Software project

8 Kingdoms

Presented by:

- Vojtěch Hlaveš
- Václav Krajíček
- Petr Zíta

Realisation team

- Project leader
  - Mgr. Marta Vomlelová Ph.D.
- Developers
  - Vojtěch Hlaveš – Resource Manager, Net, Editor
  - Václav Krajíček – GUI, Linux port
  - Petr Wolf – World aka game rules
  - Petr Zíta – AI, Editor
  - Pavel Žoha – GUI

Project goal

- Create turn-based strategy game
  - inspired by fantasy theme
  - with complex rules
  - with 3D visualization
    - inspiration: Warcraft III, Panzer General, Warlords
  - with multiplayer game (LAN and via internet)
  - with challenging AI
  - for Windows and Linux
  - complete, playable and entertaining
    - not only GUI, or AI engine

What would we do differently?

- first analysis
  - too ambitious
  - obscure for long time during development, changed (scratched out) almost every project meeting
  - didn't consider threats for different development areas
- bad work division and project members cooperation
  - bad work synchronization
  - good game designer and chief-developer missing
  - bad cooperation with external workers
- popular technologies don't have to be the best for your project
  - TCL
- absence of iterative development (development milestones)
- no third-party game frameworks usage

Final version imperfections

- unstable
  - result of development methods
  - little time for testing
- rules
  - many game features are not very important for player to know or use
  - unbalanced
- AI level
  - various problems solved by easiest way => AI is not very clever
  - not really challenging for any good player
- TCL usage
  - not used for what we planned before
  - mostly needless
  - slows whole game down significantly

What did we scratch out? (1)

- Some planned game features were found too difficult to implement:
  - heroes, artifacts and quests (difficult and hard to learn AI)
  - flying units (problems for rules and graphic visualization)
  - ships and water units (problems for graphic visualization)
  - weather influences (needlessly complicated)
  - dynamic game events driven by TCL
  - advanced kingdoms properties (lack of time)

What did we scratch out? (2)

- spells (difficult graphical effects, hard to learn AI)
- advanced diplomacy properties - threats, compliments, etc.
- complex rules for sight
- unit morale
- soldier's pay
- and many more…

What can we suggest?

- concurrent version system
  - CVS (SVN, SourceSafe, etc.)
- solutions evidence (+ its reasons) of game analysis
- Tasks, Bug Tracking System
  - good way how to find out what (and how) is every developer doing
- log system, exception hierarchy, memory debugging

Artificial inteligence

What are the AI tasks?

- defence of its towns and buildings
- attacking on enemy (enemy targeting, attacking units selection, attack synchronisation)
- building
- unit training
- resources handling

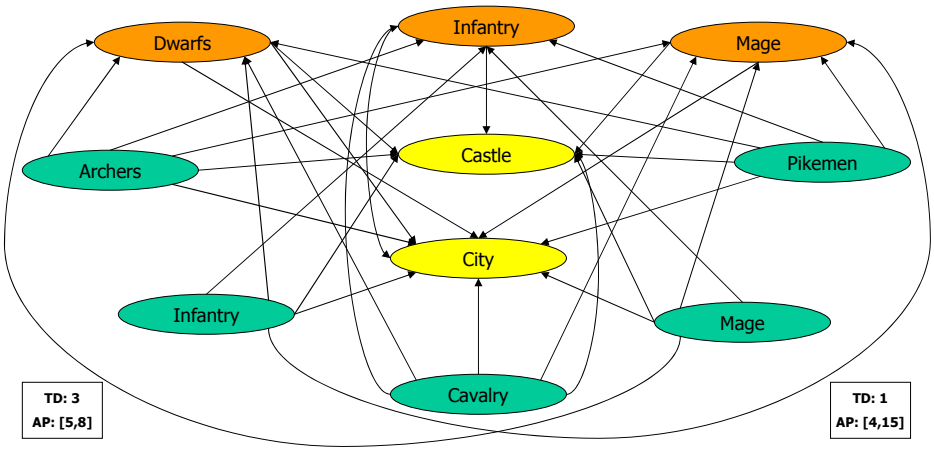
AI from programmer view

- What do I have (inputs)?:
  - game map
  - all entity information
  - pathfinding algorithms
  - rules evaluation algorithms
- What should be my result (outputs)?:
  - sequence of actions, that "looks cleverly"

Defence phase

- defence of AI towns and buildings is performed on "Defence graph"
  - graph structure with oriented edges
  - vertices are all entities relevant for defence on the map
  - in edges are stored relations and time disponibility between entities (in vertices)
  - well defined on every map and every game configuration
  - doesn't need any additional information

Defence graph - example



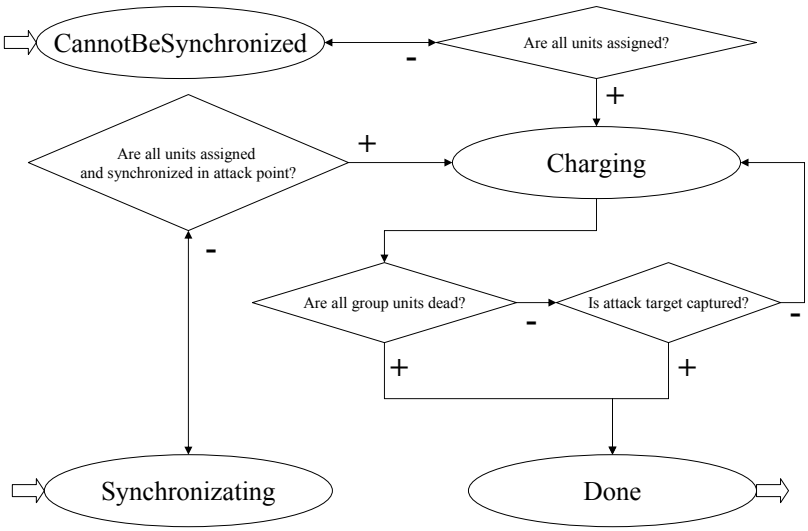
What to do with defence graph?

- AI finds out from defence graph:
  - "needs" – static entities (towns and buildings) guarded by AI
  - "sources" – units dedicated to defend
  - "threats" – possibilities of attack by enemy units
- 2 graph iterations (2 phases)
  - 1. phase: assigning of sources to needs by relative cost of need
  - 2. phase: assigning all remaining sources where are needed
- distinction between melee and firing threats
- criterions for "charge" vs. "wait and guard" method of defence
- additional computations
  - guarding of needs that are not in danger
  - attacking on non-attacking enemy units

Attack phase

- target selection:
  - static entity
    - nearest
    - most hated enemy
    - most valuable
    - least defended
  - unit
    - only in special cases
- attack group assembling
  - unit types selected by random
    - with some restrictions
- attack synchronization
- final attack processing and evaluating of attack result

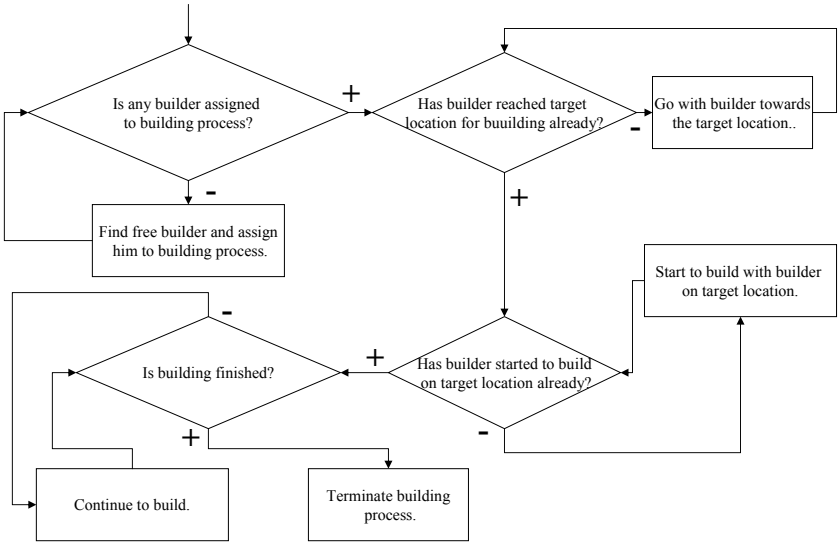
Attack life cycle



Buildings

- basic questions about building:
  - WHAT ?
  - WHERE ?
  - WHEN ?
- production buildings built on "the most safe" place
- bridge buildings
  - criterions if it is really needed to build a bridge
  - looking for the best bridge placement position
  - AI tries to have possibility to get everywhere it sees

Life cycle of building process



AI weaknesses

- Fog of war cheating
- attacking and defending works totally separately
  - unit jamming
- defend graph usage can be heavily optimized
- units have purpose from creation that is not changed during its whole lifetime
  - only exception is when enemy static entity is captured
- better money management
  - divided statically, not where are currently needed
- better unit healing and recruitment
- better unit selection
  - consider its strengths and weaknesses for its purpose

Diplomacy

- interesting game feature
- all players keep diplomatic relations between each other (there are four possible relations)
- relations can change
  - automatically as a direct result of player action
  - upon a players offers

AI and Diplomacy

- AI needs to learn to behave cleverly as a diplomatist
  - not to attack on allies
  - beware of/attack on enemies
  - quickly react on diplomatic events in game
- every AI player counts belief with regard to all other players individually
  - the belief is numeric value that is scaled on diplomatic relations
  - AI player offers always relation that corresponds to given belief
  - every diplomatic action influeces these beliefs
  - scale is not bounded below
  - AI can't bushwack !!!

Graphical User Interface

Planning

- work division, leading
- project decomposition
  - GUI
  - AI
  - rules
  - data formats & communication
  - common organisation issues
    - message system
    - logging

GUI

- many areas
  - 2D control elements
  - 3D interface – engine
  - data
    - sound
    - music
    - models
    - 2D graphics
- platform dependency
  - libraries selection

Sources & inspiration

- free libraries
  - basic IO
  - free 3D engine
- middleware
- open source games
- data
  - legal problems
  - volunteers
  - do by myself

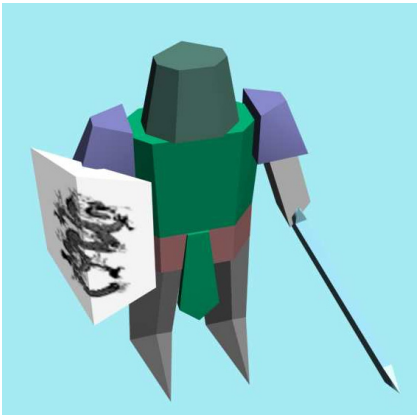
Design and specification

- we (don't) know what we want
  - bad specification
  - fuzzy rules
  - we hope it will go well
- we (don't) know how to do it
  - studying
  - we have to plan only what we understand
  - what do we have (data)

Graphics

- at the and nobody wants to do what everybody looked forward before
- what do we emulate - samples
- concept art
- demands on graphic quality are deformed by demands on whole game project
  - game is not typical software project

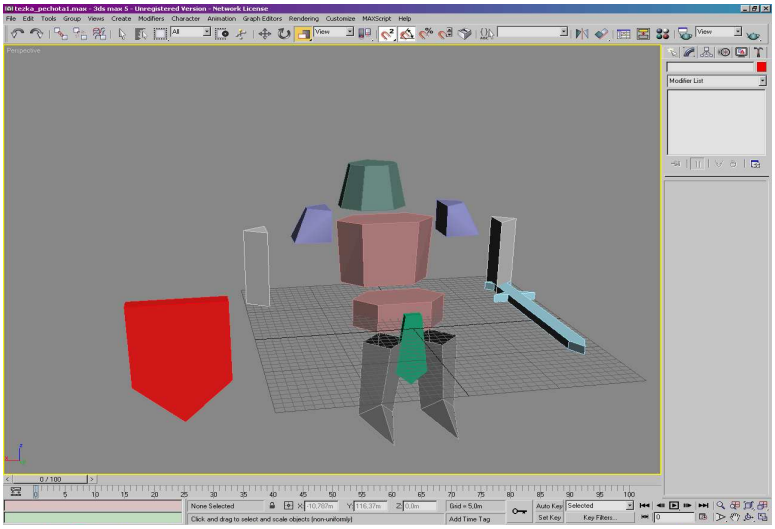
Concept art example



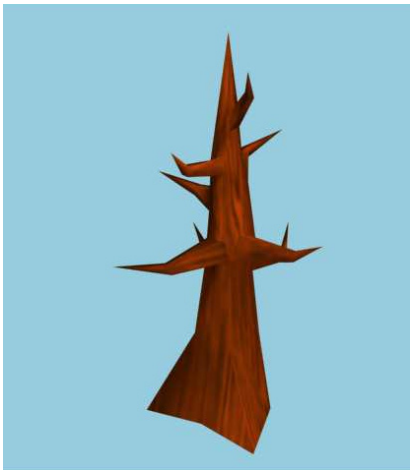
3D Studio and the others

- many people can do design web pages, but not many can do 3D models
- adopt and adjust
- create all completely by yourself
- animations
- texturing

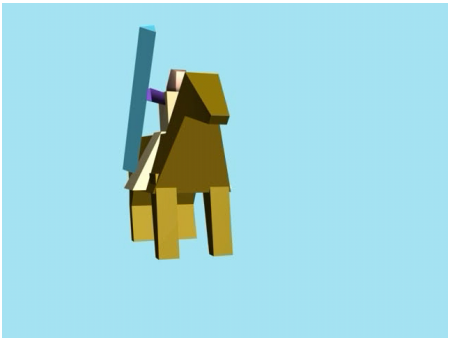
Model creation example



Texturing example



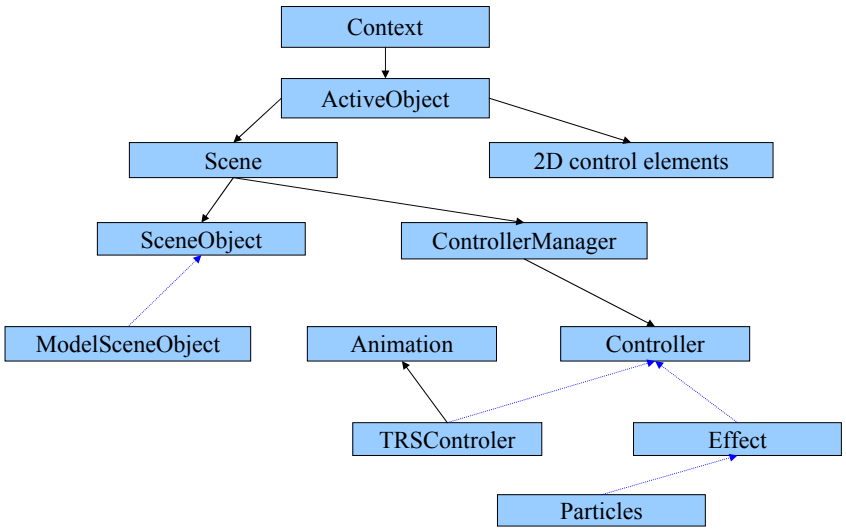
Animation example



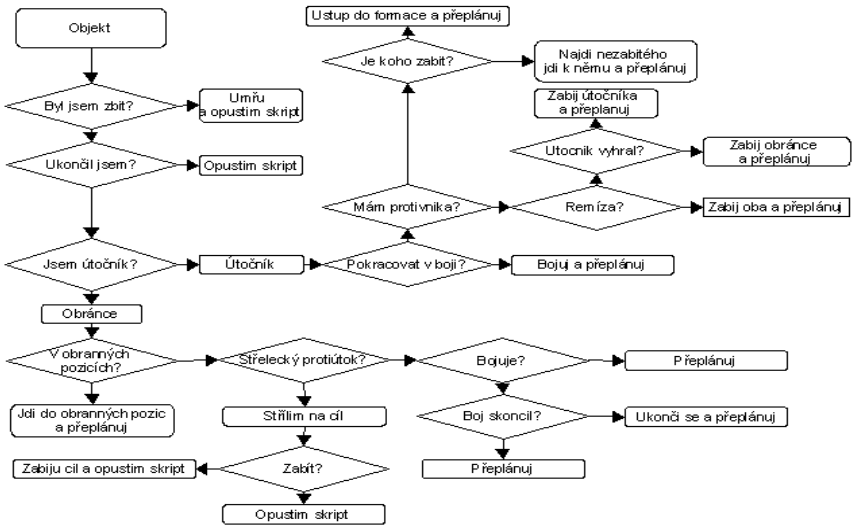
3D game engine

- map creation
  - editors
- data filling methods
- modularity
  - logic of objects on the scene
  - effects
  - realtime planning, world simulation
    - low level AI – scripts, steering

Modularity & Hierarchy



Scenario example



2D Interface

- apparently simple issue
- components
  - buttons, editboxes
- internationalization & localization
  - keyboard input, foreign fonts
  - there are methods how to design “international” GUI
  - limitations

Game itself

- drawing loop
- state machine
  - displaying data structures content
  - message sending
    - user inputs
  - message receiving and handling
    - outputs from game engine
- threads
  - immediate response needed

Low level work

- math operations
- curves
- fonts
- file loading

Resource manager

Resource Manager

- RM does
  - Load (make available)/ save resources
    - No others modules have to bother with I/O
  - Should be MultiThread save
- 8K implementation
  - RM is MT save, deal with IO, index resources, holds the data
  - Interface(s) are created by other modules
    - 'unlimited' number, calls RM functions only

Resource Manager

- Problems to be solved
  - Sharing (large) data with a module
  - 2 or more other modules want same data at one time
    - Only one data would be in the memory, wrapped in an object, which would store who access the data ATM
  - should be memory, which does not have to be saved, de-allocated by RM

Net

‘Battle net’

- Two approaches
  - Simplier
    - Web server collects informations about games only
    - The application download the IP’s
  - More difficult
    - The games run at the server
    - 1st player who connects become the operator
    - The game restarts when finishes

‘Simpler Battle Net’

- Search
  - GET /search.php HTTP/1.1\nHost: 8k.vta.cz\n\n
- Announce
  - Inform about the game (or refresh the information)
    - GET /main.php?players=%d&mname=%sHTTP/1.1\nHost: 8k.vta.cz
  - Stop announcing
    - GET /stop.php HTTP/1.1\nHost: 8k.vta.cz\n\n
    - IP is retrieved from the querry at the server
    - gethostbyname($_SERVER['REMOTE_ADDR']);
- Validation of IP
  - Server opens socket and connects to the game

‘More difficult Battle Net’

- Access to the server
  - (to be able to run the game itself)
- The games cannot end, but keep in loop
  - ‘Server’ parts has to be separated from ‘Client’ parts
- Can use the Simpler version to inform about the ports opened

Other features

- User validation (profiles, passwords)
- Bans of IP’s
- Non-playing players can observe the game / private games
- …

Links and references

- Official Homepage
  - http://kralovstvi.sourceforge.net/
- 1st public presentation at KUZ in May 06
  - http://www.cs.cas.cz/kuz/
- ChipOnline.de (85%)
  - http://www.chip.de/downloads/c1_downloads_20228710.html
- freenet.de (4/5)
  - http://download.freenet.de/archiv_8/8_kingdoms_8125.html

Questions?