Group 1:

Group 2: Megahard doors

Fight Stigma
Software science students are normal students

Minigames
- Logic
- General computer skills
- Networking

Software science is fun
- Student life, drinking partying

Start
White screen “what happened last game? What were you thinking?”
- Determine what character you are
- Character creation by logic

Awesome hacking skills
- Fast typing
- Hacking world castle
- Facebook hack
- Explosions and shit

Go to school through networking
- Make VPN
- connection with TU

At school you get an A
- You find out you're not a robot-student-space bear but a normal girl
- Time to party!

Dance and shots
- Dance mini game
- Get more drunk and black out
- Back to beginning

Feedback
Not serious → serious
Start off with character creation → twist (main message?) → different minigames should play into this. In between games should say something about getting better.

Group 4: Abcdefghijklmnoqrstuvwxyz

Two electrical engineering, two software science students
Serious game, easy to understand, software is 'abstract', whereas harware is tangible.
Clear link to study subjects

- Electricity basics taught in high school
- Signals taught in first year EE

Game elements follow naturally

- Subjects should have elements that are inherently fun
- Homework has a puzzle-like structure
- Homework game

Student profile
High school with physics and science or Mathatmatics B + physics

Alumni Profile
Build
Precise

Key learning goals

- Knowledge, skills, attitudes
- Picked three of those

Teach something about logic gates

- Basics of signal processing

Concept art

- Memr- y mon
- AND-mon

Feedback
How to make it engaging

- Puzzle, but still homework
- Homework is not fun
- Intrinsic vs extrinsic
- The moment you know it is going to help you, you start to notice

Group 5

-
- Idea of the game
- Story line
- Game consists of smaller games from different courses
- Start out as an engineer with amnesia on a spaceship that finds out the ship is not working. All computer systems have failed and trhough simple ideas he has to get it working again. He has to solder basic circuitry in order to get the system to run again, then get the ship going in course through tsimple coding so that the automated guiding system works again.

Very important
Make it feel like the game came first and then the assignments, instead of building a game around all the assignments.

Tradeoff
If you start right away doing busy work in order to get to the long time goal, it can get boring, try go get them hooked a little bit before. Reach first goal very easy, before things start getting ugly.

Group 3: Frankly Three – The universe of software science

- 2.5D platformer
- Player modifies his avatar's code improve it.
- Explore space environment and solve puzzles
- Different levels for different topics.

How we make learning fun

- Teaching abstract concepts by concrete puzzles
- Lot of direct reward
- Pleasing aesthetics
- Scaffolding
- Self-determination theory

Self-determination theory

- Competence (feedback loops)
- Autonomy (finding knowledge)
- Relatedness (Customize robots, aesthetics and coding)

How to continue

- Elaborate logic and modeling level
- Find appropriate assets
- User tests
- Describe other levels

Feedback

- Good job, theories etc.
- Platform game → most are puzzle games. Makes it more difficult to align the learning content with the game mechanics itself.
- Intuitively know how a platformer works. Not associated with study. Play it, and then afterwards think about it
- Reply : we know users don't like to read.

Group 6: Metaforum: Quest to the second floor

Serious game to promote the Sofware science course

- Not a learning game, but a persuasive game
- The idea is that if the student likes the kind of game he sees

Concept

- Point-and-click adventure
- Minigames embedded in the 3D world for the learning concepts

Story

- The player is a student coming to the TU/e for a demonstration in Metaforum. There is a sudden snow storm that blocks the whole building. Lights are out. The player is able to call the emergency service from his mobile and he is instructed to go to the second floor of the building. Switches around the place with puzzles. Can't use the stairs, then you die. Turn lights on, activate the elevators to go up the floors

Feedback

Difficult for the player to understand, don't resort to texts and explanations.

Group 7: Project S.O.S.

A game about explosions, excitement, fun, learning.
Hard to explain in 5 minutes what Software Science is.
Meelopdagen are hard to take. A game should take this over.
Arcade game with minigames.

Focus on calculus, logic and set theory, and programming.
Create a computer assistent board, make exercises, let others solve them (community).
Program the way a line is drawn.

Future work
name, work out the games, game-prototypes, more courses, etc.

Feedback
Drawing is not an arcade games. Seems boring.
Where are the explosions?

Group 8

Concepts

- Control traffic lights (using logic gates)
- Control a subway system

Future ideas

- Particle effects
- Sound effects
- Limited number of logic gates per level
- Lots of different types of output
- Score (based on time and number of gates used)

Feedback

- Timer clock
- Reward experimentation: multiple ways

Our group

- How do we know which blocks are which?
- Click on a block
- Give them a color, if the color is red, you control the block.

- How about RPG mixed in with a bit of coding?
- Terminals, color coding for blocks you can control, editor, open-world etc.
- Autocomplete function (hints, etc.)

- Mechanics, mechanisms
- Aestethics are important, because that motivates you
- Main character should be someone you want to help, you feel related to.
- Make it such that in the beginning, you don't have to do programming, gradually introduce puzzles. Start of as a fun platformer.