrokkener en geïnteresseerder de gebruiker raakt met de speelwereld en hoe meer kennisoverdracht er plaats vindt. De learning activities moeten een gebruiker prikkelen om meer te willen leren dan dat hij op dat moment weet en kan.
Reflection is het moment waarop de gebruiker nadenkt over het nut van de *learning activities* die hij heeft ondernomen en de strategie bepaalt waarmee de volgende stap wordt ondernomen, zonder uit de speelomgeving te stappen. Het is belangrijk dat er voldoende feedback aan de gebruiker wordt geleverd teneinde goed te kunnen reflecteren op de situatie.
Games genre is het type of categorie spel dat gespeeld wordt. Dit kan variëren van strategiespellen tot simulatiespellen tot actiespellen en het kan ook een combinatie ervan zijn.
Game mechanics zijn de spelregels die gelden in de speelwereld.
Game achievement is het niveau dat de gebruiker heeft bereikt tijdens het spelen van de Serious Game. Dit niveau kan worden aangegeven door spelscores, een hoeveelheid virtuele grondstoffen of een tijdsmaat. De game achievement geeft de gebruiker een gevoel van beloning en tegelijkertijd vormt het een goede maatstaf om de gebruiker te beoordelen op prestatie.

De applicatie en methode
In de testapplicatie doorloopt een gebruiker het proces van requirements elicitation en elaboration. Het doel van deze testapplicatie is een 'proof of concept' te tonen waarin twee belangrijke rollen worden gedemonstreerd die Serious Gaming binnen het proces van Requirements Engineering kan vervullen; **motivatie** en **interactie**.

De bedoeling is dat de gebruiker eerst een papieren versie invult van een Use Case, om bekend te raken met het proces van RE-gerelateerde velden invullen. Hierbij raakt de gebruiker gewend aan de soort input die het proces van Requirements Engineering vergt.

Daarna speelt de gebruiker de Serious Gaming testapplicatie als digitale vervanging van hetzelfde proces dat hij net op papier heeft doorlopen. Door middel van interactie met een te realiseren systeem, in dit geval versimpeld als auto, construeert de gebruiker een lijst van

functionele requirements en een Use Case. Dit gebeurt door middel van tekstuele invoer door de gebruiker, die direct feedback op zijn input krijgt.

Use Cases

Bij het implementeren van de testapplicatie ligt de focus op het genereren van Use Cases. Use Cases zijn veelgebruikte, scenario gebaseerde modellen die het gedrag van een informatiesysteem beschrijven door middel van het stapsgewijs in kaart brengen van de gebruikelijke gang van zaken, de zogenaamde Basic Course of Events. Hiernaast worden onder andere de alternatieven op die stappen, de te ondernemen acties indien er een fout optreedt en de deelnemende actoren opgenomen in de Use Case.
Er is gekozen voor een Use Case, omdat dit een zeer toegankelijk informatiemodel binnen requirements modeling is. Door de simpele opzet, een formulier met verschillende invulvelden die door de requirements engineer worden ingevuld, kan iedere belanghebbende zien wat er is genoteerd en krijgt men daardoor directe feedback. Daarnaast neemt het invullen van een Use Case aanzienlijk minder tijd in beslag dan bijvoorbeeld een workshop of enquête. Hoewel enkel tekstueel, geeft het een summiere beschrijving van een proces dat wordt doorlopen en biedt het een basis voor verdere modellen, die gedetailleerder en technischer van aard zijn.

De complete lijst van invulvelden kan worden aangepast naar believen. Hieronder staat een lijst met mogelijke invulvelden. Kulak en Guiney [**Kulak**] stellen, dat er zo min mogelijke invulvelden moeten worden gebruikt om de Use Case zo duidelijk mogelijk te maken zonder redundanties.

Invulveld	**Uitleg**
Use Case Name:	Naam van de Use Case.
Summary:	Korte samenvatting van het proces welke wordt gerepresenteerd in de Use Case.
Basic Course of Events (BCoE):	Stapsgewijze toelichting van het gerepresenteerde proces, zonder fouten of misstappen. Deze stappen zijn abstracte weergaven zonder concrete invulling. Er wordt verduidelijkt *wat* er moet gebeuren (bijvoorbeeld: 'De opdracht moet bevestigd worden') en niet *op welke manier* (bijvoorbeeld: 'De gebruiker klikt op de OK knop') dat moet.
Alternative Paths:	Alternatieve paden die gevolgd worden indien er in één van de stappen in de BCoE een wijziging van de verwachte situatie optreedt (bijvoorbeeld: input in de vorm van een string in plaats van een integer). Er wordt aangegeven in welke stap van de BCoE deze wijziging zich voordoet.
Exception Paths:	Alternatieve paden die gevolgd worden indien er in één van de stappen in de BCoE een fout optreedt (bijvoorbeeld: de slagboom zit vast). Er wordt aangegeven in welke stap van de BCoE deze fout kan voorkomen.
Triggers:	Acties die deze Use Case in gang zetten, het antwoord op de

	vraag "Wanneer of waarom zetten de actors deze Use Case in gang?"
Assumptions:	Aannames die gelden ten tijde van deze Use Case.
Preconditions:	Precondities die waar moeten zijn voordat deze Use Case wordt doorlopen.
Postconditions:	Postcondities die waar moeten zijn nadat deze Use Case wordt doorlopen.
Business Rules:	Business Rules die van toepassing zijn op deze Use Case (bijv. het pakket mag niet meer dan 5 kilo wegen).
Author:	Auteur van de Use Case.
Date:	Datum.

De testapplicatie

De testapplicatie waarin de voorafgaande theoretische context samenkomt, is een simpele Serious Game die geprogrammeerd is in een combinatie van PHP, JavaScript en HTML. Een korte uitleg:

Deze simpele applicatie zal de gebruiker de **opdracht** geven om een bepaald proces in korte stappen te beschrijven, in dit geval '*het starten van een auto en wegrijden'*. Hiertoe krijgt hij enkele invoervelden voor tekst tot zijn beschikking. Deze invoervelden worden automatisch gevalideerd doordat de applicatie de input continu scant.

Daarnaast wordt er een zogenaamde **'to-do' list** bijgehouden, een checklijst met enkele steekwoorden die de gebruiker helpen om deze stappen te formuleren.

Verder wordt er een **score** bijgehouden die die gebaseerd is op compleetheid van de checklijst en de tijd die de speler nodig heeft om deze checklijst te completeren. Deze score bedraagt

100 x aantal antwoorden goed + (300 – aantal seconden bezig)

Waarbij een antwoord goed wordt gerekend als het overeenkomt met de steekwoorden die op de to-do lijst staan. Er wordt ook een fictieve highscore weergegeven naast de huidige score, als motivatiemiddel.

Door te drukken op **de knop 'Gebruik deze data'** wordt er door de testapplicatie een Use Case gegenereerd op basis van de door de gebruiker ingegeven data, aangevuld met enkele andere punten zoals de naam van de gebruiker en de datum. Omdat het in dit artikel slechts een proof of concept betreft, worden alleen de kopjes 'Use Case Name', 'Summary', 'Basic Course of Events', 'Business Rules', 'Author' en 'Date' ingevuld.

Interactie

Een belangrijke functie van een Serious Game is het laten interacteren van de gebruiker met het systeem. Interactie is de mate waarin de Serious Game aan de gebruiker om respons en

betrokkenheid vraagt. Ouderwetse, statische applicaties stellen vragen over een onderwerp, waarop de gebruiker antwoord geeft. Deze vragen zijn doelgericht, waardoor de gebruiker gelimiteerd is in antwoord geven. Daarnaast hoeft de gebruiker niet 'out of the box' te denken, omdat daartoe geen opdracht of mogelijkheid wordt gegeven.

De combinatie van de to-do list en de scoretelling speelt een grote rol daarin. De gebruiker wordt een uitdaging geboden om de to-do list zo zorgvuldig mogelijk af te werken en vereist daarbij respons en betrokkenheid van de gebruiker. De input van de gebruiker kan in de applicatie meteen worden gevalideerd om te kijken of de ingegeven woorden overeenkomen met de woorden uit de to-do lijst. Bij een overeenkomst kleurt het desbetreffende woord in de to-do lijst groen, zodat de gebruiker meteen positieve feedback krijgt over zijn input. Doordat de gebruiker zijn input nog kan wijzigen voordat de input definitief wordt ingegeven, stimuleert de Serious Game de gebruiker bij het ontwikkelen van een reflectiekader. Een reflectiekader vormt in dit geval het kader van woorden die hij kan gebruiken voor een correcte input. In een toekomstige opdracht kan de gebruiker teruggrijpen op dit reflectiekader om te beoordelen of hij een juiste input heeft gegeven of niet, zelfs zonder dat de applicatie dat aangeeft. Bij een verdere uitwerking van dit idee kan men denken aan een lijst van trefwoorden die synoniemen zijn voor het bijbehorende woord uit de to-do lijst.

De scoretelling werkt ook als instrument, aangezien die punten verstrekt worden naar gelang de to-do list afgewerkt is. Iteratie, het herzien van bestaande artefacten met als doel ze te verbeteren is een veelvuldig gebruikt concept in methode engineering [**Hoppenbrouwers**]. Dit begrip komt erop neer dat gebruikers de input blijven nakijken en verbeteren zolang de to-do lijst nog niet compleet is afgewerkt. Elke iteratie, elke keer dat de gebruiker zijn input nakijkt, wordt het antwoord verfijnd, wat uiteindelijk leidt tot een beter eindresultaat.

Doordat de gebruiker in de Serious Game met het systeem interacteert en die interactie naar twee kanten verloopt, zowel van de gebruiker naar systeem en andersom, vindt er feedback plaats, die volgens het eerder genoemde framework van Amri Yusoff essentieel is om een goed leersysteem neer te zetten. Deze feedback zorgt ervoor, dat de gebruiker de *learning activity*, in dit geval het doorlopen van het RE-proces, zo goed mogelijk kan uitvoeren en een reflectiekader op kan bouwen.

Motivatie

Een andere belangrijke functie van een Serious Game is het motiveren van de gebruiker. Een gemotiveerde gebruiker is enthousiast, gefocust en betrokken. Hij is geïnteresseerd in wat hij doet en beleeft er plezier aan en hij doet zijn best. Dit gedrag komt vanuit de gebruiker zelf, gedreven door zijn eigen voorkeuren en ideeën en niet door een externe invloed. Er kan onderscheid worden gemaakt tussen extrinsieke en intrinsieke motivatie.

Intrinsiek

Intrinsieke motivatie vindt plaats wanneer een persoon een wil of drang heeft om bepaalde handelingen te verrichten waarmee tot een zeker doel wordt gekomen [**Garris**]. De gebruiker wordt daartoe intern geprikkeld door meerdere invloeden, zoals het interessant of plezierig vinden van een activiteit.

De Serious Game die in de testapplicatie wordt uitgewerkt, zal een mooi uiterlijk hebben, een duidelijke en toegankelijke interface. Deze factoren zorgen voor een interessant ogende applicatie, waarbij de drempel om eraan deel te nemen, laag is; de gebruiker kan meteen beginnen met het invoeren van tekst. De applicatie nodigt de gebruiker als het ware uit om te worden gebruikt.

Daarnaast wordt er in het spel een score bijgehouden. Deze score moedigt de gebruiker aan om het juiste antwoord te geven, maar biedt ook een motivatiemiddel om het antwoord te blijven verfijnen tot het maximaal aantal te behalen punten bereikt is. De highscore, een fictieve score die naast de huidige score van de gebruiker zelf wordt weergegeven, is met voorbedachten rade een lage score. Doordat de gebruiker zijn huidige score kan zien en met de highscore kan vergelijken, vormt dit een extra uitdaging om de highscore te verbeteren.

Extrinsiek

Extrinsieke motivatie vindt plaats wanneer een persoon een activiteit uitvoert als middel om een doel te bereiken. De gebruiker ondergaat een activiteit omdat hij de uitkomst ervan op waarde schat [**Garris**].

Domeinexperts worden vaak ingeschakeld bij Requirements Engineering. Ze hebben extensieve kennis van het in te perken kennis- of business domein, die ze kunnen overbrengen aan de andere stakeholders in het project. Aan het begin van een project worden ze ingezet om een situatieschets te vormen, waarbij in eerste instantie begrippen worden geïntroduceerd, die later worden verfijnd.

Een voorbeeld is een informatiesysteem dat in een ziekenhuis wordt aangelegd om een medisch dossier van patiënten bij te houden; een mogelijke requirement is dat per patiënt, zijn of haar aandoeningen moeten worden bijgehouden. In eerste instantie hoeven software engineers alleen te weten dat er verschillende soorten aandoeningen zijn waaraan een patiënt kan lijden. Zo weten ze dat de initiële datastructuur in elk geval een manier moet hebben om per patiënt de aandoeningen bij te kunnen houden. In een later stadium worden deze eisen verfijnd naar verschillende soorten aandoeningen.

De opdracht in de testapplicatie, samen met de to-do lijst is een metafoor voor extrinsieke motivatie; de gebruiker wilt de opdracht zo goed mogelijk uitvoeren en houdt zich daarbij aan de to-do lijst. Hij vult de stappen in en probeert de to-do lijst zo goed mogelijk af te werken omdat hij het uiteindelijke doel wilt bereiken: het vervullen van de opdracht.

Hoofdstuk 5: Analyse

"If you automate a mess, you get an automated mess."
- Rod Michael

De testapplicatie bestaat uit drie webpagina's, index.php, ucsg.php en generate.php. Allen zijn te vinden op http://sg.zoja.nl.

Uitleg over de applicatie

Index.php is de introductiepagina van de testapplicatie (zie fig. 7). De gebruiker wordt hier verwelkomd met een afbeelding van een mooie auto en een vrolijke tekst waarin wordt uitgelegd dat de gebruiker een spel zal spelen dat een Use Case gaat genereren. Ook wordt er gemeld dat er een score aan de gebruiker zal worden gegeven, die hij of zij kan gebruiken om de prestaties in het spel te vergelijken met die van anderen. De tekst op de webpagina:

"Welkom bij The Car Game!

Dit spel laat jou als gebruiker op speelse wijze deel uitmaken van het modelleren van een informatieproces. Het proces dat gemodelleerd wordt is 'in een auto stappen en wegrijden', met alle bijbehorende controles en handelingen!

Jij wordt uitgedaagd om dit zo compleet mogelijk te modelleren en dit in een zo kort mogelijk tijdsbestek te doen. Je zult een score ontvangen die aangeeft hoe goed je hebt gepresteerd! Vergelijk het met de scores van anderen en kroon jezelf tot supermodelleerder!

Het resultaat van dit spel zal een Use Case opleveren, een informatiemodel zoals die je net hebt ingevuld. Een Use Case is een informatiemodel, waarin alle stappen van een bepaald proces zijn vastgelegd, met hun uitzonderingen, pre- en postcondities et cetera.

Wees echter niet bang, er wordt geen technische kennis van je verwacht!

Als je denkt dat je klaar bent voor de uitdaging, druk dan op 'Verder'!

Veel plezier!"

Figuur 7. Screenshot van index.php.

UCSG.php is de vervolgpagina erop; hier begint het echte spel. Het scherm is in drie delen opgesplitst in drie verticale kolommen.

Linkerkolom
De linkerkolom geeft de score aan van de gebruiker, evenals uitleg over hoe de score tot stand komt (zie fig. 8a). Tot slot staan er ook twee fictieve highscores in de kolom, zodat de gebruiker daaraan zijn score kan meten.
"*Score*
[Score van de gebruiker]

Highscores:

763.8 – Jeffrey
629.9 - Stijn

Je score wordt automatisch bepaald door de compleetheid van de To-Do Lijst in je invoer en de tijd die je erover doet om de To-Do Lijst te completeren.

Items in je To-Do Lijst kleuren groen als je een juiste invoer geeft."

Middenkolom
De middenkolom bevat de invoervelden voor de gebruiker. Er wordt gevraagd om de naam van de gebruiker, de datum en een belangrijke regel die geldt bij het proces van een auto starten en ermee wegrijden. Deze belangrijke regel is in de 'echte wereld' een business rule, een regel die geldt in de werkomgeving van het te modelleren proces. De bedoeling is dat er bij dit veld 'De gebruiker moet in het bezit van een rijbewijs zijn" wordt ingevuld, wat ook wordt aangegeven met een hint. Verder moet de gebruiker een reden invoeren waarom hij of zij het proces van een auto starten en ermee wegrijden wilt beginnen, wat in RE-termen een

'trigger' heet, een voorval of oorzaak die het te modelleren proces in gang zet.
Daaronder wordt de daadwerkelijke opdracht uitgelegd; het per stuk invoeren van de stappen in het proces (zie fig. 8b). Hiervoor is een groot tekstinvoervlak beschikbaar van meerdere regels. De bedoeling is, dat de gebruiker een complete Basic Course of Events invoert. Door het gebruik van een score die bijgehouden wordt en het feit dat de input constant wordt gevalideerd aan de hand van de To-Do lijst in de rechterkolom, is de gebruiker gemotiveerd om een zo compleet mogelijk antwoord te geven in zo kort mogelijke tijd. Onderaan de middenkolom is er een knop zichtbaar, die het parseren van deze Serious Game in gang zet. Na een druk op de knop wordt de inhoud van de inputvelden naar generate.php gestuurd, die de gegevens parseert in een Use Case.

"Je naam:
De datum:
Wat is de belangrijkste regel die geldt bij het proces 'een auto starten en wegrijden'? (Moet je iets in je bezit hebben om te mogen rijden?)
Wat is de reden dat je het proces 'een auto starten en wegrijden' wilt beginnen? (Waarom wil je in je auto rijden?)

Opdracht:

Geef in korte stappen aan wat er moet gebeuren bij het proces van in een auto stappen en wegrijden. Doe dit volgens de manier die je bij rijlessen leert, dus met controle en verstellen van alle noodzakelijke zaken. Deze korte stappen worden vooraf gegaan aan het nummer van de stap en afgesloten met een enter, dus
1. Beschrijving van stap één.
2. Beschrijving van stap twee.

Vul de stappen in van het proces 'een auto starten en wegrijden'.

Invoer

Als je klaar bent en je antwoorden hebt gecheckt, druk dan op de knop 'Genereer Use Case' onderaan de pagina om je antwoorden definitief op te sturen."

Rechterkolom
De rechterkolom bevat de To-Do Lijst, een lijst met trefwoorden die de gebruiker kan gebruiken als checklist (zie fig. 8a en 8c). Elke input van de gebruiker wordt tegen gevalideerd aan de hand van de To-Do Lijst. Voert de gebruiker een term of zin in die één van de woorden uit de To-Do Lijst bevat, dan kleurt het desbetreffende woord in de lijst groen. Zo krijgt de gebruiker directe feedback op zijn input.

"To-Do Lijst:

1. Bandenspanning

2. Stoel

3. Spiegels

4. Veiligheidsriem

5. Starten

6. Spiegels kijken

7. Schakelen

8. Wegrijden"

Figuur 8a. Screenshot van de bovenkant van ucsg.php.

Figuur 8b. Screenshot van de onderkant van ucsg.php.

Figuur 8c. Screenshot van de To-Do Lijst in reguliere staat (links) en na correcte input van de gebruiker (rechts).

Generate.php is de laatste pagina van de Serious Game. Op deze pagina wordt de data van de voorafgaande pagina geparseerd tot een Use Case (zie fig. 9). Hoewel het geen volledige Use Case is, is dit uiteindelijke product van de testapplicatie een informatiemodel dat de gebruiker door het spelen van de Serious Game heeft gegenereerd, met zo min mogelijke hulp van buitenaf. Twee zaken zijn van tevoren ingevuld, de Use Case Name, namelijk 'een auto starten en wegrijden' en een Summary, een korte samenvatting van het proces. De velden Actors, Triggers, Basic Course of Events, Business Rules, Author en Date heeft de speler op autonome wijze zelf ingevuld.

"Gefeliciteerd! Aan de hand van The Car Game heb je een heuse Use Case gegenereerd. Je eindscore is [Score van de gebruiker].

Use Case Name: Een auto starten en wegrijden

Summary: De auto wordt op meerdere onderdelen gecheckt en kan daarna zonder moeite worden weggereden.

Actors: De bestuurder genaamd [De naam van de gebruiker]

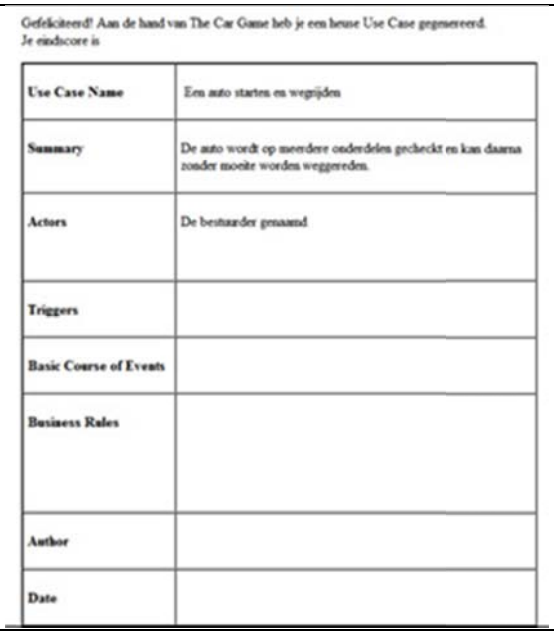
Triggers:

Basic Course of Events:

Business Rules:

Author:

Date:"



Gefeliciteerd! Aan de hand van The Car Game heb je een heuse Use Case gegenereerd.
Je eindscore is

Use Case Name	Een auto starten en wegrijden
Summary	De auto wordt op meerdere onderdelen gecheckt en kan daarna zonder moeite worden weggereden.
Actors	De bestuurder genaamd
Triggers	
Basic Course of Events	
Business Rules	
Author	
Date	

Figuur 9. Screenshot van generate.php

De tests

Deze testapplicatie moest worden getest om te kijken of het geslaagd is als proof of concept. Hiervoor heb ik drie proefpersonen bereid gevonden om een instructie aan te horen over Use Cases. Alle velden van de Use Case werden behandeld volgens de uitleg in [**Kulak**].

Na de instructie kregen de proefpersonen de opdracht om eerst een papieren versie van een Use Case in te vullen om bekend te raken met de vragen die er tijdens een Requirements Engineeringsessie worden gevraagd. Het onderwerp van de Use Cases mochten de proefpersonen zelf bedenken, als ze maar een proces kozen dat ze goed kenden. Geen van de drie proefpersonen had van tevoren kennis van RE.

Hierna hebben ze de testapplicatie doorlopen en afsluitend een enquête ingevuld met enkele vragen over de gebruikte methoden.

Deze vragen waren:
i. *Wat vond je van het invullen van de papieren Use Case?*
ii. *Wat vond je van het spelen van de Serious Gaming Use Case?*
iii. *Wat vond je het voornaamste verschil tussen beide soorten Use Case methodes?*
iv. *Vond je dat je met de Serious Game meer interactie had met het proces dan met de papieren Use Case? Motiveer je antwoord.*
v. *Vond je dat je met de Serious Game meer gemotiveerd werd om een correct antwoord te bedenken dan met de papieren Use Case?*

Gebruiker A is een 25-jarige student Commerciële Economie aan de Hogeschool Arnhem en Nijmegen. Hij koos bij het invullen van de eerste Use Case het proces *'Het zetten van een kopje koffie'*. Het invullen van de complete Use Case op papier duurde 17.36 minuten. Het invullen van de Serious Gaming Use Case kostte hem 4.25 minuten en hij scoorde daarbij 735 punten.

Zijn antwoorden:
i. Saai, langdradig.
ii. Leuk, uitdagend.
iii. De Serious Gaming Use Case was leuker, je had niet echt het gevoel dat je een opdracht aan het uitvoeren was maar meer een spel aan het spelen.
iv. Jazeker, je denkt meer na over de antwoorden die je geeft. Je krijgt ook meteen feedback.
v. Ja, je wilt een goede score neerzetten.

Gebruiker B is een 22-jarige studente Communicatie aan de Hogeschool Arnhem en Nijmegen. Zij koos bij het invullen van de eerste Use Case het proces '*een fietsband plakken'*. Over het invullen van de papieren Use Case deed ze 14.45 minuten. Het invullen van de Serious Gaming Use Case kostte haar 4.08 minuten en scoorde daarbij 752 punten.

Haar antwoorden:
i. Niet bijzonder interessant.
ii. Leuk, het was steeds zoeken naar de juiste invullingen. Het was uitdagend, maar wel gemakkelijk.
iii. De interactie die je met een computerspel hebt is veel meer dan met een simpel formulier waarop je wat invult.
iv. Zoals ik al eerder schreef, met een computerspel heb je het idee dat je meer vrijheid hebt in het antwoorden dan met een formuliertje. Vooral het lijstje met dingen die je nog moest doen was erg handig, zoiets zou bij dat formulier ook van pas komen.
v. Ja, er werd meteen gecontroleerd en je wist meteen of het antwoord goed of niet goed was.

Gebruiker C is een 24-jarige student Bedrijfscommunicatie aan de Radboud Universiteit Nijmegen. Hij koos bij het invullen van de eerste Use Case het proces '*noedelsoep maken'*. Over het invullen van de papieren Use Case deed hij 8.55 minuten. Het invullen van de Serious Gaming Use Case kostte hem 3.28 minuten en scoorde daarbij 792 punten.

Zijn antwoorden:
i. Saai, niet boeiend om eerlijk te zijn.
ii. Op zich wel leuk, weer eens wat anders.
iii. De tweede was een stuk interessanter dan de eerste, simpelweg maar wat invullen zonder dat je weet wat eruit komt is niet heel boeiend, maar ook een minder saaie omgeving helpt wel de aandacht erbij te houden.
iv. Ja, zouden ze vaker moeten doen! De papieren versie gaf mij niet echt het gevoel dat ik iets nuttigs aan het doen was, gewoon simpelweg wat invullen. Het spel zelf was wel leuk om te spelen en het was wel leuk om te zien dat er aan het eind nog wat werd opgeleverd

ook.
v. Ja, ik wilde de highscore verbeteren, maar daarnaast maakte die todo-lijst het spel wel erg gemakkelijk om een juist antwoord in te vullen.

Resultaten

Na afloop van de tests kan geconcludeerd worden dat de Serious Game de gebruikers meer motiveerde dan de papieren versie. Ook gaven de gebruikers aan dat de Serious Game meer interactie met het proces bood. Door gebruik van verschillende hulpmiddelen, waaronder de to-do list die de gebruiker een duwtje in de juiste richting geeft, de directe feedback, konden de gebruikers zelfstandig en met minimale instructie een Use Case genereren.
De scoretelling hielp de gebruikers gemotiveerd te zijn en een goede prestatie neer te willen zetten.

Met een simpele applicatie is getoond, dat het proces van Requirements Modeling in een Serious Game kan worden gevat, en dat het een positievere ervaring levert dan het doorlopen van de vragenlijsten bij een reguliere papieren Use Case.

Hoofdstuk 6: Alternatieven

"A picture is worth a thousand words but it takes 3,000 times the disk space."
- Auteur Onbekend

Naast de in dit artikel gebruikte Serious Gaming methode zijn er natuurlijk ook alternatieven te bedenken voor de oplossing van dit probleem. Elk van die methoden heeft zijn eigenschappen, waarop ze zijn uitgekozen. Hieronder zijn de grootste tegenstellingen en typeringen opgesomd, elk met hun voor- en nadelen.

Statisch versus Dynamisch

Statisch
Een Serious Game is statisch in de zin dat de elementen van het spel statisch zijn, maar wel onderdeel uitmaken van een dynamisch spelproces, zoals Scrabble. Door het statische karakter is het relatief goedkoop en snel te maken, omdat de spelelementen maar eenmalig hoeven te worden geïmplementeerd, er hoeven geen mutaties of animaties te worden gemaakt. Een nadeel is, dat het minder betrekkend is dan een soortgelijke, dynamische applicatie.

Dynamisch
Een dynamische Serious Game spreekt gebruikers waarschijnlijk het meeste aan. Zoals een mooie film meer tot de verbeelding spreekt dan een enkel shot uit die film, heeft een goede Serious Game met animaties, kleur en dynamiek meer te bieden. Het nadeel ervan is, dat het allemaal geïmplementeerd moet worden, wat tijd en geld kost.

Tekstueel versus Grafisch

Tekstueel
Een Serious Game in tekstuele vorm kan veel uitleg geven over een proces. Tekst is een toegankelijke en veelgebruikte vorm van informatieoverdracht waar gebruikers mee bekend zijn. Verreweg de meeste artikelen, handleidingen, instructieboeken et cetera maken gebruik van tekstuele uitleg.

Grafisch
De uitdrukking 'een beeld zegt meer dan duizend woorden' kan hier van toepassing zijn. Het zicht is één van de zintuigen die gewend is om in korte tijd veel informatie op te nemen. Het overdragen van informatie en kennis is in grafische vorm dus erg geschikt. Ook kunnen beelden worden onthouden, naast tekstuele uitleg. Het nadeel van een grafische applicatie is, omdat het dynamisch is, van economische aard.

Virtueel versus Fysiek

Virtueel

Er zijn in dit artikel al veel voordelen van een virtuele simulatie- en leeromgeving voorbijgekomen, maar voor de compleetheid volgt hier een korte opsomming:
- Tijd; implementeren en aanpassen van software duurt minder lang dan bij fysieke varianten, simulatie kan op eigen geplande tijd worden uitgevoerd.
- Kosten; dure apparatuur kan worden gesimuleerd door middel van een digitale versie, processen kunnen worden versneld zodat tijdsgebaseerde simulaties veranderingen kunnen tonen op de lange termijn.
- Praktische zaken; software kan openlijk en wereldwijd toegankelijk worden gemaakt, bijvoorbeeld door middel van Internet, software kan relatief gemakkelijk worden gekopieerd en verspreid. Dit laatste kan ook in het nadeel werken.

Fysiek

Naast alle positieve punten van de virtuele versie van een Serious Game, zoals aanbevolen in dit artikel, heeft een fysieke variant ook zijn voordelen. Hoe gedetailleerd een simulatie van een bepaald proces ook is, het is en blijft een simulatie van een proces en daardoor een benadering van de werkelijke handelingen in dat proces. Een persoon die getraind is om een chirurgische ingreep puur en alleen virtueel te oefenen, zal bij het uitvoeren op een levend wezen toch een andere ervaring ondergaan. Daarnaast zijn er fysieke eigenschappen die virtueel niet of nog niet kunnen worden nagebootst, zoals het herkennen van geuren en zaken die het tastzintuig benodigen.

Hoofdstuk 7: Conclusie en discussie

> "Discovery consists of seeing what everybody has seen and thinking what nobody has thought."
> ***– Albert Szent-Gyorgyi***

Conclusie

In dit artikel werd gekeken naar de mogelijkheid om Serious Gaming als een hulpmiddel bij het proces van Requirements Engineering te gebruiken. De onderzoeksvraag, die gedefinieerd is als "*Hoe kan de essentie van requirements modeling worden ingekaderd in een spelachtige procedure?"* is beantwoord door middel van een theoretisch onderzoek naar de benodigdheden, mogelijkheden en functionaliteit van zowel Requirements Engineering als Serious Gaming. Hierna is een theoretisch kader gevormd voor een Serious Game met betrekking tot het RE-proces. Dit kader leidde tot de bouw van een testapplicatie, waarin enkele grote voordelen van de combinatie van Serious Gaming en Requirements Engineering werd getracht te tonen in een proof of concept-uitvoering. In deze testapplicatie, die vooral twee belangrijke functies van Serious Gaming demonstreerde, te weten *interactie* en *motivatie*, werd de gebruiker door een Serious Game geloodst, waarin de stappen werden beschreven van het proces 'een auto starten en wegrijden'.

Drie proefpersonen hebben daarna onder begeleiding een papieren Use Case ingevuld. Hierbij deden zij enige kennis op van de stappen uit het RE-proces tijdens de fasen van requirements modeling en requirements validation.
Daarna hebben ze zonder begeleiding een Use Case gegenereerd door de testapplicatie te doorlopen. Door alleen de instructies te volgen die ze werden gegeven door de testapplicatie, hadden de proefpersonen voldoende kennis en begeleiding om het RE-proces dat in de applicatie werd gesimuleerd, te voltooien. Dit is mede te danken aan de mechanieken die bij een Serious Game horen.
Bij de korte enquête die daarna werd gehouden, gaven alle proefpersonen positieve feedback op de Serious Game en werd de Serious Game Use Case methode geprefereerd boven de standaard, papieren invulmethode van Use Cases genereren.

De onderzoeksvraag is hiermee voldoende beantwoord; er is een theoretisch kader gevormd over hoe de essentie van *requirements modeling*, het beschrijven van de essentiële entiteiten in het gemodelleerde proces, hun eigenschappen en hun onderlinge relaties in tekstuele en grafische vorm, kan worden gerealiseerd in de vorm van een Serious Game. Ook de maatschappelijke veranderingen op gebied van acceptatie van computerapplicaties in het dagelijks leven en de veranderingen in denk- en leerprocessen door invloed van technologie die die acceptatie ondersteunen, zijn uitgelegd.

Serious Gaming speelt een steeds grotere rol binnen software engineering, zowel als methode om kennis over te brengen aan anderen en daarbij het leerproces te faciliteren, en als middel om software- of zakelijke processen te simuleren en te ondersteunen.

Discussie

In dit artikel is onderzoek gedaan naar de mogelijkheid om het proces van Requirements Engineering, en dan in het bijzonder Requirements Modeling, te simuleren in een Serious Game. De theoretische concepten die zijn aangedragen in verband met Serious Gaming zijn relatief nieuw in vergelijking met de fundamenten en grondslagen van andere onderzoeksgebieden binnen software engineering. De lijst met sterktes en zwaktes van deze nieuwe methode is dan ook niet uitputtend, en er is nog veel ruimte voor onderzoek omtrent dit onderwerp.

Het doel van dit onderzoek was het vaststellen van een theoretisch kader rondom de werking van Serious Gaming in Requirements Engineering. Doordat er steeds meer literatuur verschijnt over dit onderwerp en het vanuit steeds andere perspectieven wordt bekeken, staat de stof zoals beschreven is in dit artikel niet vast; ze is zoals ze op dit moment is. De experimenten die zijn uitgevoerd in dit onderzoek zijn slechts verkennend; het principe van een Serious Game die een Use Case kan parseren, wordt gedemonstreerd. Dit is gelukt, de beschreven principes zijn werkend en functioneel, maar er is nog niet sluitend bewezen dat een Serious Game als middel om het Requirements Engineeringproces simuleert of ondersteunt, in de praktijk werkt. Een mooi onderwerp voor toekomstig onderzoek.

Toekomstbeeld

De generatiekloof tussen oud en nieuw zal vergroot worden door de digitalisering; de jongeren zijn verzadigd door de moderne media, in het bijzonder het Internet. De oudere generaties blijven daar bij achter en dus ook in hun communicatie met de jongere generaties. Serious Gaming is een uitstekend middel om communicatie tussen generaties te bewerkstelligen. In de toekomst zullen spellen meer worden gebruikt voor communicatie en andere doeleinden dan alleen vermaak, dus zijn het Serious Games. Serious Gaming zal steeds meer buiten leeromgevingen worden toegepast zoals in de commerciële sector.

Literatuur

[Ambler] Ambler, S.W. *The Object Primer: Agile Model-Driven Development with UML 2.0*. Cambridge University Press, 2004.

[Christel] Christel, M. G., en Kang, K. C. *Issues in Requirements Elicitation*, Software Engineering Institute, CMU/SEI-92-TR-12 7, September 1992.

[Chen] Chen, M. P., en Wang, L. C. *The effects of type of interactivity in experiential Game-Based learning*, National Taiwan Normal University. In Learning by Playing: Game-based Education System Design and Development, 2009.

[Concept] ConceptDraw Pro. Afbeelding van Swim Lane Diagram. http://www.conceptdraw.com/products/img/cd5/article/part2/SwimLane_2.jpg

[Corti] Corti, K. *Games-based Learning; a serious business application*. PIXELearning Limited, 2006.

[Hood] Hood, C., Wiedemann, S., Fichtinger, S. en Pautz, U. *Requirements Management: The Interface Between Requirements Development and All Other Systems Engineering Processes*. Springer Berlin Heidelberg, 2008.

[Hoppenbrouwers] Hoppenbrouwers, S.J.B.A., Weigand, H. en Rouwette, E.A.J.A. *Setting Rules of Play for Collaborative Modeling*. In International Journal of e-Collaboration vol.5 (2009) nr.4 p.37-52, 2009.

[Intel] Intel IT Manager 3. http://itmanager3.intel.com/en-us/default.aspx

[Keegan] Keegan, V. *Games can have a serious role to play.* The Guardian, 11 December 2008.

[Kulak] Kulak, D., en Guiney, E. *Use Cases: Requirements in Context, Second Edition*. Addison-Wesley, 2004.

[Loucopoulos] Loucopoulos, P., en Karakostas, V. *Systems Requirements Engineering*. McGraw-Hill, 1995.

[McPhee] McPhee, C., en Eberlein, A. *Requirements Engineering for Time-to-Market Projects*. Ninth Annual IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS 2002), 2002.

[Nuseibeh] Nuseibeh, B., en Easterbrook, S. *Requirements Engineering: A Roadmap.* ICSE 2000 – Future of SE Track, 2000.

[Prensky] Prensky, M. *Digital Game-Based Learning*. McGraw-Hill, 2001.

[Pressman] Pressman, R. S. *Software Engineering: a Practitioner's approach*, 6th edition. McGraw-Hill Education, 2005.

[Sommerville] Sommerville, I. en Kotonya, G. *Requirements Engineering: Processes and Techniques.* John Wiley & Son Ltd., 1998.

[Stone] Stone, B. *Serious Gaming: Virtual Reality's Savior?*. Human Interface Technologies Team, School of Engineering, University of Birmingham.

[Susi] Susi, T., Johannesson, M., en Backlund, P. *Serious Games – An Overview.* School of Humanities and Informatics, University of Skövde, Sweden, 2007.

[Trusim] TruSim Interactive Trauma Trainer. http://www.trusim.com/?page=Demonstrations

[Verbraeck] Verbraeck, A. *Serious Gaming: Tomorrow's solution for today's problems*. Sun Microsystems: Noodpakket voor IT'ers Congres, 2009.

[Watergame] RO2 Watergame. http://www.watergame.nl

[WikiSG] Wikipedia artikel over Serious Gaming. Http://en.wikipedia.org/wiki/Serious_game

[Yusoff] Yusoff, A., Crowder, R., Gilbert, L., en Wills, G. *A conceptual framework for Serious Games*. 2009 Ninth IEEE International Conference on Advanced Learning Technologies, 2009.

[Zyda] Zyda, M. *From visual simulation to virtual reality to games*. IEEE, 2005.