

Personal information

Name: Eero Olavi Tuovinen
Date of birth: 20.5.1981
Nationality: Finnish
Contact info:
E-mail: eero.tuovinen@arkkikivi.net
Mobile: 0400-734794

Games breaking the mould

What follows are a number of concept sketches I wrote during a couple of days in December for a work application. The company in question wanted mobile games directed at new customer segments, which of course requires reinventing the form.

The initial requirement was for just one game, but I found the design parameters intriguing enough to do several, approaching the problem from different viewpoints. A part of the appeal is probably the fact that I've never before designed games for this particular platform: a mobile phone offers portability and networking opportunities, while demanding easy controls and play sessions of at most fifteen minutes. A quite different field, I'd say.

I found out after writing these that networking is still a no-no, though, so that blasted a couple of the best designs right then and there <sigh>. Well, their time will come later, and at least I have something ready for the next step in mobile technology. Be that as it may, the networking issue and some discussion about theme convinced me to send only one of these for final judgement.

While the games here are all designed as beginner-friendly, my goal is for each game to appeal with it's quality as well. A good game should entice gamers and humans equally. I couldn't put these into any kind of order (apart from picking the one for the judges), but there's certainly some more and less developed concepts in there. The game I finally picked for the application is the last one of the lot, so you can check yourself to see how I evaluated these babies.

Contents

PUZZLING INTIMACY	2
EXIT	5
BRIGIT'S TOWN	7
SMASH THE CAD!	10
THE MYSTERIES OF BRIGMAN BAY	13
CITY SPOOKS	15
ELEANOR'S DREAM	20
WIZARD JAMBOREE	23

Design parameters employed

- Mainstream themes familiar to general public.
- Entry-level gameplay.
- Technical platform: the average phone.
- Overall goal: a breakthrough product for new audience segments.

Language

I'll generally use 'she' for the player and 'he' for the character controlled by the player, to simplify the text. When the character has a gender, I'll use the other gender for the player. Nothing whatsoever is meant as far as gender politics are concerned; women can play games just as well as men as far as I'm concerned, and vice versa.

Puzzling Intimacy

This was a really strong contender for the first place, but I'd have felt queasy for submitting two games while leaving the rest behind. The choice was a hair's breadth affair, though, and was based mainly on believing the dictum of "Best innovation always wins." Although somewhat innovative, the Puzzling Intimacy is basically an adventure game at it's core.

Target audience: The same as television drama shows like *Sex and the City*, *Six Feet Under* or *Gillmore Girls*. That is, teenager to adult and male to female, without much difference. No idea how to get that group to try this, though.

General concept: A very freeform adventure game utilizing some heavy-duty roleplaying techniques. Skips much of traditional adventure game structure in favor of imitating the dramatic logic more closely.

Details

Technical framework: Mixed text and graphics with some graphs and menus, basic stuff. Some relatively complex graph theory, but nothing that can't be handled.

Theme: The player directs the life of a more or less ordinary person she creates in broad strokes. The game's about relationships, so the themes are ones of love, camaradie, passion and hate. The decisions of the player create a story that ends when the player's ready, more or less.

The general setup is similar to a drama show, with, say, an Indian family moving to Canada or something. Although the milieu is set in general terms, the player will craft the actual story through her decisions.

Character creation: The game starts with a series of questions, fixing the sex, age, and nature of the protagonist. The player gets to pick a picture for him as well, in the finest tradition of CRPGs.

The important step comes when the player creates a couple of secondary characters as well, and allocates a number of *relationship points* between them. For each such point the player picks from a list a definition for the kind of relationship the protagonist has to this secondary character: love, antipathy, kinship, work relationship or whatever. The player gets to pick the sex, age and other characteristics for these characters as well, to create the scene she wants.

After that, the game allocates some relationship points for each secondary character as well. Each character gets as many points as the player invested in him, minus one. The game might create tertiary characters (characters the protagonist has no relationship with initially) in this allocation, and if it allocates enough points, those tertiaries might get to allocate some relationships as well. Likely most characters will have relationships mirroring those the protagonist has towards them, but it's not a given that love or hate is mutual, for example.

When the relationship points are initially allocated, the result is a *relationship map* which the game creates and lets the player view. It shows a graph of the different characters and their relationships to one another, and hints of the events to come.

Game turn: The game is played in turns, each taking ten minutes at most. The player is limited to a number of turns per day, though, so the game will likely span weeks before getting finished. A day of real time might depict a week of in-game time, as far as story structure is concerned.

During a turn the player will react to *events* rising out of the relationship map. The game picks suitable events from a large list based on the statistics and relationships of the participating characters, as well as previous events. The event is a short description of the situation and a multiple-choice question about how the protagonist should react to it.

While the events might have immediate consequences (like, in an extreme case, the death of a character), their real import is in the changes that happen in the relationships of the characters. An event might be about a change that's already happened between non-protagonist characters, in which case the player can opt to react by changing some of the protagonist's relationships. Break a friendship with a guy who betrayed his father, say. Alternatively the event might be simply a plot turn not instigated by relationship changes, but even in that case it most likely will lead to some.

After the events the player has the option to actively affect the relationship map. She has two options in this regard, to pick the *manner* or the *relationship*: the former means that she guides the protagonist to do something proactive (like date another character, or look for work), while the latter means directly changing the protagonist's relationship values. The kicker is that while choosing the manner lets the player control what the character does, she cannot then control how the relationships are changed based on the action. Conversely, if she opts to influence the relationship map, that will trigger some manner of action from the protagonist resulting to that change, but the player has no control of how it comes about.

When the player has chosen the allotted actions (say, two or three) for the turn, the turn ends. The game comprises a short narrative of the turn's events for the player, based on the events and player actions.

The events are chosen pretty much based on the last turn's changes to the relationship map and the resulting relationship values. The result is that while the separate events are chosen in a very chaotic manner, a narrative of some kind emerges: if a love scene, for example, is only possible when both sides have three points of love invested in one another, then it has to be foreshadowed by events leading to that rise in love relationship levels.

The changes instigated by the events are of various types, giving the player a wide variety of choices. The common event might simply switch relationship values around, while a more serious situation could create or burn relationship points permanently. An event resulting in death of a character, for example, would destroy all relationship points invested in that character.

The same logic holds with player actions as well. The key to the crunch of play here is that the kinds of actions allowed are directly dependent on the relationship values in question. The character can only go on a date after gaining at least a point of affection, or can only plot murder with three points of hate. That kind of thing. To get the events and actions the player wishes, she has to manipulate character relationships to suitable values.

The game ends when a majority of relationship points of the protagonist character hinge on one event. Such an event is always extreme in some way, and the choice the player makes sets the tone for the ending.

Further elaboration

The above description is really a simple framework for a relationship-based adventure game, and can thus be twisted this way and that depending on interest. here's some elaborations:

- To make a game more alike to classic adventure games, set definite win-loss conditions instead of player-supplied ones. If the protagonist goal is to find true love before his father succumbs to a fatal illness, for example, it's a whole different game.
- Although the above description is not too interested in the milieu of the game, it certainly has much to do with the kinds of events that come up. One option would be to base this kind of game on a licenced property, letting the player try her hand at directing the familiar characters to a happier ending. A better option than making another platform action game, I feel.
- To get more definite crunch, it's a simple matter to include subgames of different kinds, triggered by the events. The whole relationship thing could be subsumed into a top-down adventure game, even, although that kinda loses the point.

Designing for the target audience: The idea here is to lift the adventure game out of a couple of binds that make the genre pretty unplayable. These are linearity, procedural game play and nailing play onto a space-time grid. All are pretty serious fallacies, and not really the point even in the masculine puzzle solving games.

Exit

This is a contemplative action game. I wasn't sure about working this idea at all for this collection, but decided to go with it because of pure appeal and simplicity. The other ideas are really flashy and exciting, but this is immensely doable.

Target audience: Everybody, more or less. The symbology used by the game is so universal, and the gameplay so simple, that it shouldn't set any barriers for play. The goal is to include people who like Solitaire or Minesweeper, even if other games pass them by.

High concept: The concept is simply stated: an action game with the guy to the right as the protagonist. Yeah, he's the guy from the traffic sign, and the boulders from another sign are about to squash him.

Details

Technical framework: Extremely simple two-color graphics, but also quick, fluid gameplay. Quite a number of game modes, though, so the engine has to be generic enough for the possible sprite interactions.

Theme: He's the Exit guy, the one in the green exit signs, traffic lights and traffic signs. He's famous! He's featured in commodity products, warning symbols, anywhere and everywhere you need a simple and unambiguous character. The real everyman.

However, now his family is threatened; remember the "Children on the road" sign? That's his family in the picture, and now they're gone! As this is an action game, I won't bore anybody with rationalization. The Exit guy is going to save his family, and to do that, he has to jump through our hoops.

What play looks like: Bright colors. No statistics of any kind to be seen, changes in play state are shown by color changes. The Exit guy and everything else is always in one color, and the background is in a highly contrasting color. Play speeds up slowly and slows down suddenly, making for a dynamic impression.

When the Exit guy is hit, or the game speeds up, the opening colors of white-on-blue are replaced by black-on-yellow, which are later replaced by red-on-yellow. Very primal, and following the symbolic logic of traffic signs and such.

Play experience: Gameplay is more or less the normal action fare: the Exit guy runs, jumps and otherwise avoids hazards, races through traffic sign land, even hits and punches from time to time. The game consists of multiple different simple scenarios like those on the right. The game is instantly understandable on the instinctual level, thanks to the symbology used. Controls are the direction buttons, nothing more.

The play of a given scenario starts slow, even glacial. The Exit guy is very robust, and hitting things or otherwise failing in the action will just squash or bounce him a bit. It takes real effort to really hurt him. However, the game speeds up slowly, and it becomes easier and easier to take a total hit.



There are plenty of scenarios, and the player has no real control over what's played next. No explanations are ever given about what's going on, but mostly it's pretty self evident. The goal is always the same, finding the exit. The thing is, exit is not found through continuously better play, like in other games. No, just playing will sooner or later result in the game getting so quick that the player cannot, however skillful she is, avoid the disaster.

Instead of playing better and better the player will have to play *differently* to find the exit and escape the scenario. For example, if the scenario is about dodging boulders, perhaps the Exit guy should choose his moment and climb over to see where the boulders come from. Or maybe he has to jump on a boulder and balance on it while it bounces to the exit. It's that kind of lateral thought that will find the exit, not just continuing to play the scenario.

However, when the exit is found, the Exit guy gains points for the time the scenario lasted, so play can be about choosing the moment of exit. The game is saved between scenarios and the points accumulate. Gaining enough points will open new scenarios, until finally the last one is played and the Exit guy saves his family. The thing is, although the points are shown and even a high score list is kept, the opening of new scenarios for play is not displayed anywhere. When a session is started the next scenario to play is chosen semi-randomly from among those opened, so the player gets to replay the situations from time to time.

Further elaboration

There should be perhaps a dozen different scenarios at least, although more wouldn't hurt. When the game is started one of these scenarios is chosen for play. The player will have to figure out how to find the exit, meanwhile gaining enough points. After the first exit, if the player gained enough points, another scenario is opened in the list of potential scenarios. The more scenarios opened, the more points the rest will require. Also, the player will have to replay the old scenarios several times to gain the points to open the final scenario. The game places no emphasis at all on play-through, but it's possible.

The scenarios themselves range all around and through the world of symbols, being by no means repetitive. Exit guy will have to navigate mazes made of highway connection signs, race through hills from "humps in the road" signs, choose quickly between different hazard symbols (fire, poison, corrosive... you know), dodge washing instruction symbols in a spin-dryer, duel with Katakana Samurais, carry a cross around, lead a war between flags, put a coat of arms on the throne while arranging heraldic symbols, disassemble a sickle and hammer and translate hieroglyphs, to mention a few possibilities.

And through it all, there is not one word in the game. Everything is expressed in symbols.

Loss condition: An important part of the game's nature is how it handles player failure: when the Exit guy hits something he shouldn't or the player fails in some other way, it won't for the most part mean game ending. Rather, the Exit guy will just bounce around somewhat and continue play. Only a major mistake (defined for each scenario separately) will end the game.

Control scheme: Through all the scenarios the controls are handled with directional buttons, the simplest possible control scheme. The game can be played with only one hand. The controls are never explained, but this simplicity allows the player to find out herself what's what.

Designing for the target audience: Well, the game is easy (all scenarios start very slow), light (simple scenarios) and appealing (tell me the Exit guy isn't a great protagonist). Should be quite the hit.

Moreover, this is a pure minigame in the sense of being fast to play. Like the classic mobile game *Worm*, Exit is sufficiently streamlined to become something an ordinary person might play while waiting the buss. However, there's the idea of a continuum of depth here, starting with a constantly speeding reflex action game and progressing to opening up new scenarios for play. A very versatile design, intended to build a personal relationship towards the player.

Brigit's Town

This one is basically Nintendo's Harvest Moon made multiplayer and improved somewhat. I feel that Harvest Moon is the quintessential example of blending adventure and building games, and as the theme is favorable, there's no reason to not learn from there.

Target audience: Young adult non-gamers who could play if there were the right kind of games, like *Sims*. The game is simple to understand and play, but also includes enough substance to entice actual gamers. The perfect gateway drug, in other words.

General concept: The player controls a guy (or a gal) who arrives to an agricultural town and starts a farm. He buys seeds, plants, hoes and finally harvests the produce, which he sells on the market. A little resource management game, in a way. There's also some interactive stuff towards other players, and the preferred amount of adventure gaming.

Details

Technical framework: The game is server based, with a client program that downloads the game state and uploads player commands at the end of play session. One day of game time is roughly fifteen minutes, which is the sweet spot of daily play for players. One server should hold from five to five hundred active players, the game is flexible in that regard.

A simple 2D perspective from above the character is quite sufficient for the game, with preferred amount of graphical elaboration. The individual farms have to be graphically presented, but other parts of the town can be dealt with textually.

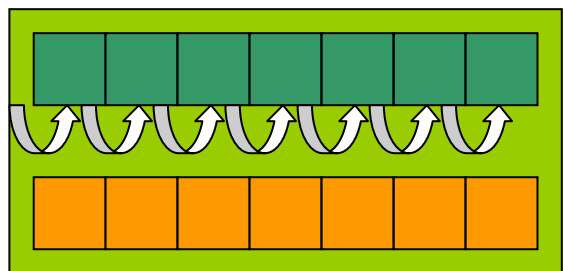
Theme: The graphics and dialogue are warm and cartoony. Success in the game is based on cooperation and honesty, and these themes are reflected in the representation. No realism about farmwork is sought; rather, it's depicted as nostalgic, virtuous and clean.

What play looks like: The character stands on grid-based terrain, representing his farm. The player sees a couple of squares to each direction, like the typical handheld adventure game. The player moves the character around the farm, doing all kinds of chores, mostly with a simple one-button control system. Press the button to pick up the bale of hay, press it a second time to give it to the horse, that kind of thing.

When the character moves out of the farm, play becomes menu-based; the player chooses where the character is going from a menu of known locations, and there the character is. Most locations, like the market, are themselves menu-based, while some are like the farm represented as terrain where the character moves.

Play experience: The player starts the game day when her character wakes up. He does some farm work, which repeats from day to day with slight variations. After the farm upkeep (about five to ten minutes), the character has some free time to spend in forest or town, in different subgames. After nightfall the character can only check statistics, leave messages for other players and do such minor things, so he might as well go sleep. The player can pause the game freely, as it's not connected to the server until the turn ends.

The central issue for the player is to manage the time the character uses in daily chores like taking care of the animals and crops. The chores change with the seasons and the player becomes more proficient in doing them, but in the main they are repetitive. The idea is a calm, unhurried experience of methodical fingerwork, relaxing for the player instead of stimulating. The real play decisions are made in choosing the investments for the farm.



The defining feature of Harvest Moon are the crop patches, which are methodically watered every day, one by one. The meditative repetition forms the backbone of the game's rhythm.

The resource management part of the game is at first somewhat simpler than in Harvest Moon, which is reasonable considering the game's goals. It's however again complexified by player interaction, about which more details below.

Farm management: When play first starts, the character is nearly destitute. By taking a loan and starting to work he builds a farm. The player gets to place the farm buildings on empty, grid-based land after paying for them. The buildings can either be bought or the character can build them himself, but that takes several days of game time.

After building a home for the character the player can invest in seeds (a couple of different types) or animals (likewise). The rest of the land of the farm is then prepared for grazing or farming by the character. In practice the player spends most of multiple game days laying out furrows and ploughing them, and with patience the fields come ready.

When the crops or animal products are grown, the character can take them to the market to be sold. With the money he can then pay back the loan and continue investment, perhaps getting more efficient tools to work larger fields. Or new clothes, as the case may be; a little luxury never hurt anybody.

All this is achieved through a single-button control scheme that's easily learned. The point is in establishing a rhythm in doing a chore; a typical chore might be moving grain to a silo, a question of pressing the button to pick up grain, moving to the silo and pressing again to leave it there. As these chores are repeated every day, a hypnotic peace overcomes the player.

Adventure game: This is basically sprinkled on to taste, depending on player commitment. The idea is that if he does the farm chores quickly, the character has time to go out and about somewhat. The most common target is the market, where he goes to buy seeds and sell produce. Another place could be the forest, where he can saw some timber or collect berries. Or the coast, for fishing. There could be NPCs that offer different perks and sidequests as well, depending on the extent of the game.

Multiplayer aspects: All the player turns are played independently, and there's little player communication. Interaction, on the other hand, happens in multiple forms. The most important is the free market: all the market interactions in the game are calculated by the server based on player action! Here are some examples:

- When the server is first started, the town is empty of people but full of free land. The first player to start play becomes the mayor of the town until a new one is elected. Land is free to claim, but the mayor can redistribute unworked land when more players come up. Thus land is free at first, but might raise in value later. A natural limit for the number of players on the server is the amount of land in the game.
- Every day the market buys an amount of produce dependent on the number of players. However, the players get to set their own prices, and the cheapest are always bought first. Thus a player might not be able to sell everything he produces, and has to discard spoiled goods in the worst case.
- Players can team up and form a family: the farms are joined up, and both players work the same farm. Players have to play their turns at different times of the day, though, and coordinate their work somewhat otherwise as well. Thus trust is required for the combine to work.
- When there's only a little free land left, there's the option for a new player to play a child of a family instead of a stranger. She will have the choice of staying on the farm or starting her own.
- Players can also buy and sell stuff with each other on the market. It's a perfect exchange model, so if one wants to sell some seeds for a low enough price and another wants to buy, a sell is achieved. Other stuff to buy and sell could include animals and lumber (for building), for example.

- Players can visit each other's farms. Although there is no realtime communication, they can leave notes and observe how the other is taking care of his place.

All in all, there's enough substance in the game to warrant a simple web forum for players to interact in more comfortably. There the more committed players can negotiate on prices and search for farming partners.

Meanwhile the occasional player can perfectly well play the game by taking the fifteen minutes to run a day now and then. It's no big deal to miss a day of work: the computer plays those turns for the player, saving the money until the player comes back to invest it in some manner.

Further elaboration

The game is a personalized multiplayer resource management game, and as such easily expandable. The two key ideas are the resource management and multiplayer, which both can be emphasized through different ways:

Resource management: The amount of investment options controls the complexity of the game to a great degree. Will the player have a choice between slow but expensive and fast but cheap crops? Can he buy a milking machine to skip a chore? Or fertilizer to speed up production?

The choice balance is a situation where the resource management starts simple and stays that way for first-time players, and only becomes more complex if the player searches for that. So the extreme solution is that at first the player has almost no management choices at all, and the options are only opened through the adventure game: talk with the guy at the store, and he lets you buy different crops, that kind of thing.

Multiplayer options: There are a great many, but the most interesting one is the idea of a living economy: what if a character could get an oven and start baking bread? He would need flour for that, which is pretty expensive. But if another character got a mill... then the latter could grind some crops into flour, which he could sell to the breadmaker. The bread could perhaps give strength to the farmers to work harder, and in any case it would be valuable to sell out of town.

What's more, the multiplayer economy can control the options the players have: at server start the play is simple and options few, as the players have to carve a town out of the wilderness. But when they have satisfied the outgoing market for enough days (a simple level system), perhaps new options open up. Get the entire town to level three and suddenly you can buy the oven and the mill and be baker and miller. At that stage some players could perhaps downsize their farms to make more time for these new jobs. This way specialization emerges, and the whole town gains prosperity from player cooperation.

Considering this angle we see that the game has potential to be much more than Harvest Moon; the basic idea of relaxed farm simulation can be the basic form of play, but a player can move beyond that to an accessible and light multiplayer economic game. The enforced day-cycle and automated market make sure that the game doesn't become laborious for the players, but the development options and interaction with the other players keep the game fresh. Ambitious players could even take it as a goal to build as prosperous a town as possible!

Designing for the target audience: The key question in the final calculation is the appeal of the game for a non-gamer. Although the game represents a traditionally hardcore genre (resource management), recent developments have proved that it can be made approachable for non-gamers as well. I cite *Sims*, which belongs to exactly the same genre. The strengths of *Brigid's Town* in this regard are it's warm and fun theme, simple and repetitive basic gameplay and finally, the interesting interactions with the other players, among which the player can seek friendships and alliances. The intention is for the game to be like the recent multiplayer *Sims*, but with a more constructive gameplay, reformatted to a mobile platform.

Smash the Cad!

This game's a competitive puzzle game, most resembling Worms. What sets it apart is the method of fighting, which is kind of similar to the Incredible Machine. In any case, Worms is an obvious choice for a mobile game, networked or hotseated. Smash the Cad! is perhaps the weakest design here as far as the design parameters are concerned, but it does have potential if I'll ever come back to it.

Target audience: Young adults looking for a quick, fun and simple game. *Smash the Cad!* is very humorous, simple and fast to play, so it has potential to break through as a social game in the Japanese model.

General concept: Players from one to N (whatever the programmer is comfortable with) take turns in building machines to squash the other players' cads, which are the characters controlled by the players. There's a time limit to the turns, and the machines others build only start moving when a player's turn starts, so the player has to both dodge doomsday machines by the other players, as well as build some of her own. There's a high intrinsic element of luck, and the implementation of machines themselves is highly humorous. Hilarity ensues.

Details

Technical framework: The game is hotseat multiplayer, so the players need one mobile phone to play. The engine is realtime 2D sidescroller, like a platformer. A zoom option on the playing field might be helpful, but can be designed around.

The actual engine is predicated on logical combinations and a very rough positioning system instead of any kind of machine physics, so the memory demands should not prove excessive.

Theme: Zany humour without a whit of sense. No background plot really – the cads just are convinced that there can be only one. A suitable graphic might be either some very distinct characters in *Micro Machines* style, or identical amorphous blobs that can be different colours.

The cads themselves spew inappropriate commentary all through player turns, giving them some characterization. The commentary comes to the screen as text, at times obscuring totally whatever the player is actually doing. This is not a game to be taken seriously.

What play looks like: A player's turn starts with a black waiting screen, but immediately when he activates the game by pressing a button everything starts to move. The playing character is in the middle of the screen in the usual platform game manner, usually edged in by some doomsday machine of another player's. After dodging (jumping, running) the trap the player starts to build her own machine: she presses a number of buttons to place objects on the screen, hoping that they result in something lethal to the other player's cad. There is, however, a time limit to the turn, and when the time ends the player's cad is left wherever it is and another player gets the opportunity to... Smash the Cad!

Play experience: Fast and furious. The player is primarily concerned with dodging whatever the other players have concocted and moving to a suitably safe place in the terrain. When enough players have spent turns dodging machines, the ones still in the game start to build more machines. The player tries to put together something fast enough so the other players don't get their own doomsday cadgets ready before she does. Preferably something that squashes the cads of other players and doesn't blow up in the face of her own.

The action is situated on a level with different platforms and other typical platformer implements. The player generally has only a vague understanding of the level being played in, but it's tight, and usually the cad has to run blind to escape disasters.

Machines: The platform action is pretty simple; the real meat is in building the machines. During her turn the player can at any point leave her cad where it is and start planning a machine, or perhaps add to one already operating.

The machines are built out of parts picked from different lists, like in *Incredible Machine*. The parts are stuff like cogs, candles, cats, fans, fertilizer sacks or dynamite. They all interact with each other in different ways, so there's quite many effects that can be achieved with the machines. The player picks the parts she wants and just drops them down onto the playing field wherever she wants to. For example, the player might try to get the correct parts for a conveyor belt that drops a safe on an unsuspecting cad.

The machine interactions are completely prescribed and have very little to do with physical reality. Most are horribly lethal to the cads, just like in an old Warner Brothers cartoon. Give the cad a giant mallet, and he'll try to squash the others if they come close. Drop a beehive on a cad, and he'll have to run for his life. Drop a cad into a pool of water and he'll sure drown, unless he's on fire, in which case the water vaporises. That kind of zany humour.

The single part machines are the simplest, but the better effects are all based on combining different parts. Almost any parts can be combined to get *some* effect, but for the most part there's no guarantee that it benefits the player who did it. The idea is that the game is too chaotic for anybody to control, so skill has a very limited part in the proceedings.

After dropping the machines on the field the player can correct their positioning and combine them further with his cad. Move the cad next to a candle, press the button, move the cad and the candle next to the dynamite, press the button, run. To further this principle, the playing field is of course full of stuff to begin with, so the players can just use that instead of delving into the menus.

The platform game really only uses two buttons: one to jump and another to manipulate stuff on the playing field. All the other buttons, a ludicrous amount of them, are reserved for controlling the menus. One menu button is the main button, opening a menu of subcategories (animals, chemicals, mechanical implements, so on), while another press chooses a subcategory, in which there's all kinds of implements. "That's too slow, I only have a minute!" Right, and nobody knows what's in the categories anyway, because they are randomized at the start of the match. You can learn them, but that takes time...

The alternative solution to the main menu are all the other buttons (all, to make mispushes more likely), which each have a completely logical, totally lethal and all too zany function. The button just next to the menu button picks whatever item the last player picked last. The two buttons over there, one of them picks the item you picked last, while the other picks dynamite about to blow. They're randomized each turn. One button just picks a random item, and so on. The player can through experimentation and blind luck stumble on a faster way to do things.

Turn structure: Each turn is one minute long, real time. When a turn starts the player has a second or two to understand the situation her cad is in before the safe drops. After that it's all running. Any machines the player activates only start moving on the next player's turn, but any machines moving continue until stopping for some reason. Cads can perfectly well perish on another player's turn, so better get somewhere safe before your turn ends.

The next turn always goes to the player whose cad is in imminent danger when the last turn ends. Players might lose cads without ever getting the chance to dodge the danger, but they'll be soon back in the game: the game counts death points, and a cad squashed on another player's turn is only worth one point, while one lost on your own is worth ten. On your next turn your cad drops out of the ceiling of the level, most likely straight into a trap set by another player just for the occasion. The last player to hit a predetermined amount of death points wins the game.

Designing for the target audience: The goal of the game as a product is to support the grapevine effect in hooking new players, and to keep them by supporting social gaming culture. The eastern Asian countries are the model here: Japanese gaming habits are primarily social, with arcades and console gaming nights between friends common. As a straight consequence of that

gaming is a socially functional hobby, making getting into it much easier. The hardcore gamer population is supplemented in Japan by an enormous (comparatively speaking) amount of occasional gamers in a much healthier model than in the west.

Smash the Cad! strives to hook non-gamers by being a game easily introduced and meant to be played with a group. It's not perhaps a game a non-hobbyist would get himself (then again, there is none such by definition), but it's definitely a game that a player will want to introduce to his friends.

The turn-ordered multiplayer nature of the game supports play style that's social rather than immersive: the game's played as a complimentary social activity, with the phone changing hands and the majority of players interacting in other ways while one plays her turn.

The Mysteries of Brigman Bay

This is a multiplayer detective game, like the boardgames Clue and Mystery of Peking, only more complex. I figure that the boardgame connection as well as the literary tradition behind it make it easy to explain and attractive to newcomers.

Target audience: Gamers and newcomers, really. The game should be very attractive to women as well, being that it features intuitive decisions, clear logic and no numeric analysis.

General concept: The player is one of the many detectives in Brigman Bay, a town riddled with crime. The detective rises with dawn and hears about a committed crime: robbery, defacement, even murder! The police is powerless! Through clear perception, good manners and astounding logic the detective figures out the crime. The quickest player wins the day.

Details

Technical framework: A top-down adventure engine. Server-based play with the clients downloading the game-state at the start of a session and uploading results at the end. Preferable amount of players per server from 5 up.

Theme: Constant ribbing on detective stories. The player chooses his character archetype from the English (looks like Sherlock Holmes), American (Humbrey Bogart), boy (Hardy Boys) or girl (Nancy Drew) detectives. The rest of the game is filled with that kind of humour and references to literary predecessors. The town of Brigman Bay is itself a general small town, with parts modern and parts victorian.

What play looks like: The player moves around with her character and collects information on the crime of the day. After she has enough to pinpoint the guilty party she either chases the criminal, holds an expose event or tells the police, depending on the crime. Each day of play time is fifteen minutes long, and the player fails, if she cannot find the criminal in that time. After the sun has set the detective can still wander the streets looking for hints on the crime of the next day.

Play experience: The play experience is structured by the mystery of the day. Each crime has clues left at the scene and witnesses, and the detective can research both. From these he gains facts like “Next the criminal went north, to the marketplace” or “He clearly had a mustache”. The game keeps a list of these for the player, but she will have to decide herself when she knows enough. The pool of suspects depends on the mystery, too, but is usually quite small.

The town of Brigman Bay is much larger than the actual area where the usual crime can be solved (they have to be solvable in under 15 minutes after all). This is so that the crimes can happen in different places and to different people; the town is full of NPCs (non-player characters) with their own habits, so change is good.

Additionally, there are several facilities around Brigman Bay the detective may feel like visiting. Hailing a cab will save some time getting from place to place; going to the police station might be necessary, if the criminal is already in their books; the pastor of the church can reveal if any of the witnesses are lying, and so forth.

Advancement: At the end of the day, the player who solved the crime fastest gets a special prize. Also, all players who managed to solve the crime get a level as detectives. These detective levels are mostly a badge of honor, gaining the players titles like “The Master Detective” and the like. However, the level also limits participation in some mysteries: some days there are two mysteries in town, and only detectives above certain level can try to solve the second one.

Mystery Structures: The basic structure of a mystery is quite simple: there is a pool of suspects (“Only those three were in the car!”), each with somehow differing traits. These can be in appearance (“The one who did it had a scar in his cheek!”) or motivation (“The only one to benefit

from the murder was the nephew!”), but the job of the detective is to just find enough clues to close off all but one of the suspects. If a harder mystery is wanted, some of the clues can be false.

The creation of a mystery is simple enough that a limited kind of mystery can be created by an algorithm. A simpler solution is to build easy-to-use tools for mystery creation and let a person churn out some.

Further elaboration

There are all kinds of elements that can be added to the basic concept above. I'll give some examples:

- **Detective statistics:** Each detective has three abilities; Perception, Charm and Knowledge. Each potential clue is rated in one of these from zero to three, and only the detective with a high enough ability finds it at all. Crime site clues are under Perception, witnesses are under Charm, and all other clues are under Knowledge. Each character is created by dividing four points between the abilities. Each mystery is barely solvable with only one point clues in all three areas, two point clues in two or three point clues in one area. Thus different players will have a radically different time with the mystery, increasing playability.
- **Perks:** Each detective can carry one perk and store more. These are stuff like rollerblades for faster movement or a sidekick to send do routine investigation. They would be gained mostly by being the fastest to solve the crime of the day.
- **Crimes:** Let the players commit crimes for different perks. After the sun has set the player can choose between a variety of crimes for different perks, and has to play a subgame to determine how well he pulls it off. The next day, his crime will be one of those being solved. The detective goes to jail if somebody solves the mystery, though. Characters gain criminal levels as well as detective levels, and titles like “Master Criminal”.

Designing for the target audience: Detective games are a recognized part of the family boardgame scene, so the idea behind the game is simple to grasp. Because there is no abstractions and resource management usually seen in the adventure genre, it's easy to play the game from the start. The game is learning-friendly, because the player can continue figuring out the puzzle even after the time runs out, to learn to do better the next day.

City Spooks

This is, in essence, a phone-assisted live action roleplaying game similar to the old game of Assassin. What makes it suitable for general consumption is the low level of commitment and performance required. Of the games here I feel this has perhaps the greatest potential, as it rides the same social wave as parkour and geocaching, for example; city is becoming a space of self-expression for the urban youth, and there's room for a game in that.

Target audience: The game is targeted at the trendy young adults. City Spooks doesn't look like a computer game, but rather resembles certain sports and extreme hobbies, making it an obvious match.

General concept: Players are divided into loose, shifting alliances, and try to "conquer" blocks of the city for their alliance. To do this, they try to meet with their allies, while simultaneously avoiding the opposing team members. Players recognize each other based on identifying marks they input into a server.

Details

Technical framework: A server-based, text-output game. Server has to take a couple of hundred players, at least.

Theme: No particular theme. Player alliances are defined on the basis of popculture references, so the whole affair has a rather metagamey feel to it. The game gains meaning from competition and real life social activity, like an actual sport, instead of imaginary content.

What play looks like: Players, all living in the same city, log on to the server and start revealing identification details to other players in their alliance: where the player is, age, sex, color of hair, clothes, whatever the player wishes. At the same time the player scans the database to find another member of the same alliance conveniently near. The goal is for the two to meet and trade passwords.

Meanwhile, members of the other alliances gain some of the same information the player has put out to arrange the meet. A player can gain points in the game by finding a member of the other alliance as well as one of her own, so the player who arranges the meeting has to be careful in what she's revealing and where she arranges the meet.

The game is inconspicuous to non-players, although it's played on the streets and in the buildings of the city. At worst, a player might approach an outsider by mistake.

Play experience: When the player first joins the game she's asked some basic questions to determine her starting allegiance. The questions are of social nature, to give the teams some character. "What's your favourite music style?" or "What hobbies do you have?" would be typical. Based on these, the player is assigned to an alliance by the server.

The player can choose herself when to play and how long. When she logs on the server, she can scan the activity of other players or start some action herself by starting to arrange a meet. Players have the option of getting a warning in their phone when something big happens, to react quickly.

The central experience in the game is the actual contact between the players. For the most part players don't know each other, so the game is about spotting a player of the game from the crowds of the city. What's more, the player doesn't know who might belong to an opposing alliance, so she has to be careful in revealing herself. It'd be easy to carry some unmistakable sign and reveal it on the server, but then she'd be easy pickings for anyone who arrived first. The main strategy in the game is in imparting contact information to your allies without revealing yourself to the opponents: there could be prearranged code words, or you could simply know each other by sight, or obscure references could be made. (In the team of *MTV Generation* you might try for contact with "a guy

dressed like **Justin** in *Cry Me a River*”, in the hopes that more members of your team know the reference than in the opposing teams.)

Finally, when the contact is made, the players exchange codes supplied by the program and find out who caught and whom (the game articulates itself in terms of a spy-game: the players are spies for their alliance, trying to catch spies belonging in the other alliances and meet with their own); the players can't know for sure who is on whose side without inputting the code of the other.

The suspense of arranging the meeting is boosted by the simple human contact between players. They'll know something of each other from the server (“Hey, we're both in the *Soccer maniacs* alliance!”) and from the fact that both play the game, so there's already some basis for acquaintance. Thus the game has a strong social aspect as well.

When the player has played the game for some time, she'll get to know other players. That only makes the game more complicated: players might have to start using code names to hide themselves, and members of the same alliance can develop mutual codes of communication. But the alliances shift constantly, so the players cannot know for certain that a given player is still their ally!

In a particular game session the best results of play are achieved by cooperation within the team: after contacting a player of your own team, you'll know for sure one player who's on your side this week (or however often the alliances are scrambled). Two players are much more efficient in hunting opposing team members than one, so you might arrange some cooperation. Likewise, if you're caught by another team, you can then reveal the information about that particular player's current allegiance on the server, perhaps helping your own allies to catch her.

The structure of the game is such that players can choose very freely how much they play. The player is “safe” and out of the game when she's not logged onto the server, so there's no pressure to continue playing to stay ahead. Two particular players are rendered “inert” towards each other as far as the game is concerned for 24 hours when they meet, so there's not much to be gained by excess play either: after a couple of hours of play you'll probably have met everybody who's going to play tonight.

Rules: The rules of the game are very simple; most of the action is *emergent* in the sense of game theory: human social structures supply most of the game experience.

The simplest kernel of the rules go like this: the alliances are randomized at the start of the week, effectively resetting much of the game state. When players input data about their current whereabouts (or whatever they feel useful) to the server, all players of all alliances can see everything right away. Players can *mark* other players on the server as their *target*, but can only have one such at a time. When two players meet, they can exchange codes (they don't have to), revealing team allegiances to each other. Such an exchange scores points as follows:

- If both are in the same alliance, one point to each and one to their alliance.
- If the players are from different alliances and neither has targeted the other, neither gets points, but their alliances gain one point.
- If the players are from different alliances and one has targeted the other, the targeting player gains one point, her alliance gains one point, and the target and her alliance lose one point each.
- If both have targeted the other, the marks counter each other and the players don't get points. Their alliances still do, one each.

In any case, the contacting players are rendered inert to each other, and cannot exchange codes for 24 hours. At the end of the week the points of the alliances and individual players are calculated and winning alliances and best players are chosen. The final player scores are calculated by adding the player's personal score to his alliance's overall score, so there's an incentive to work for the alliance.

The above is the simplest form of the rules. All that is required is a structured database for revealing information, a code generation routine to recognize players, a point record, and a player

database. This is enough to produce a complex emergent game of alliances, betrayal and doublecrossing.

Further elaboration

However, the game experience can be deepened by adding details. These need not complicate game play, but only serve to add player options. The base rules are really just a simple spy simulation, so anything that makes sense in that context can be implemented.

Controlled alliances: As outlined above, the player alliances could be partially based on metagame consideration of player hobbies and interests. This way members of the same alliance will always have some feeling of solidarity and affinity.

The alliances have to change from time to time to keep the game fresh, but the change could be gradual and partially player-controlled: each player is asked a new multiple choice question each day, and the answers are saved in a matrix that's used to calculate alliances. Players with many same answers are more likely in the same alliance, while those with many differences would be in different alliances. New questions (and dropping of old from the calculation) will slowly change the alliances.

Similarly, a player could be given the change to *prefer* some alliances to others. Perhaps the player's point record will give her influence in this regard, so that the better players can choose their alliances more easily.

Finally, there's no reason for a player to always be only in one alliance. It could be possible for players to be part of several, to make the strategy of the game more complex. In this case the alliances *and* players would all score points once for *each* alliance relationship of the players.

Allowing this kind of semirandom alliance structures lets players build alliance identity for their "character"; one player could be the core of a given alliance, who everybody knows and would never think could be in any other alliance. Another could be jumping from alliance to alliance constantly, so that nobody knows if they can or cannot trust him.

The only thing that cannot be allowed is completely player-controlled allying, because the obvious winning strategy in the game is to know all your allies, in which case you never need to make contact with possible unknowns.

Limited database: In the above rules kernel everyone has access to all information instantly, which means that friends are just as likely as foes to make contact when a player moves to reveal herself. This can be easily changed:

There could be a time delay in how fast the data becomes accessible outside the alliance, to give the allies some lead time. Or the system could keep a record about ally contacts players have made, and the data would first go to those the player has trusted in the past, regardless of current alliances.

More ambitiously, part of the process of data insemination could be random, so the player doesn't know what the opponents know. The player could even give priorities to "data security" of each snippet, to get some control over who knows and what.

The data control aspect is a whole another subgame. Like spy fiction, success and defeat in this game depend on who has information and how he uses it.

City geography: As an added wrinkle in the game, the alliances could be concerned with "control" of the city instead of (or in addition to) abstract points. Say that the city the game's played in is divided into a number of "spook zones", which are where the contest is played in. In case of Helsinki, for example, these could simply be downtown city blocks and squares, like Stockmann, Kaivopiha, Metrotunneli, Forum, and so on. In any case the zones would be just large enough to allow some searching of other players, but small enough that there is a good change of finding them.

When players are setting up meetings, they will have to specify which spook zone it's going to happen in. Successful meetings and catches will score normally, but the alliance points are calculated for each spook zone separately. The overall scores of the alliances would depend on majority control of the spook zones, so they'd have to try to win each zone separately.

Also, the inertia of players towards each other would be changed: the players would still be inert after contact with each other for 24 hours, but only *for the spook zone the contact happened in*. Thus the same allies could travel through the zones and score them all within the day, but be very conspicuous to the opposing forces at the same time.

This structure would focus the play considerably. That would happen anyway when the game becomes established, as players would learn to congregate in places where others hang out as well. But this way the game designer could pick the places of congregation, perhaps giving players a more interesting and varied experience. "Today we'll take the northern blocks, let the Black alliance have the rest!" and that kind of thing.

Player hierarchy: There are various perks the players could earn by good play. Specifically, player scores within alliances and between all players in the city could be used as a resource the player uses to buy perks, or some perks could only be gained by being better than others. Still others would depend on player votes. Some examples:

- **General titles:** Good players could get spook titles ("Agent 007", "The Ghost", et cetera), cool avatars in the virtual system and such.
- **Alliance leader:** The player with the best performance weighted with votes of alliance members would be the leader of the alliance. The benefit would be getting to name the alliance, and that's a cool title itself.
- **Efficiency boosters:** There are a number of different ways to give players power in the game: a player could be allowed to mark more than one target at once, or could score more for marks of a certain alliance, or could have "hit points" against point loss, or could control their own allegiances more effectively, or could get foreknowledge about the allegiances of others, or could limit access to her own data on the server, and so on.
- **Alliance powers:** Alliance membership could give perks as well. The ultimate in interactivity would be if alliances gained perks through control of spook zones outlined above, and the players in the alliance benefited from this. The game would be effectively a kind of human-interaction CCG.

In general, perks would help to differentiate between players, which increases gamer appeal. They are not however strictly necessary for the target audience, at least in the short run. If the game succeeds, they're simple to add.

Organizer events: Nothing prevents the game organizer from adding special events that change the rules for a time. The possibilities here are various and depend greatly on the local scene – how popular the game becomes and who plays it.

For example, GPS-located game tokens could be used to create catch-the-flag scenarios. Especially cool if the token is big, colorful and in the form of a sponsor brand symbol. The target audience likes nothing better than running around the city with a ludicrous, giant foam hamburger ;)

Subgames: Linked to perks and organizer events, software subgames are easy to implement in the design. As an example, a perk could specify that the player is "armed and dangerous". When such a player is successfully caught, she has the option of "going peacefully" or "fighting for it". The latter would be a short minigame, and success would mean no point loss or gain for either party, as the spy escapes from the enemy. Losing, on the other hand, would close the player out of the game for 24 hours, as her spy identity dies in the escape attempt. At its simplest, the minigame itself is just a round or two of rock-paper-scissors, but the options are of course limitless.

The server dimension: The most ambitious option in expanding the concept is postulating a whole another, internet based game that can be played as a complement to the real-world game, *Matrix* style.

In this model the activity database is web-accessible, and players can log in to check out what's happening. They can also become "operators" for the players on the streets, sending them messages and reporting on developments. A significant advantage.

Unlike the basic rules, the data put out by the players goes through their operators, and is "secured" in different ways, just like outlined in an above entry. However, the game for the operators would be to find out data about the opponents and hide data about their allies.

The operator players would have allegiances and would score like the other players, but instead of making contacts on the field, they'd play a game of information exchange with each other. There'd also be subgames for taking data with force. The main question would then be, who the operator helps? They wouldn't know the allegiances of other players either, except through gaining that data from the system.

The two modes of the game would be quite equal, with the same player able to play either, even at the same time (just call a meeting somewhere with web access). The social situation could become interesting, with the "operators" much better informed than "field agents", but the latter having to actually risk exposure on the field. Just like a modern spy movie.

Designing for the target audience: Kind of obvious, isn't it? The game is built for success, becoming a kind of sport, as far as popularity is concerned. This is not so much a game but a potential hobby.

Eleanor's Dream

This is a weird platform game based on my love for sprite graphics. It has some similarities to the NES platformer Little Nemo and the PC adventure game Out of this world, but at the core is a game of fundamentally different values.

Target audience: From children up, actually anybody. You cannot lose in the game and don't need much skill at all. The subject matter should attract just about everybody. It might be a little too girlish for some boys, but that's their loss. Perhaps requires reading skill for full enjoyment.

General concept: A platform game where you cannot die and aren't trying to get to the finish. Instead play moves through wonderful different terrains and the point is just exploring those. The worst hindrance for getting somewhere is having to jump or climb in some places. Really. The challenge lies in getting to someplace definite.

Details

Technical framework: 2D platformer, as large levels as humanly possible. That many levels, as well.

Theme: The player plays Eleanor, an inquisitive, sweet child. Every night she dreams and gets transported to the Dreamland, where she has all kinds of adventures. The Dreamland is a colorful, wondrous place, quite different from Eleanor's waking world. Her adventures in the Dreamland define in the end how she'll end up in the real world: will she find her heart, or become grey like some do?

The Dreamlands are full of wondrous landmarks, like giant mushrooms, dwarves and faeries, dragons, aliens, the Large Door On the Plain and so forth. No sense at all, but very poetic and pleasing in itself. The game is really all about theme.

What play looks like: A platformer set in a fantastic milieu. Eleanor moves through the world in a leisurely platform game, coming to all kinds of places. There are some dangers, but they don't kill and force you to start again: instead, Eleanor gets transported to a different dream. Falling into a hole leads to an underground dream with dwarves (the levels are indeed as graphically intricate as feasible), getting bitten by a fish leads to a water dream, and so on.

Play experience: The main mechanic is the Dream meter, which measures how much dream Eleanor still has left that night. Every change of dream spends a little bit, and time spends it as well. A single session of play takes at most fifteen minutes or so, after which Eleanor wakes up anyway.

When a play session starts the player will answer a multiple-choice question about Eleanor's day. These range from questions controlling her actions to questions controlling her fate. "What did Eleanor meet on the road today?" for example, or "What Eleanor thinks about the cranky old lady who lives upstairs?" All the answers given by the player at the start of sessions are saved in the game state, so a kind of a profile of answers can be generated.

Actual play starts after the question is answered. Eleanor appears somewhere in the Dreamlands, and the player can direct her wherever he wishes. At first the player will likely just want to explore, but after playing the game some time he'll find an idea about where to go. Then the game gains a kind of a challenge: find the place you want to go to before the Dream meter runs out.

The general platform play is however as dreamlike as possible, with little repetition and many surprising developments. Especially modes of travel should be considered. Eleanor could possibly have a flying ability from the start.

The key ideas of the game are all structural. I'll outline them from the design perspective below:

The Dreamlands are a set of interconnected, strongly themed levels of the game. After playing a while the player can usually recognize which Dreamland he is in. There is no hierarchy

between the lands, the only thing of importance is where Eleanor wants to go. The player can find faster or safer ways from one land to another by experimentation.

The starting position of Eleanor in the dreamlands is always defined by the answer player gave to the question at the start. There are a multitude of starting positions, and it's quite possible that Eleanor starts just beside the thing she wants to see.

Advancing in the game is tied to two things, the answers the player has given to the questions at the start, and the current night's dream. The answers are the only thing saved in save states, Eleanor starts all nights otherwise identically. When the player has played enough sessions (answered enough questions) the residents of the dreamland start talking to Eleanor, and the questions start becoming deeper, perhaps vaguely threatening some times.

There's all kinds of things the residents of the dreamlands tell Eleanor (not quests or anything like that, just colorful, poetic images), but the only actually important thing is that there exists a number of endpoints for the game (obviously not called that, they each are considered separately) in the dreamlands. These are places like the Heavenly Closet, where someone says Eleanor should go to speak with the Queen of Heaven. Or the Mushroom Squire, in the deepest part of Boggly Woods.

These endpoints all are in remote, but not particularly difficult, areas of the dreamlands. Most of them, however, will require some rational planning, because there's a time limit in the form of the Dream meter. The player will have to try to get to start from a good place, and perhaps solve a simple hardship. Like, when trying to get to the Mushroom Squire, perhaps she has to first get some gold from the dwarves for the ferryman. Really simple, just a tad more demanding than aimless wandering.

When Eleanor finally reaches one of these endpoints of the game, she's given a question about something. If the player answers "correctly" (in a way pleasing to the questioner), the game ends with a short story about Eleanor's fate.

The different endpoints indicate different futures for Eleanor, in a kinda poetic and hopefully touching fashion. She can take the offer of the Queen of Heaven to stay there with her forever, in which case she might become a church deaconess as an end to the story. Or she could become the bride of the Black Knight when he proposes, which could again mean something else in the ending story.

An alternative ending is if the player doesn't find any of the endpoints. Through multiple games the dreamlands get somewhat darker and the residents start to give simple, foolproof advice about how to get to one of the endpoints. If the player desists, one day when starting the game, the player is informed that Eleanor has grown up and joined the grey, triggering a short story.

The lesson is that there's no particular goal in Eleanor's Dream. The player gets to explore the dreamlands and to get to know Eleanor through the session-starting questions and her dreams. The exploring is rewarded, when the player gets a wider choice of endings. The choice of ending is simply a thematic statement from the player, one ending is not better than other. The player might play the game multiple times to see different endings, as well.

The game has a clear dramatic arc in the form of the question system, as it simply ends when the game is played enough times. A suitable number of sessions is probably somewhere over twenty but under forty, depending somewhat on the extent of the dreamlands.

Further elaboration

A modicum of traditional adventure functionality can be inserted, as long as its point is making exploration easier rather than segmenting the world like adventure games do usually. For example, Eleanor could awaken a golem that follows her everywhere just because, and helps by breaking walls. The walls could be climbed over anyway, but this way is a little faster. And the golem goes away at the end of play session, and has to be awakened again the next time. The only place where absolute adventure logic is allowed are the endpoints at the end of the game.

The dreamlands can be made more dynamic by including some changes that happen when the time goes by. The entire topography of some levels could change in certain situations. Possible triggers include number of sessions played, answers to some key questions, actual time or date and actions Eleanor does in the dreamland.

More generally, the game could randomize the general layout of the dreamlands when first started, so that each play-through has different platformer problems. Graphics stay the same, but the structure of the levels changes enough so exploration is required from veterans of the game as well.

Designing for the target audience: What makes Eleanor's Dream attractive is the gameplay. From the start the computer games have been built in a hierarchical, performance centered manner. To enjoy a typical action platformer, the player has to have a high hand-eye coordination and the kind of disproportionate motivation that's so typical of gamers. A normal person can hardly be expected to train dozens of hours to master a game.

Eleanor's Dream strives to sidestep these issue of the genre by removing any obvious winning conditions and making progress a matter of interest. The game becomes entertainment without losing in interactivity when the focus of the design is turned to the experience of playing through the game, instead of the challenges therein.

Such a game is the perfect first game for young children, as well as adults who never had the patience for the hypercompetitive games. The game's dreamlike theme attracts both children and adults.

Wizard Jamboree

This is the game I finally chose for my work application. Wizard Jamboree is simply a party game. For some reason there are no really good party games for mobile platforms, despite the order of magnitude increase in functionality. A really good party game is one you can literally play in or alongside a party, and this is much more likely to happen, if you can carry the game console around.

Target audience: Occasional and diversionary gamers. As is usual for party games, this probably won't appeal to hardcore gamers. The game is supposed to be played with friends or acquaintances as a complement to a relaxed party or social evening.

General concept: A compilation of microgames, party games and party accessories bound together by a program personality. The game is played through a party, and is only terminated for scoring when the party ends. Players can change wildly without the game stopping, and the role played by the game in the party can be customized by the players.

Details

Theme: Light, funny absurdity. The game centers around The Wizard, a pompous oracle character that wants to advise the players in having fun. The Wizard realizes that he's a program, and thinks of himself as a "utility program, a 'party organiser' perhaps, but certainly no common game". The unity of events in the game is supplied by the party going on meanwhile, so there's not much internal theme apart from The Wizard. The medieval vocabulary is an affection of the character, and doesn't show up in actual content.

What play looks like: The participating players do something else, chat or cook or dance or whatever, while one player plays her turn. The mobile is rotated from player to player as their turns come around. From time to time a player might exclaim in reaction to the game, or the players gather to play a microgame all at once, but otherwise the game is very restrained, not interfering with other entertainment. The game is started at the beginning of the evening and players come in or drop off freely. At the end of the night the game is finished with a final round of turns, which determines the winner.

Play experience: When a player first comes into the game, The Wizard asks her name, sex, age and one other profiling question (hobbies, whatever). From then on, The Wizard will from time to time want to "talk" with the player in question, in which case the other players give her the phone.

What The Wizard will be about depends on numerous factors, but most likely it will be a microgame The Wizard wants to play. These are very simple reaction, puzzle or just guessing games, lasting as little as five seconds. Additionally, The Wizard frequently asks all kinds of questions ("Are you having fun?", "Isn't there really any girls in this party?", "Which of your fellow players do you like best?", that kind of thing), most of which it uses to tailor it's behaviour.

In addition to the microgames, The Wizard might give "quests" or "oracular advice". The former are bigger games played by all the participating players (or as many as can be scrounged up at the moment), while the latter are simply suggestions The Wizard has for running the evening and the players' love life.

As the evening progresses a given player might play as few as five or as many as fifty turns with The Wizard, depending on the players. All the while The Wizard gathers information about the party and the players, and adapts the games it offers to that information. The microgames and replies the players give to The Wizard are scored according to a complex formula. When the game is finished, The Wizard will award the players with "prizes" according to it's whims.

Overall, the player experience is one of bemusement with the ridiculous game program. The game is not played for performance at all, but rather as pure amusement, to see what the program

comes up with next. Many of the things the game does make no sense at all, which is part of the entertainment.

The microgames are very simple, mostly one button affairs. They center around pressing the button at the right moment, or figuring out what the game is about before time runs out. In no case will they last more than ten seconds. Players of *Warioware* know what I'm about. The typical microgame might be about moving a cocktail glass around to catch dropping ingredients of a cocktail mix, for example.

When the game progresses, The Wizard gauges how well the players do in the games and corrects the difficulty levels accordingly. It makes a big point out of displaying all kinds of statistics about how the players stack compared to one another.

The actual point of the microgames is to provide structure and surprises for the game, and to give some breathing time to the players. It's the traditional game component of the whole, and necessary to root the experience. The amount of microgames compared to other content depends largely on the kind of party and amount of players participating, as well as the response of the players to the different things the Wizard proposes.

The quests, on the other hand, are the high points of play. Instead of offering a microgame for the player, The Wizard commands her to gather the other players around for a quest. These are not played on the device at all, but rather between the players. Some examples of what the quest might be:

- Similar to "Spin the Bottle", the Wizard might pick one of the players for a funny task or a semi-embarrassing question. "Jonathan, crawl under the table! Pronto!" or "Lisa, tell us, would you, who's your sweetheart?" Enforcement is of course handled the same as in Spin the Bottle, i.e. not at all. The Wizard will want to know if the "Chosen One" did what it asked, though...
- The Wizard knows dozens of party games, one of which it might ask the players to play. These are stuff like Lapp Bluff, the abovementioned Spin the Bottle, All on One Side, Baby If You Love Me, I Never Did That and so on. The genre is well defined, so there's all kinds of options for The Wizard to pick, based on the age and sex of the players, as well as the type of party. The Wizard instructs the players in detail, and will want to know the results of the game when it's finished.
- There's a variety of computer-assisted party games as well, that is, games that can be played by a large group without everybody seeing the screen. Some examples include different word games, trivia quizzes and voting games. The Wizard might assign teams, or everybody could just be on the same side against The Wizard.

When The Wizard declares a quest, the players can freely decline to obey it's frenzied commands. The Wizard might sulk some, but soon it'll come up with something else.

Oracular advice happens when The Wizard has collected enough knowledge about the party and the players. In these situations it will reveal it's utmostly logical reasoning to the players. Most likely the advice is hilariously incongruous, like an overwrought horoscope.

There's many kinds of possible information The Wizard will collect and disseminate. Here's some examples:

- When the game is started, The Wizard will initialize itself by asking about the type of party ("Social Evening", "Wild Party", "Goin' Out" and "Just Playing", essentially) and it's particulars (alcoholic or not, friends or just acquaintances, etc.). It'll use this information to guide it's suggestions during the game.
- The Wizard tries to pry details from the players as well, asking about their hobbies and friends, as well as about how the party is currently going ("Paul, old pal, is anybody still awake? Should we be quitting for the night?").

- The Wizard uses it's knowledge about the state of the party to offer "appropriate" advice to the players; this might be drink mixing hints, dance step lessons, music suggestions, advice on approaching the opposite sex, conversation topics, party food recipes or something totally unexpected. Mainly just color, but basically solid advice.
- From time to time The Wizard will also make oracles about the players, whether it's gained enough information or not. It might reveal who loves and whom, or what this particular player will be when he grows up, or how long each player has to live, or what a player's like in bed. These are somewhat controlled by the answers given by the players, but mostly the point is just to throw up strange, personal stuff, like a Cosmo questionnaire. The declaration might be privately to one player, or The Wizard might tell all.
- The Wizard might act as a pollster on behalf of a player or of it's own volition, asking the same questions from all players and then making charts of their answers. Questions like "How fun is this party on a scale of 1-10?" would be interesting for the host, especially if it'd be inappropriate to ask directly. And people are interested in opinions in general...

Game Structure: All in all, the Wizard is closer to an agent in the sense of modern software design; a senile and weird one, but regardless it has a number of functions pertaining to arranging different kinds of social happenings. It can act as a kind of a master of ceremonies, instructing and directing the party based on input. When the game is running the players can anytime interrupt the microgame cycle to ask for one of the other functions, like a hidden vote/poll, a party game, some party hints or other stuff The Wizard offers. Essentially Wizard Jamboree is the answer to the question of "What're we gonna do next?"

Ultimately, the Wizard Jamboree could be kind of a ironic joke in a very trendy sense. One could quite well arrange a "Wizard Jamboree" where the whole point is to do what the insane codger says. ("It wants us to play 'Light as a Feather, Stiff as a Board'? WTF!?!") Alternatively, it can supplement the evening to the degree wished for, to the extent of functioning as a microgame collection for single player play, dull as that is compared to having fun with friends.

Designing for the target audience: This is a very narrow design to begin with, and obviously targeted at trendy, intelligent and social young adults. The people who understand irony and doing stuff "just because". The kind of person who'll dig the Wizard Jamboree will be a party buff anyway, so it's probably prudent to market it explicitly as a party game, meant for five people or more. It's no stranger than selling halloween props for a theme party.

Technical framework: A mostly textual game, but also a bunch of sprite-based realtime microgames. Mainly hotseat multiplayer (from three players up to a couple of dozen, even). Some pretty heavy data tables and relationships, but that's a problem for production, not technology.