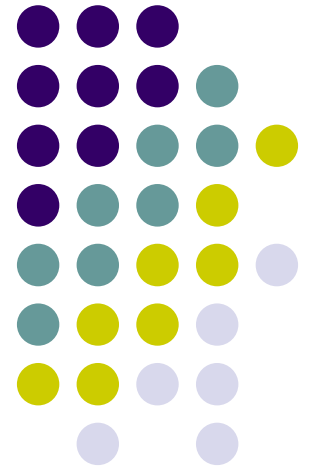


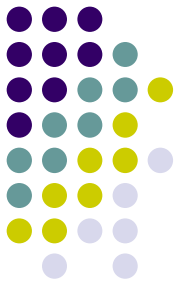
Prototyping as a game design tool

IT-University of Copenhagen
October 31, 2006

Simon Larsen



Agenda



- Who am I?
- What I'm working on currently
- The EVE method
- Sharpen your profile (if time permits)

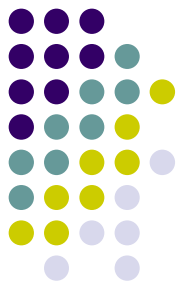


My background

- I'm a student at ITU... just like you 😊
- I'm currently working on my master thesis (together with 3 other guys)
- We are working the master thesis together with IO Interactive here in Copenhagen



What else...?



- Film industry in the 90s
 - In the mid 90s and early 2000s I worked as a film critic for "Levende Billeder" and "Jubii Film"
 - In the late 90s I worked as a film producer (in Århus)
- Game company owner
 - From 1998-2000 I co-founded a game company (never finished our game !)
- Post mortem analysis
 - I made headlines in 2001 with an analysis of the post-mortem articles from Gamasutra.com

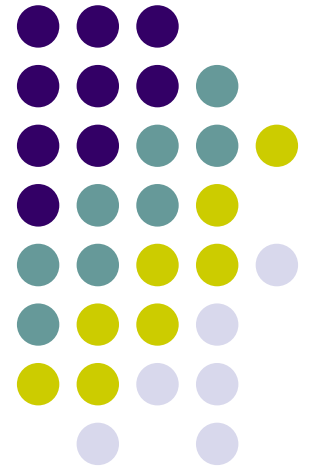
What else...?



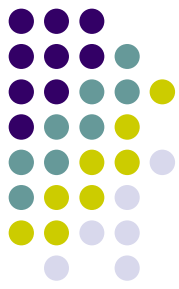
- Level scripter
 - I just quit my job as level scripter at GuppyWorks, to work fulltime at my master thesis.
- Nordic GameJam 2006
 - I won (together with my 3 thesis partners) the jury and the audience award at Nordic GameJam 2006 with the game Pen & Paper.
- Personal life
 - I'm happily married and have a wonderful son aged 1½ years.
- LinkedIn
 - Look me up at LinkedIn (<http://www.linkedin.com/in/simonlarsen>)

What I'm currently working on

Insights on the master thesis



Thesis background



- Facts
 - Game production is taking longer time
 - There are more people working on them
 - The productions are becoming more expensive
 - The market is becoming more competitive
- Hence the risk of making games is becoming much higher

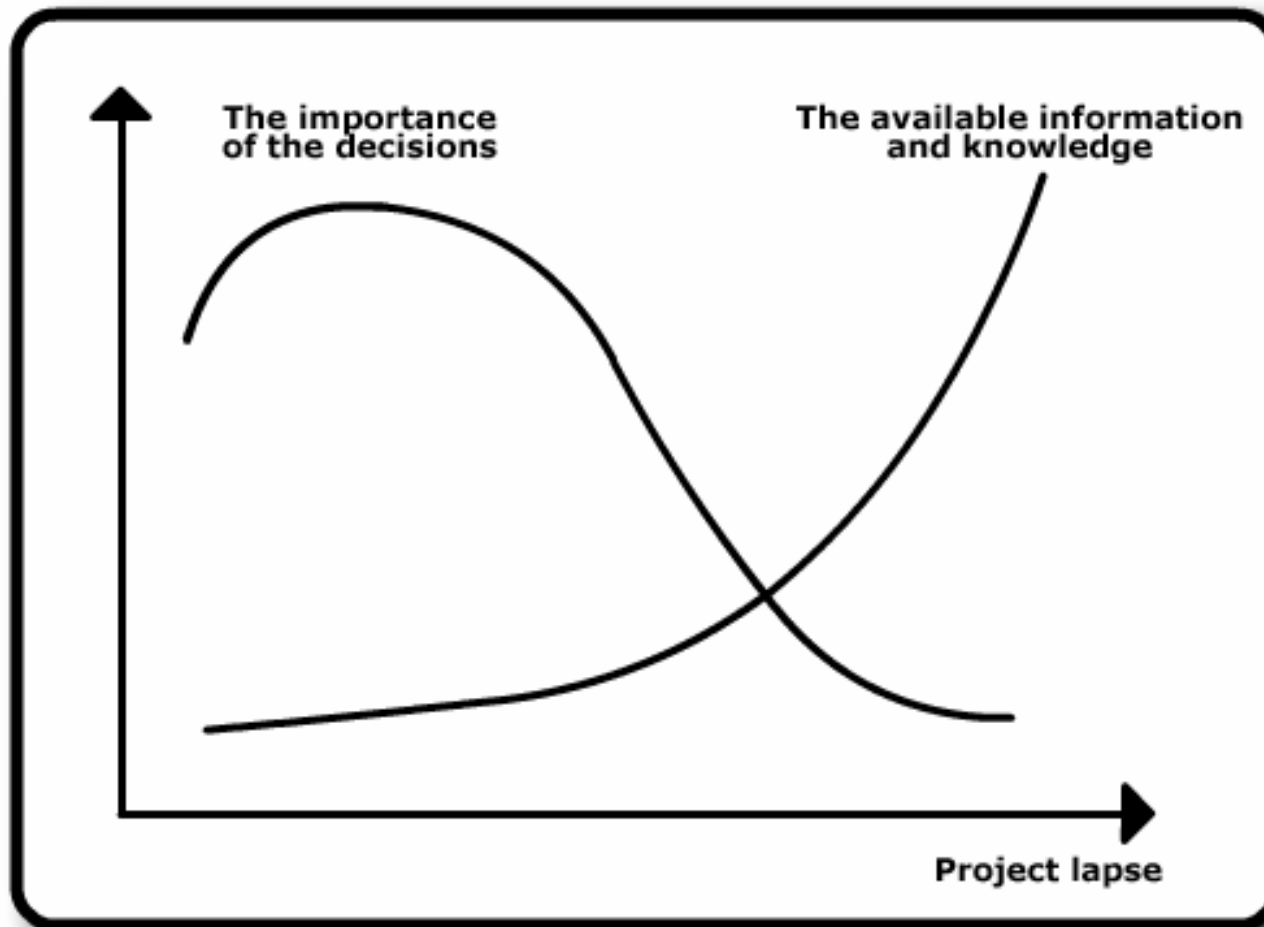




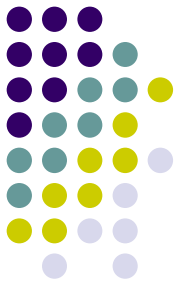
Thesis background (2)

- There is a need for ...
 - ... creating a way of testing your ideas
 - ... exploring new ways of creating games
 - ... developing games faster
- The game development industry is standing at a cross-road
- The time to change is now

Contextual uncertainty



Solution



- Reduce the uncertainty and thereby reduce the risk
- Simple, right?
- Right...?!?

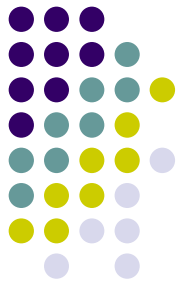




How to do that?

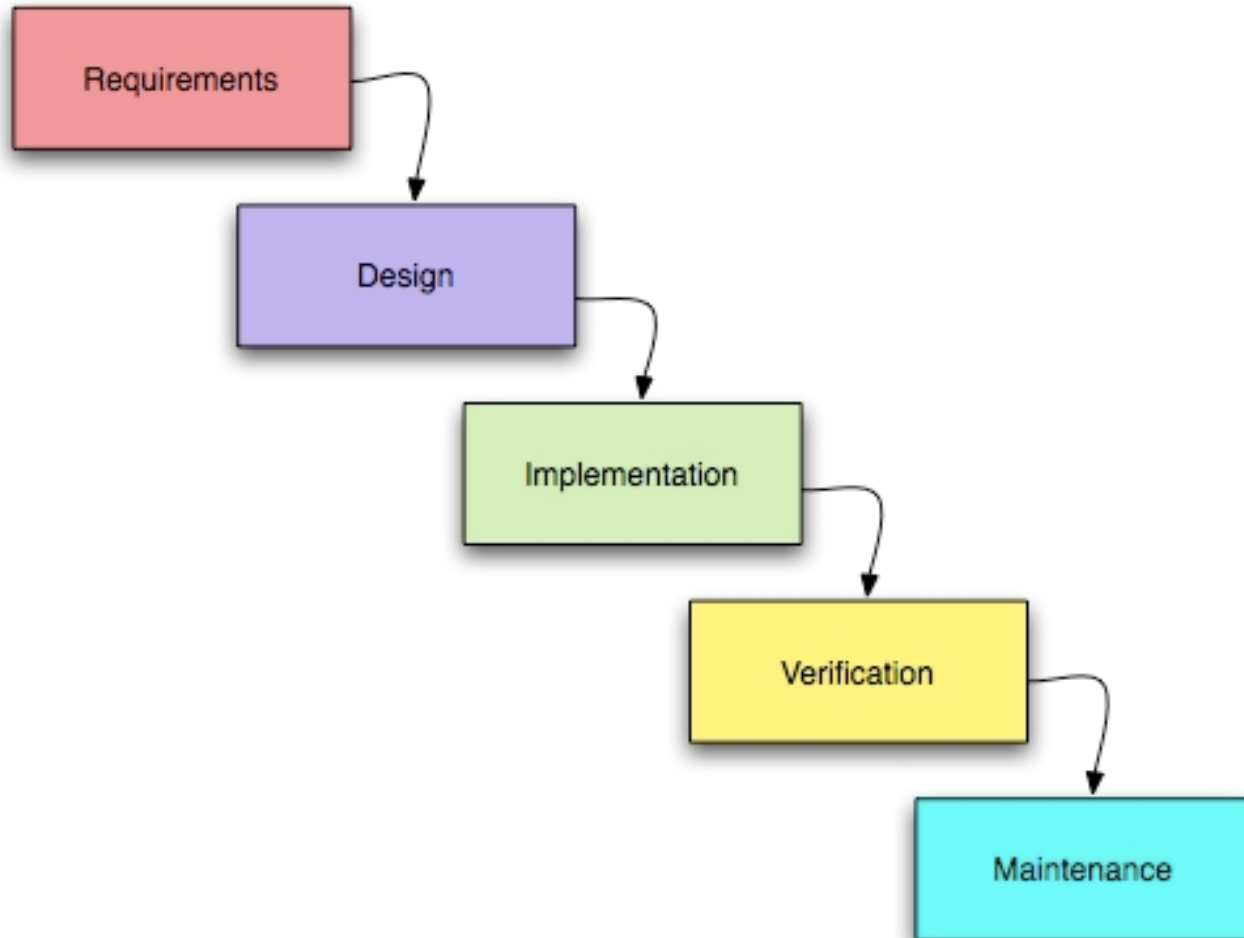
- Prototype much earlier than you are used to
- Fail and fail again
 - Failing is good (if you reflect on why you're failing)
- Prototyping = knowledge gathering
 - Prototypes != first playable or alpha version
- Delay decisions until more information is at hand

Big Up Front Design



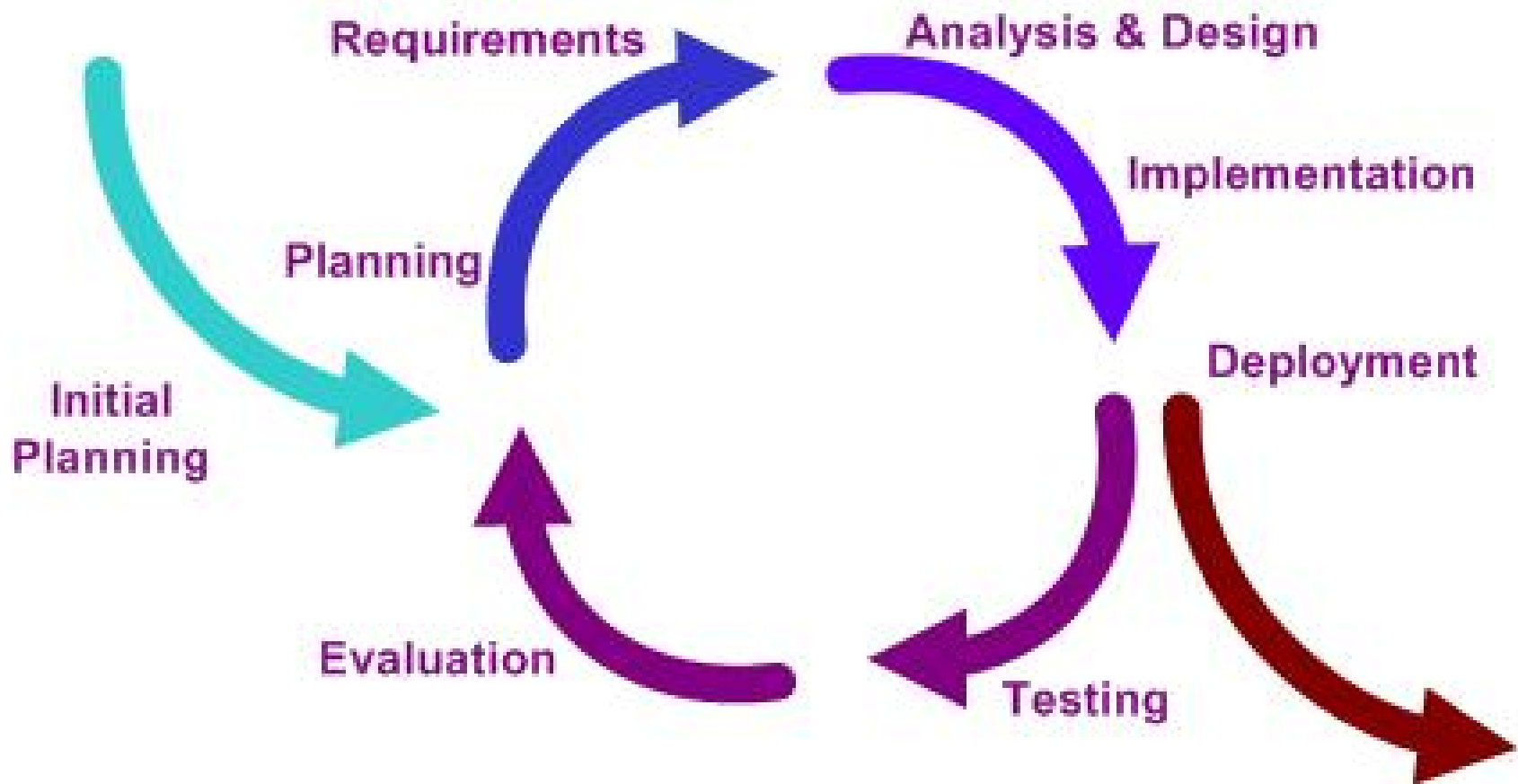
- Write a large design document with all features and elements of your game *before* doing anything else
- You always end up with the game you designed on paper in the first place
- This is the “classical” way of making software products and games

Waterfall development





Iterative development



High concept vs. mechanics

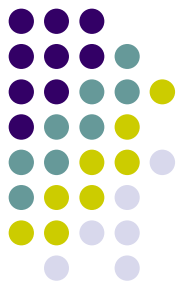


- Do not
 - Write the background story first
- Do
 - Identify the core game mechanics or core features of the game first
- Stories are not that important anyway (!)

Gameplay elements



- What would Max Payne be without *Bullet Time*?
- What would Half-Life 2 be without the *Gravity Gun*?
- Could there be Bullet Time in Half-Life and vice versa?
 - Max Life ?
 - Half-Payne ?

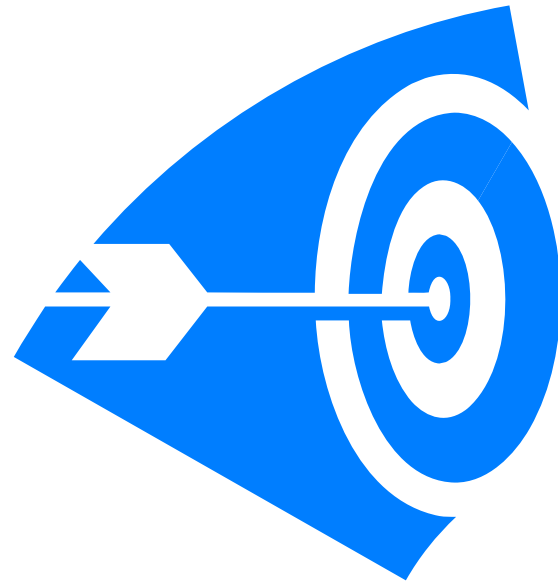


Just do it

- Ready! Aim! Fire!

vs.

- Ready! Fire! Aim!



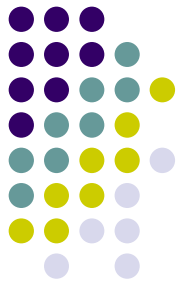
“You miss 100 percent of the shots you never take.”
Wayne Gretzky (NHL player)



Verification vs. validation

- **Verification** verifies that the final product satisfies or matches the original design (low-level engineering).
 - Built the product right
- **Validation** validates that the product design satisfies or fits the intended usage (high-level marketing).
 - Built the right product

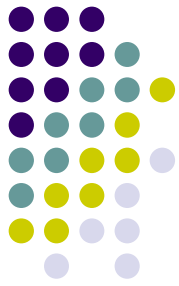
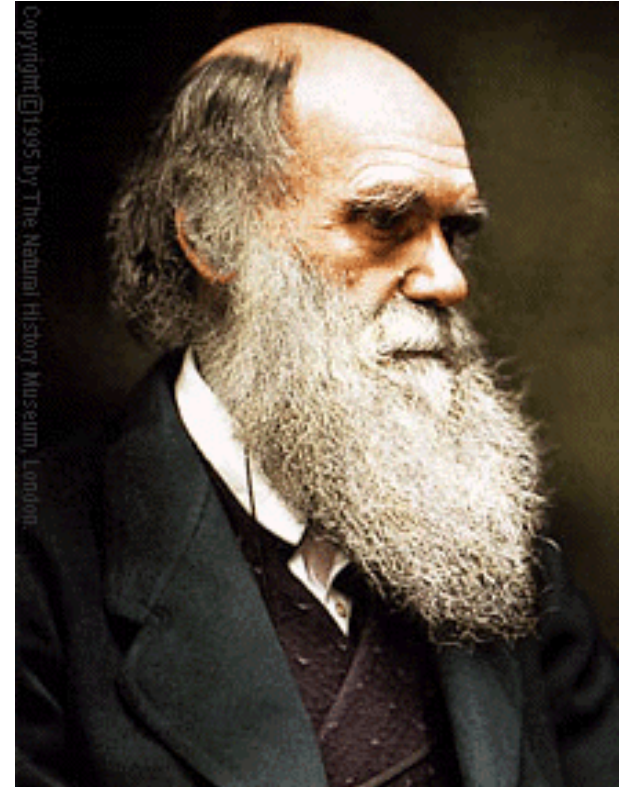
Embrace failure



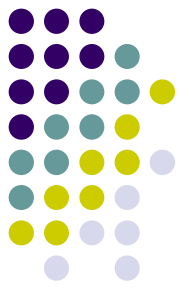
Intelligent Designer

- “It is not the strongest of the species that survives, nor the most intelligent that survives. It is the one that is the most adaptable to change”

- Charles Darwin

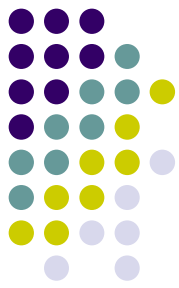


Google's new-product process



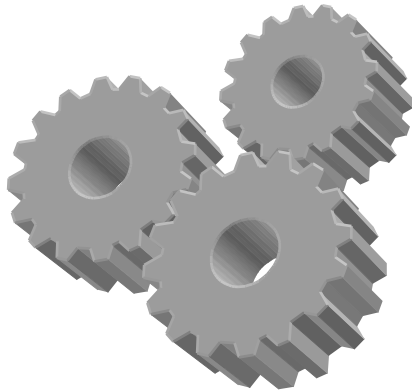
- The philosophy is; try a bunch of ideas, refine them, and see what survives

Google™



Built the toy first

- "Start with the core mechanic. Whether spring systems, swarm behavior, gravity, etc, it never took more than a few hours to get the basic theme up and running. This "toy" should be the core mechanic of the game minus any goals or decisions. There is no win or lose state, just a fun thing to play with. "



(The Experimental Gameplay Project)

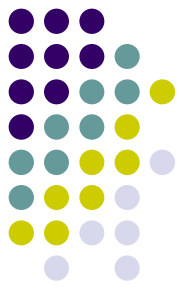
Abstraction



- Make the prototype abstract
- Ask my son what this LEGO creature says
- It doesn't have to be good looking to be good



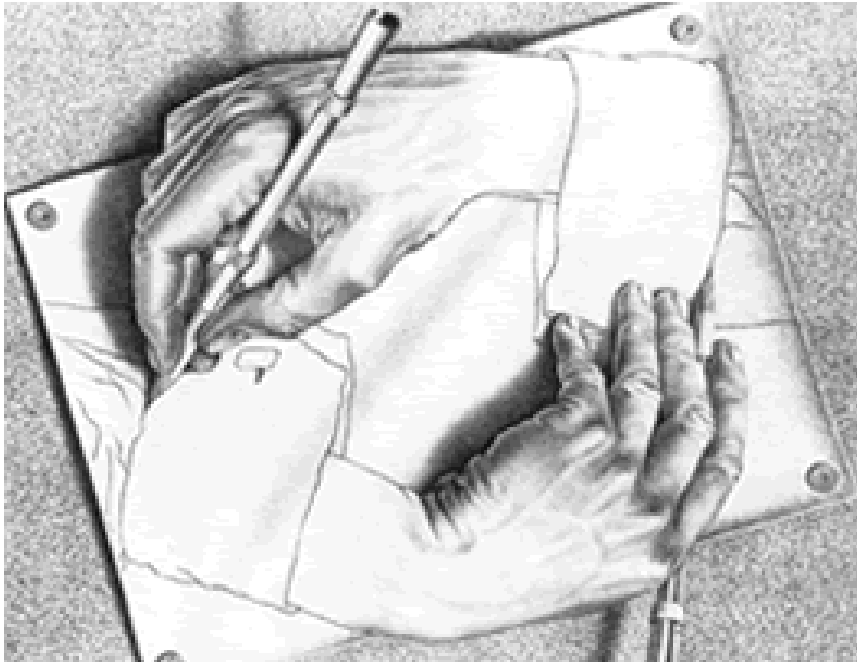
Simplicity



- The eXtreme Programming value:
- *“Do the simplest thing that could possible work”*

- Simplicity is not a very simple thing
 - It’s not “The fastest thing to make”
 - It’s not “The first thing that springs to mind”

Simplicity (2)



Ernest Hemingway's
best work:

"For sale:
Baby shoes.
Never worn"

It doesn't have to be complex to be good

The customer is always right



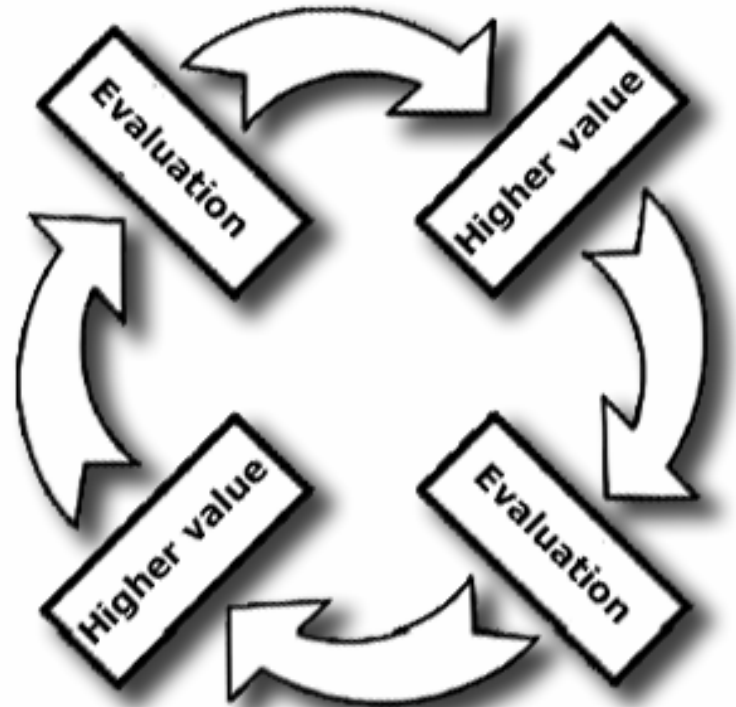
- Play-test, play-test again and then play-test some more
- Take the results from play-testing seriously
- It's not always the stupid testers you get, sometimes there actually *is* something wrong with your game



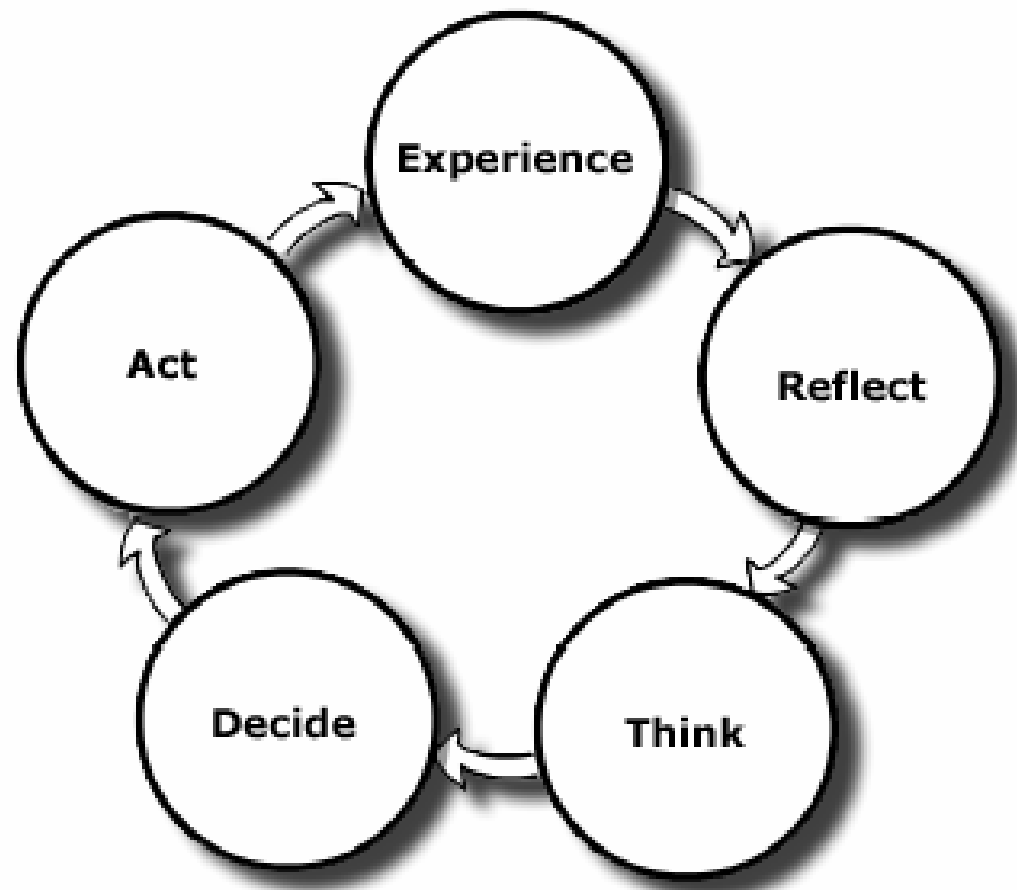
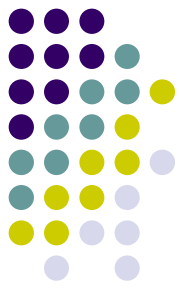


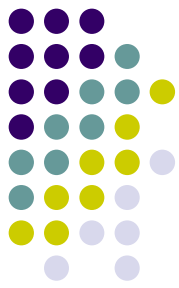
Constant evaluation

1. Release often
 2. Get feedback fast
 3. Adapt to the feedback
- Repeat from the top...



Metacognition





Rapid prototyping tools

- XNA Game Studio Express (beta)
 - SketchUp (free 3D tool from Google)
 - LEGO
 - GameMaker
 - Blitz Basic / Blitz Max
 - Flash
 - PyGame
 - C++ / OpenGL
 - C#
 - Java
 - Blender
 - Pen and paper
-
- Anything you can work comfortably in, *fast*.
 - *Must* be flexible. You must be able to create any kind of game in it.

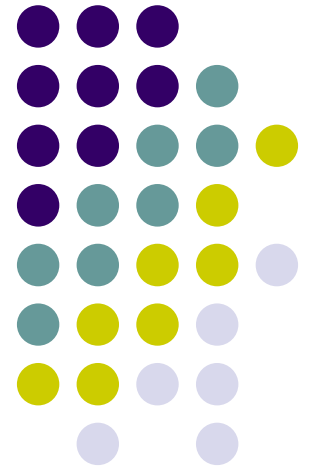


The EVE Method

Experimentation

Visualization

Evaluation

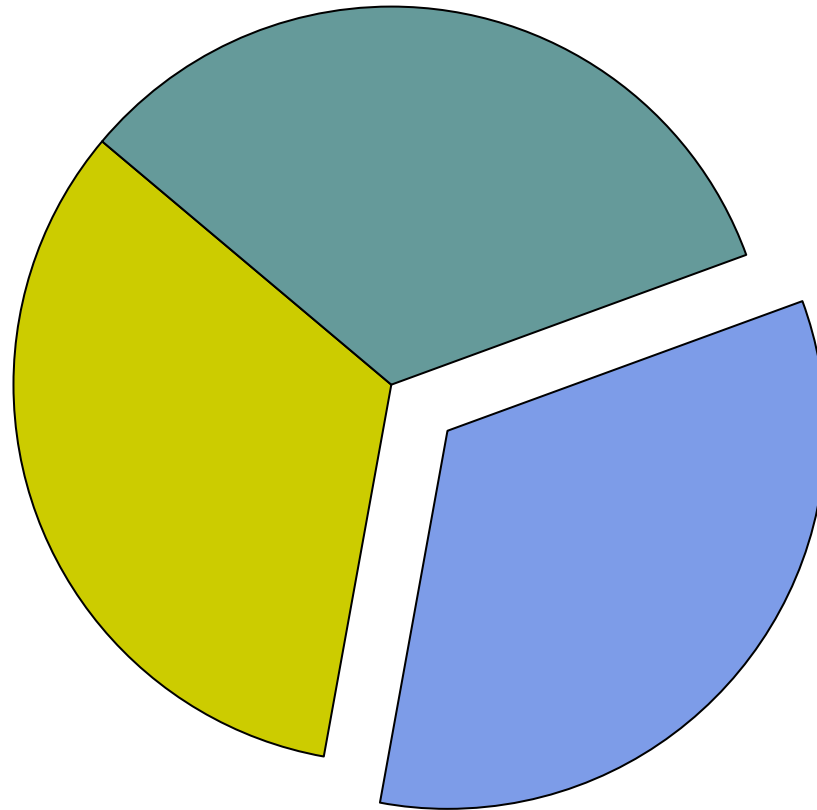


Disclaimer

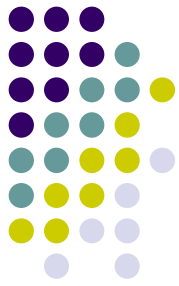


- This is a work in progress
- This is only *my* personal view of our method
- Currently the method works better for some genres than others

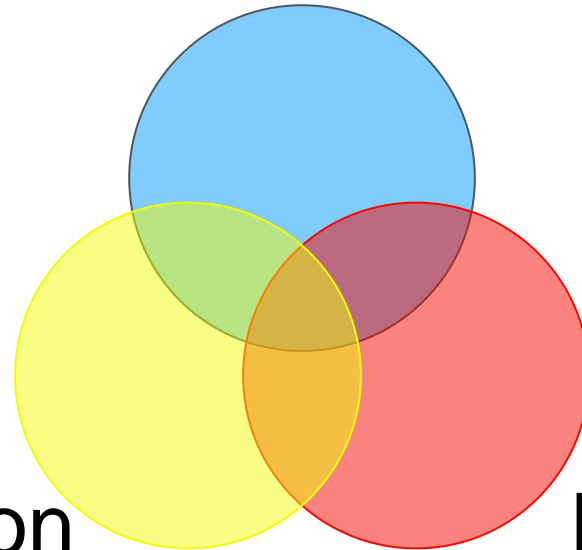
Focus



The EVE method



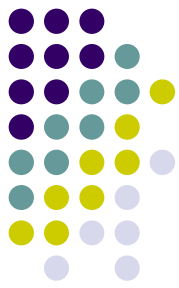
Experimentation



Visualization

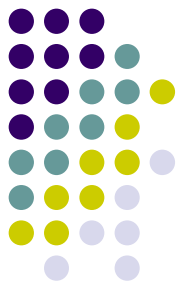
Evaluation

Experimentation

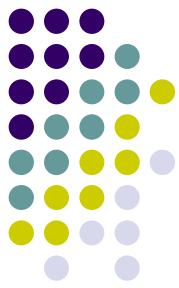


- Failing is good
 - When you learn from your mistakes.
 - Metacognition
- We do not encourage experimentation - we expect it!
 - Experimenting in a “safe environment” leads to little waste

Experimentation (2)

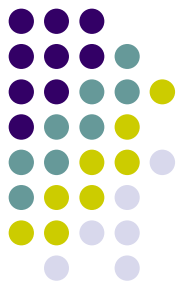


- Time limitation sparks creativity
 - Non-moveable deadlines makes the creative process much more manageable.
 - Time-boxing
- Try multiple solutions
 - Do not settle for the first solution that seems right. Create more.
 - Ready! Fire! Aim!



Visualization

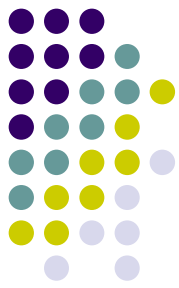
- Prototype does not mean "first playable"
 - It's a prototype not a final game
- Playing is believing
 - One prototype says more than a thousand images.
 - Make small design documents (Post-It notes?)
- Keep it simple
 - Simplicity is a virtue not a lack of creativity



Visualization (2)

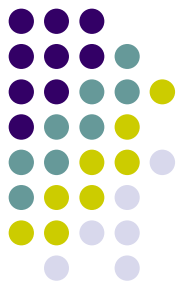
- Find out the key elements
 - If you can't find it you have already failed
- Keep the scope small
 - Much easier to test the elements individually
- Make abstract prototypes
 - Save the details for the final game
- Modular prototypes
 - Design every element so that it can be easily replace by something else

Evaluation



- Be proud of your work
 - If you don't show it, you don't get feedback!
- Play-testing is good
 - ... so do it all the time. Play-test yourself, make your friends test it. Every day, heck even every hour!

Evaluation (2)



- Time is of the essence
 - ... so make the core feature(s) first
- Make throw-away prototypes
 - Don't expect your prototype to be part of the final game
- Find the boundaries and the core of your concept
 - Given the knowledge you have obtained in the visualization phase you should be able to do this.

What makes EVE unique?



- Hopefully nothing (!)
- We are just standing on the shoulders of giants
 - Why reinvent the wheel?
 - Looking at the experience from traditional software development and mapping that to game development

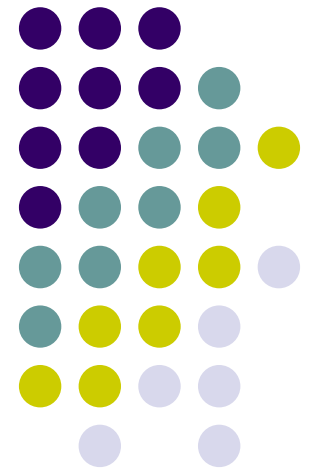
Games made with EVE



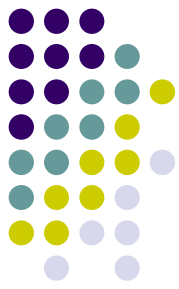
- Keyword: Buy
 - Unstable Companions
- Keyword: Sticky
 - Sticky Maze
 - Candy Stick
- Keyword: Visibility
 - Fade
 - Sponge-Bob and Patrick
 - DigiLight
 - TrackMania Levels

Questions?

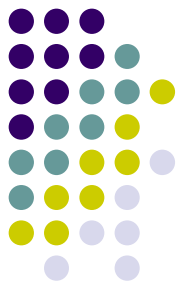
It's the end of the PowerPoint presentation and I feel fine



Questions ?



- What is the biggest challenge you're facing as a new game designer?
- What is your long term goal?
- What is your favorite game?
- What is your favorite genre?
- What is your favorite game mechanic?
- What is your least favorite game?



Thank you for your time

Simon Larsen

e-mail: simon@larsen.dk

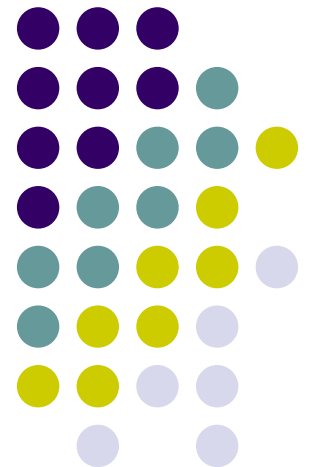
Website: www.blackwood.dk

LinkedIn: www.linkedin.com/in/simonlarsen

- Please feel free to contact me with any questions you might have.

Sharpen your profile

Advice from Harvey Smith and me



Harvey Smith



- I get asked for career advice all the time. I sent this to a university teacher recently, but it applies broadly. This same message hasn't really changed over the years.

Decide what you want to do all day. Programming, 3d art, audio, map layout, etc. Pick one. Get even more specific: Programming AI, building multiplayer action game maps, etc. Become great at that thing...so that managers want you on their team.

The idea is to become trusted and autonomous; a fire-and-forget team member, where people say, "If we put him on it, he'll make the right calls, autonomously, and it'll get done, and it'll be solid."

Be as technical as possible. Even if you have a focus on story, human interface, mission planning, or music, make sure you're as technical as possible; familiar with editing tools and software.

Be well-versed and well-rounded: Know not only your area (like software engineering), but be familiar with media theories, interface, psychology, sociology, various gaming platforms, social trends among gamers, general design, etc.

Look over the commonly recommended game design books: Rules of Play, Hamlet on the Holodeck, Understanding Comics, etc.

I love it when someone comes in, has passion for game design, is a great collaborator and communicator, can talk about Malcolm Gladwell or Scott McCloud, has read the Design of Every Day Things, has played a ton of games on many platforms, can program in C++ (or use Maya, 3D Max, UnrealEd, Source, et al), has written some fiction, can teach/lecture, has traveled, and is both strong-willed and willing to change her mind.

Quote taken from Harvey Smith's website (September 28, 2006):

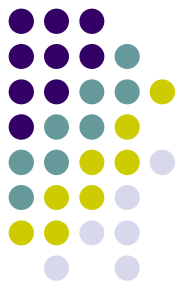
<http://blog.myspace.com/index.cfm?fuseaction=blog.view&friendID=22403116&blogID=172851080&Mytoken=27822AE3-6037-483B-9ECD62687A4969C136819483>

Harvey & me



- Be focused
 - Decide what you want to do all day. Nobody can use a "Jack of all Trades". Everyone is looking for experts.
- Be technical
 - Be as technical as possible. Learn to program! Learn at least scripting. EVERYTHING in game development is technical, NEVER think otherwise.
- Be well-versed
 - Be well-versed and well-rounded: Know not only your area. Play other types of games (even genres you don't like). Play bad games (figure out why they are bad). See movies, read books and comic listen to music. Make music, etc. Read up on engineering stuff even though you're writer/designer. Play around with sound editing programs and 3D Max
- Be active
 - Look over the commonly recommended game design books: Rules of Play, Hamlet on the Holodeck, Game Design Workshop, Understanding Comics, etc. Read Gamasutra.com daily. Become a active member of a forum for your favorite game.

Stop playing WoW and CS



- If you insist...
 - ... then at least compare them with other games in the same genre and try to figure out why they are so successful
- There are great works in areas you do not normally look
- Examples...

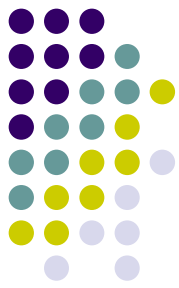
Examples (great shooter)



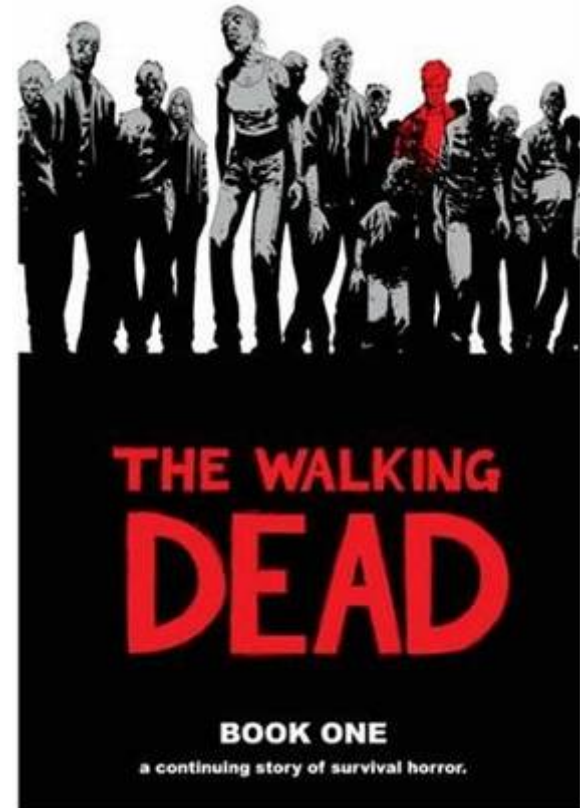
- Insomniac's *Ratchet & Clank 3: Up Your Arsenal*
- If you're going to make a 3rd person shooter – make it like this
- (Playstation 2 only)



Examples (great writing)



- Robert Kirkman's *The Walking Dead* series
- Zombies !!!
- Superb setting *and* narrative for a computer game



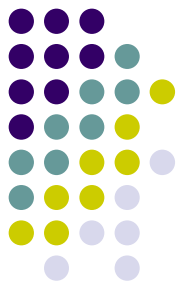
Examples (great design book)



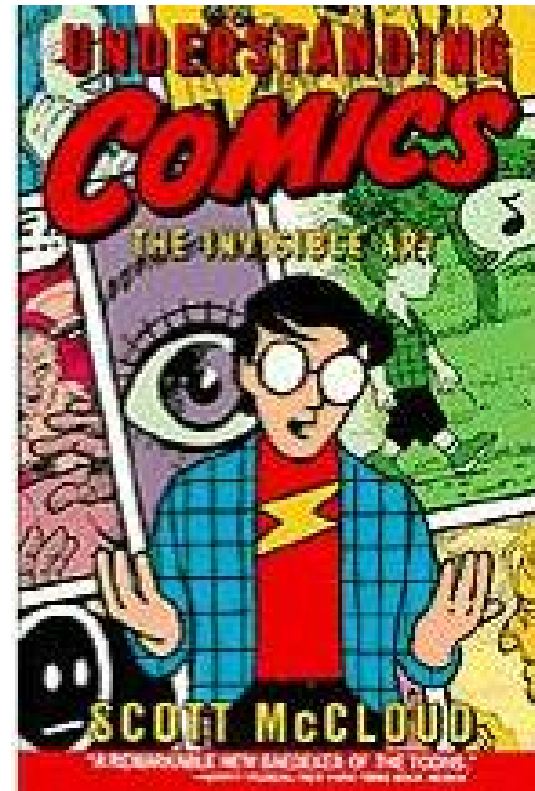
- Kevin Kelly's *Out of Control*
- Read online for free:
www.kk.org/outofcontrol/contents.php
- Very good source for simulation ideas
- Some maxims from the book
 - Complexity must be grown from simple systems that already work
 - A network nurtures small failures in order that large failures don't happen as often
 - We can only get smart things from stupid things



Examples (great analysis book)



- Scott McCloud's *Understanding Comics*
- About much more than just comics
- A MUST READ !



Examples (multiplayer game)



- Nadeo's *TrackMania Nations*
- Simple and superb level design
- Easy to learn... hard to master
- Free



Examples (great group book)



- James Surowiecki's *Wisdom of Crowds*
- Minor factual errors but great insights into hive intelligence
- Very easy read

