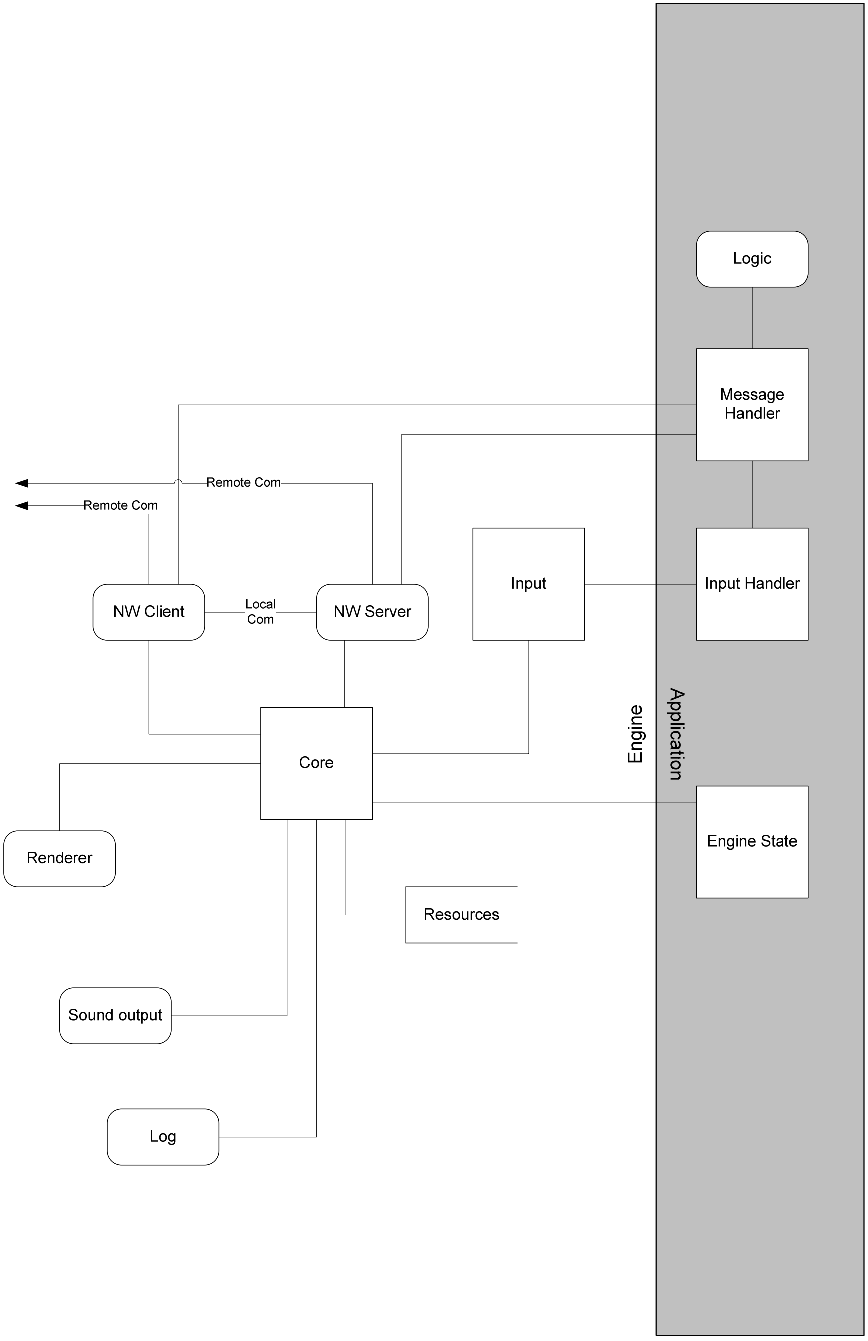
D.U.N.E.

Group 11
Klas Flodin
Kaj Sandberg
Erik Nikkola
Anders Ljungqvist
Mikael Nilsson

2.2 Overall Architecture Description



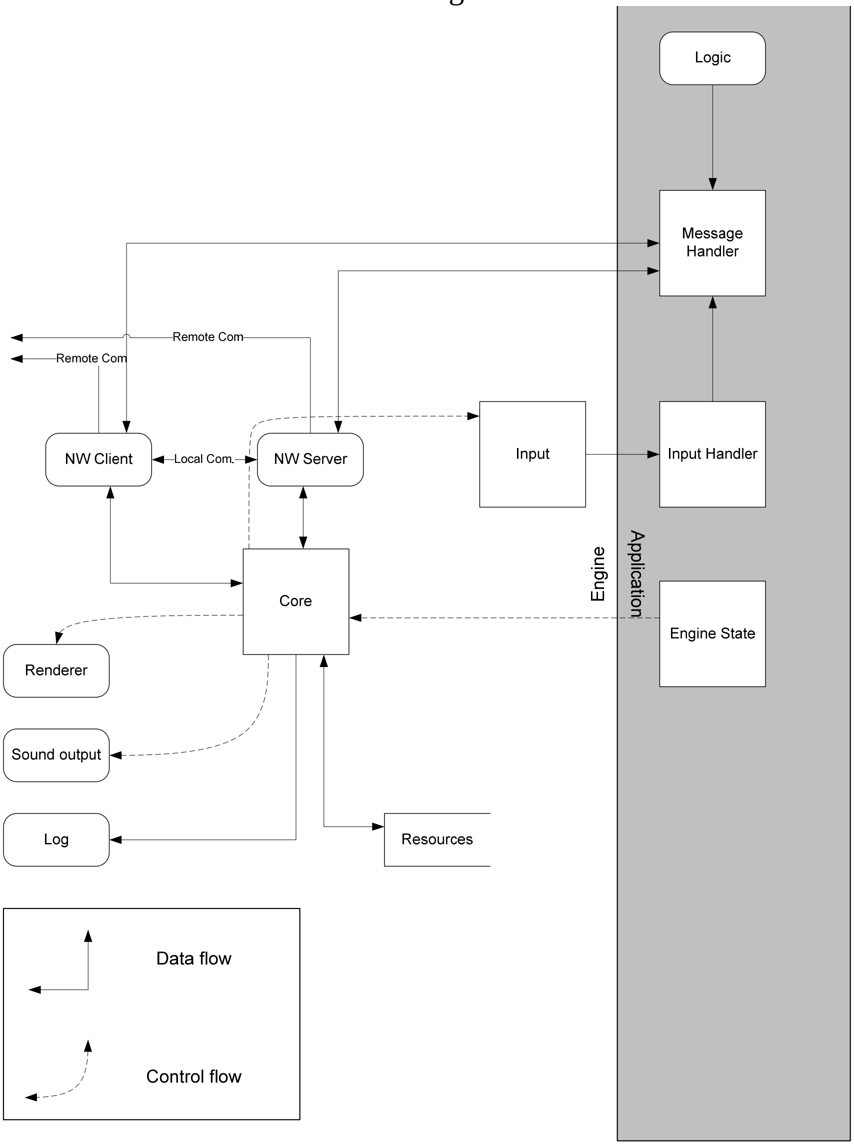
The game is built upon a Client / Server model where the engine acts as a server towards the application. It takes care of all tasks set by the application part of the game and is designed to be as general as possible so that updating the application will be made easier. It controls all output and input but the input is interpreted in the application. Within the engine exists several other Client/server relationships such as the network module.

The application part of the game handles all game specific tasks such as game logic, input handling and starting engine and so forth.

2.3 Detailed Architecture

The architecture will be demonstrated using a data flow and control flow diagram and Class Responsibility Collaborator (CRC cards). These were proposed by Ward Cunningham and Kent Beck[1] as a way of determining which classes are needed and how they will interact.

2.3.1 Data flow and control flow diagram



[1] *Kent Beck, Apple Computer, Inc.*
Ward Cunningham, Wyatt Software Services, Inc.

From the OOPSLA'89 Conference Proceedings
October 1-6, 1989, New Orleans, Louisiana
And the special issue of SIGPLAN Notices
Volume 24, Number 10, October 1989

2.3.2 CRC Cards

Engine

Core	
Responsibilities	**Collaborators**
Acts as a liason between the game application and the various parts of the game engine code.	Resources Log Sound output Renderer NW Client NW Server Input Engine State

Resources database	
Responsibilities	**Collaborators**
Store data, settings, textures etc.	Core

Log	
Responsibilities	**Collaborators**
Record important events and extecutions Record errors.	Core

Sound output	
Responsibilities	**Collaborators**
Manage sound output	Core

Renderer	
Responsibilities	**Collaborators**
Handle graphical computation and output	Core

NW Client	
Responsibilities	**Collaborators**
Network communication towards server	NW Server Message Handler

NW Server	
Responsibilities	**Collaborators**
Handle game syncronization between clients	NW Client Core Message Handler

Input	
Responsibilities	**Collaborators**
Detect I/O actions and send them to Input handler	Input Handler

Application

Logic	
Responsibilities	**Collaborators**
Verify and handle all in-game actions	Message Handler

Message Handler	
Responsibilities	**Collaborators**
Handle messaging between Application and Engine.	Logic Input Handler NW Server NW Client

Input Handler	
Responsibilities	**Collaborators**
Interpret inputs and relay to message handler	Input Message Handler

Engine State	
Responsibilities	**Collaborators**
Control Core state. (Menu, in-game, pause etc.)	Core