Software Design Techniques

Nagalaxmi Telkar

CSCI 5828

Presentation Slides

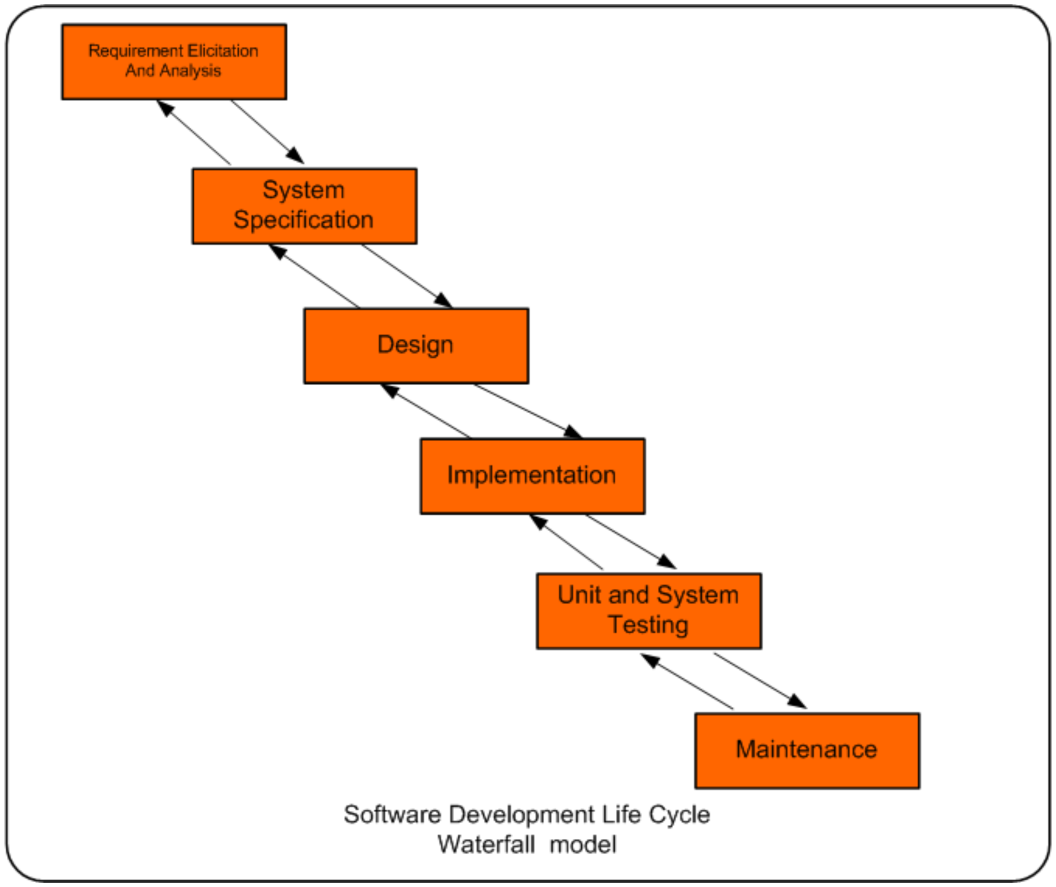
Contents

INTRODUCTION

- Computers do not make mistakes, but computer software is written by human beings, who certainly do make mistakes.
- As complex computer systems influence every facet of our lives - the cars we drive, the airplane and trains we rely on others to drive for us, and even everyday machinery such as domestic washing machines, the need for reliable and dependable software systems has become apparent.
- Developing a **complex computer system** follows a development process, or a **life cycle** similar to building a house.

Software Design Life Cycle

SDLC - Waterfall Model



Software Development Life Cycle
Waterfall model

SDLC - Waterfall Model

- **Requirement Elicitation and Analysis** involves the determination of the exact requirements of the system.
- **System Specification** is used in deriving what the system should do, without saying how this is to be achieved.
- **Design** phase is intended towards addressing how the system is to be implemented.
- **Implementation** phase is traditionally described as programming.
- **Unit and System testing** aims to trap bugs.
- **Maintenance** keeps the system updated for new changes that need to be implemented.

Software Design Process

WHY SHOULD THE SOFTWARE BE DESIGNED AT ALL?

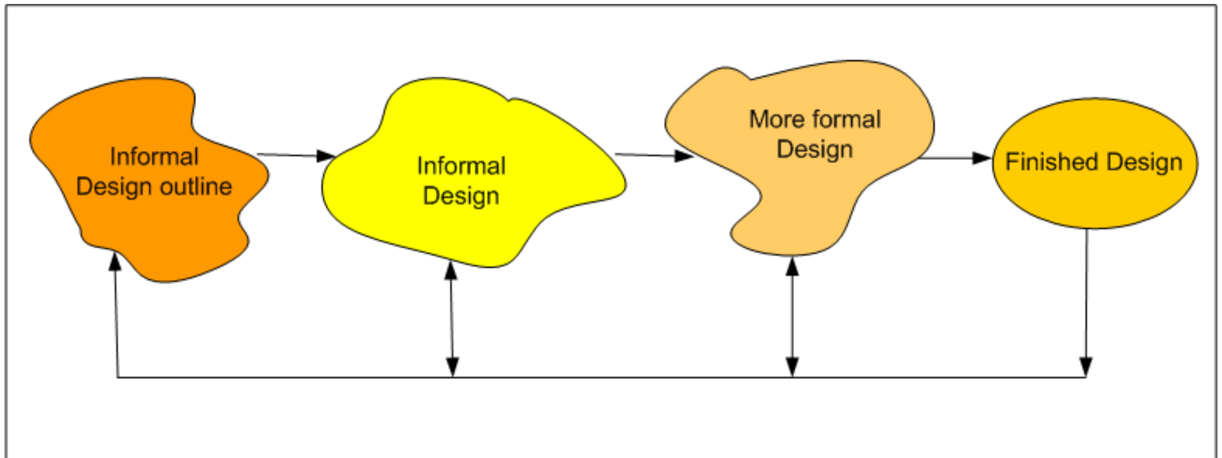
- We can't just throw few dozens of programmers to start programming without any detailed plans.
- Design is highly creative stage in software development where the designer plans
  - how the system or program should meet the customer's needs
  - how to make system effective and efficient.

Tackling Design Problems

- Any design problems must be tackled in three stages;
  - *Study and understand the problem*
  - *Identify gross features of at least one possible solution*
  - *Describe each abstraction used in the solution*

Design Process

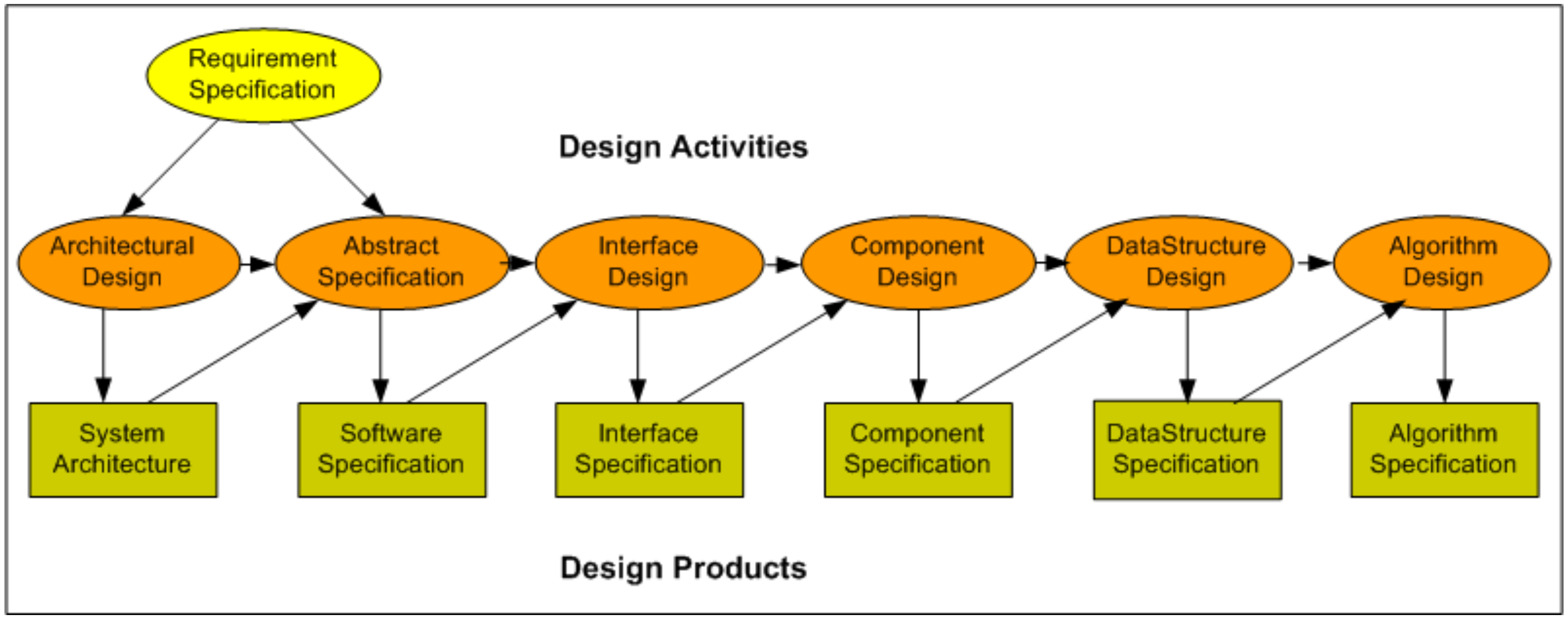
- Software designers do not arrive at a finished design immediately. They develop design iteratively through number of different versions. The starting point is informal design which is refined by adding information to make it consistent and complete as shown in the figure below:



Design Process

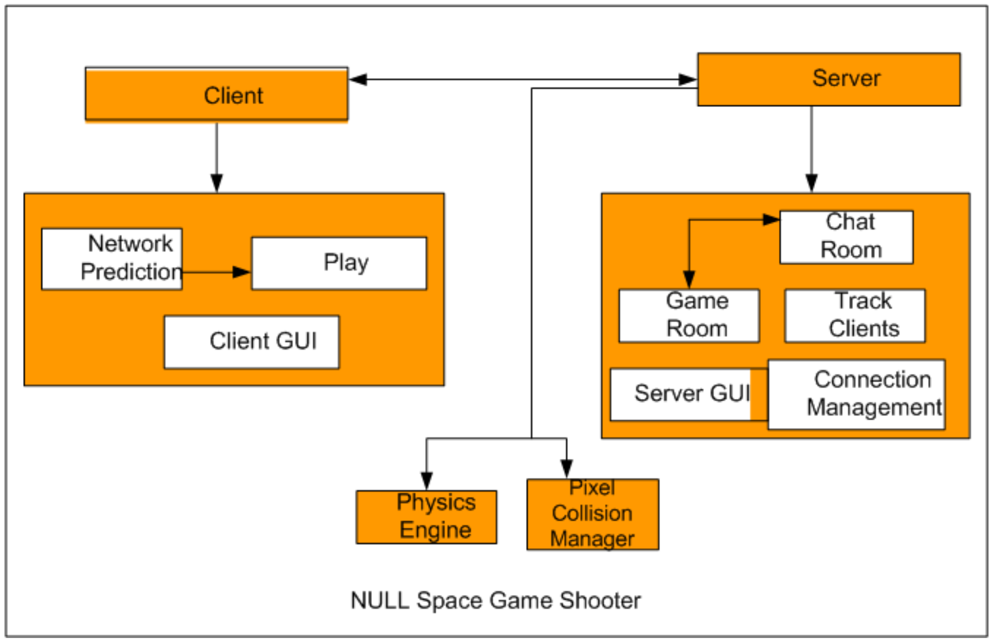
Design Process

- A general model of design process



Architectural Design-1

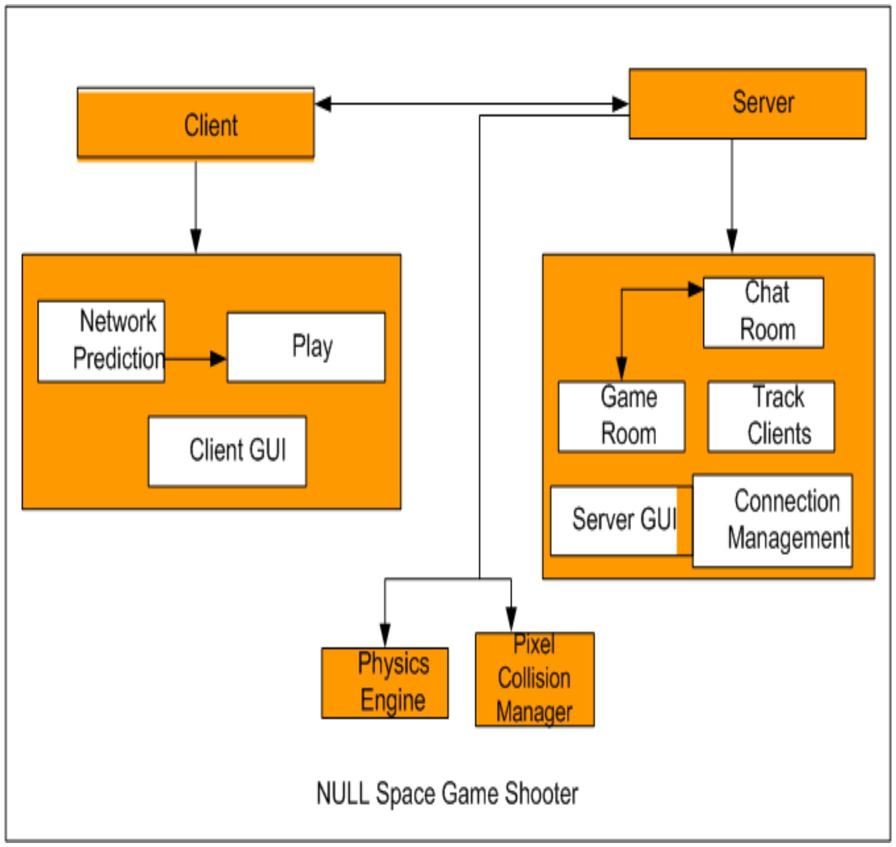
- Identifying the sub-systems and establishing framework for sub-system control and communication.
- Explained using the following architecture



ARCHITECTURE DESIGN – 2

- Activities necessary for architectural designing;
  - *System Structuring*
  - *Control modeling*
  - *Modular decomposition*
- The output of the architectural design process is an architectural design document.

Architecture Design – 3
NULL Space Game Shooter

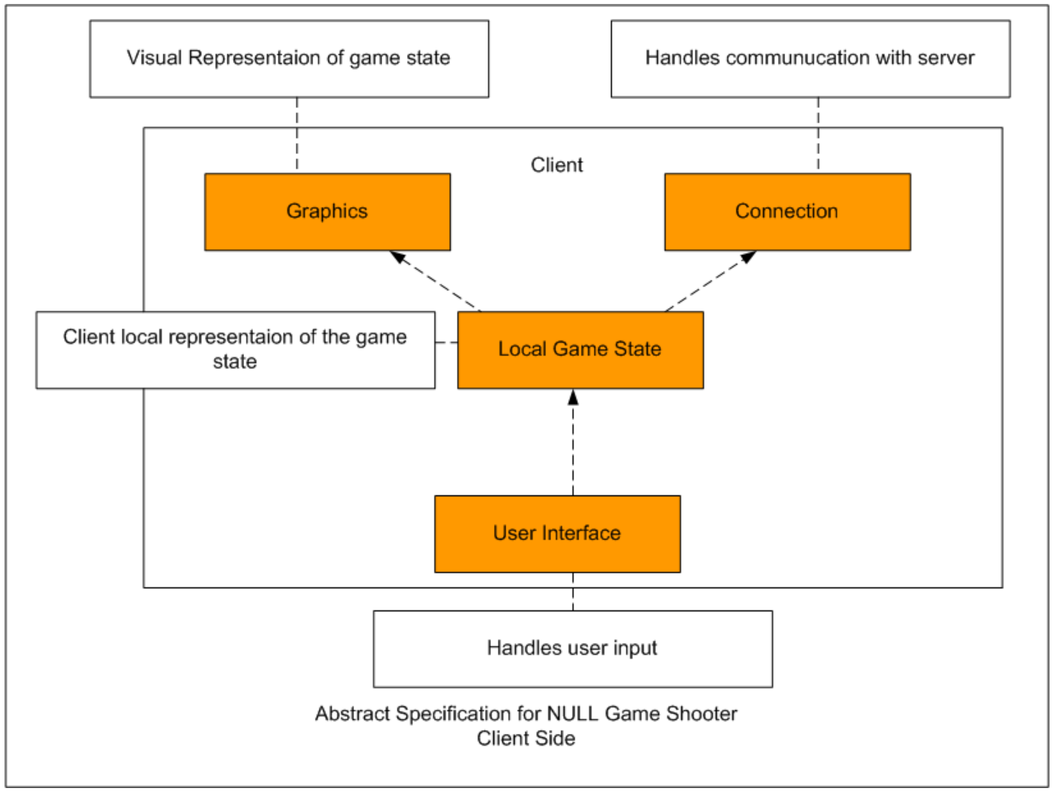


NULL Space Game Shooter

- Client and Server are the subsystems that are controlling the other modules.
- Other modules are the rectangles in white. Ex, Network Prediction, Play, Chat Room, Game Room etc.

Abstract Specification - 1

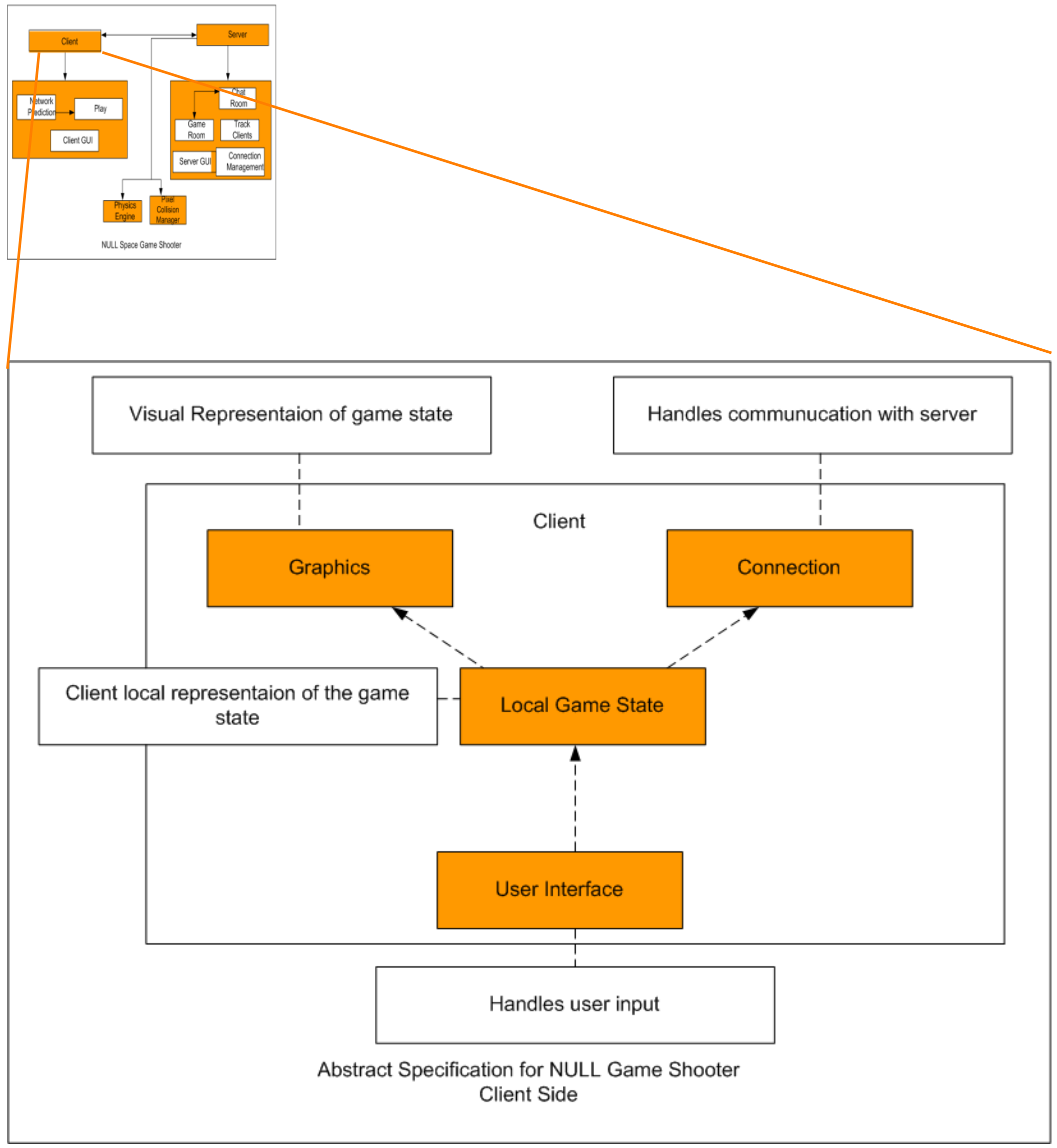
- For each sub-system, an abstract specification of the services it provides and the constraints under which it must operate is produced.


Visual Representaion of game state
Handles communucation with server
Client
Graphics
Connection
Client local representaion of the game state
Local Game State
User Interface
Handles user input

Abstract Specification for NULL Game Shooter
Client Side

Abstract Specification – 2

Client Side of NULL Game Shooter

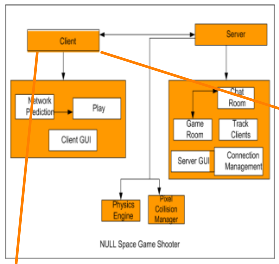


Abstract Specification for NULL Game Shooter
Client Side

- Orange Boxes are the subsystems.
- White Boxes are the services provided by the subsystems.
- Services include
  - Visual Representation
  - Communication Layer
  - User I/O
  - Client Side Copy

Interface Design - 1

- For each sub-system, its interface is designed and documented.



The text box below shows a few functions in the interface for the client. These are used by the player object. There will be separate interfaces for each module.

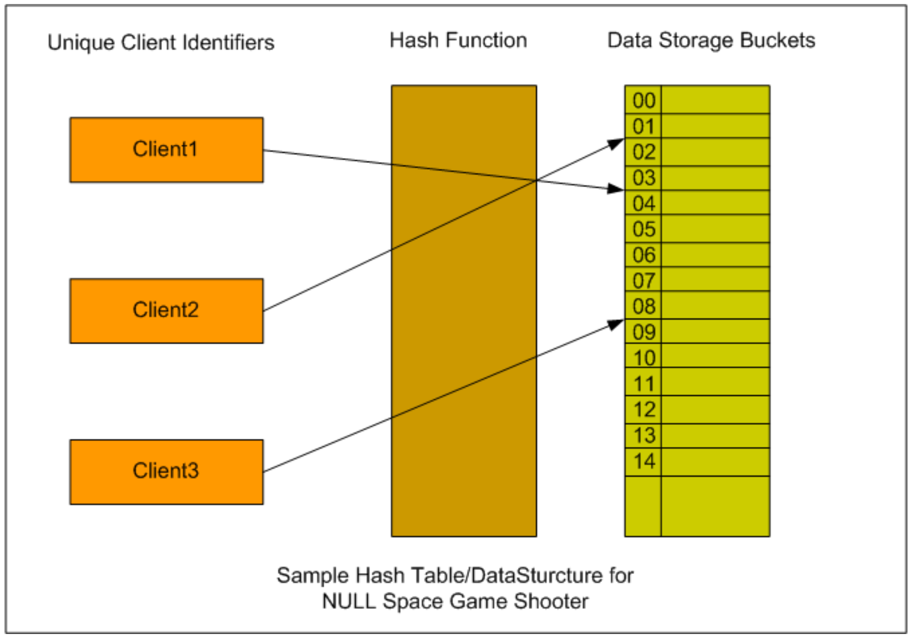
1) Move (…)
2) GetPosition(…)
3) GetShip(…)
4) CalculateDamageCollision(…)
5) RotateShip(…)
6) GetVelocity(…)

Component Design - 1

- Services are allocated to different components and the interfaces of the components are designed.
- This phase entails detailed implementation design of the interfaces that are identified in the interface design.
- Services that are allocated to each subsystem are designed as to be implemented.

DATA STRUCTURE DESIGN - 1

- The data structures used in the system implementation are designed in detail and specified.
- Example


Unique Client Identifiers
Hash Function
Data Storage Buckets
Client1
Client2
Client3
00
01
02
03
04
05
06
07
08
09
10
11
12
13
14
Sample Hash Table/DataSturcture for NULL Space Game Shooter

Client information is stored in the buckets by mapping them to the buckets via a hash function

ALGORITHM DESIGN - 1

- The algorithms used to provide services are designed in detail and specified.

Software Design Methodologies

DESIGN METHODOLOGIES - 1

- J. Christopher Jones, taken from classis work, *Design Methods: Seeds of Human Futures(Jones 1970.)*
- *'The fundamental problem is that designers are obliged to use current information to predict a future state that will not come about unless their predictions are correct. The final outcome of designing has to be assumed before the means of achieving it can be explored: the designers have to work backwards in time from an assumed effect upon the world to the beginning of a chain of events that will bring the effect about.'*

Design Methodologies - 2

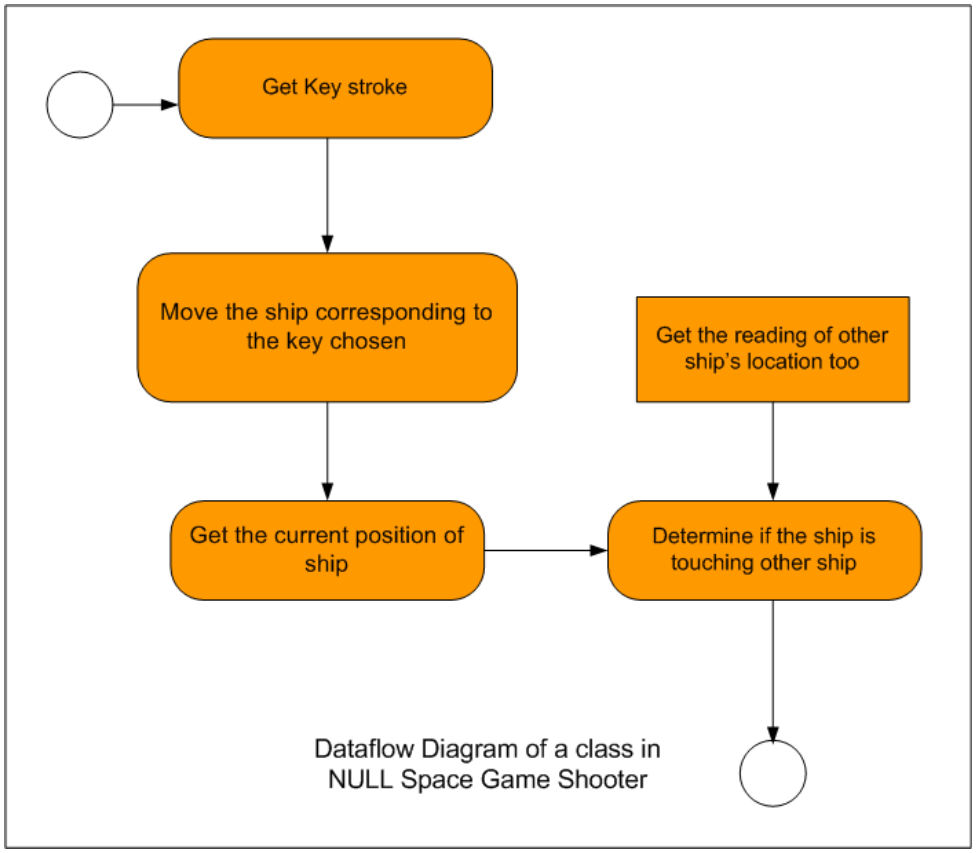
- A more methodical approach to software design is proposed by ***structured methods*** which are sets of notations and guidelines for software design.
- Two major rules of this method
  - Programs were to be broken into functions and subroutines
  - There was only a single entry point and a single exit point for any function or routine.
- ***Structured methods*** often support some or all of the following models of a system:
  - A data-flow model
  - An Entity-relationship model
  - A structural model
  - An object-oriented model

Design Methodologies - 3

- Rounded rectangles represents functions which transform inputs to outputs.
  - The transformation name indicates its functions.
- Rectangles represents data stores.
- Circles represents user interactions with the system which provide input and receive output.
- Arrows show the direction of the data flow.

DATA FLOW MODEL

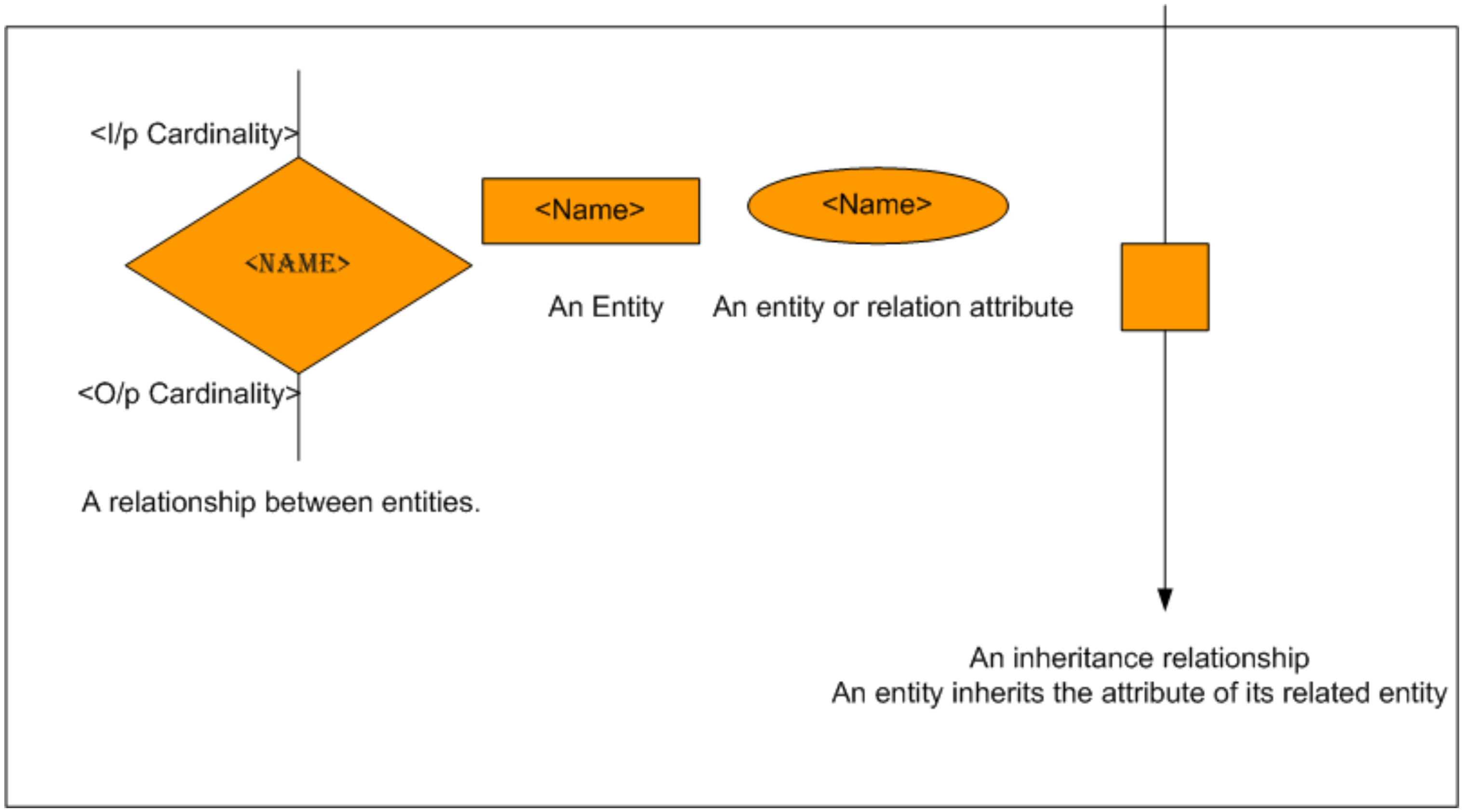
- Example for data flow model.


Get Key stroke
Move the ship corresponding to the key chosen
Get the reading of other ship's location too
Get the current position of ship
Determine if the ship is touching other ship

Dataflow Diagram of a class in NULL Space Game Shooter

Object gets the keystroke and moves the ship in the game accordingly. Based on the positions of other ships, collision will be decided.

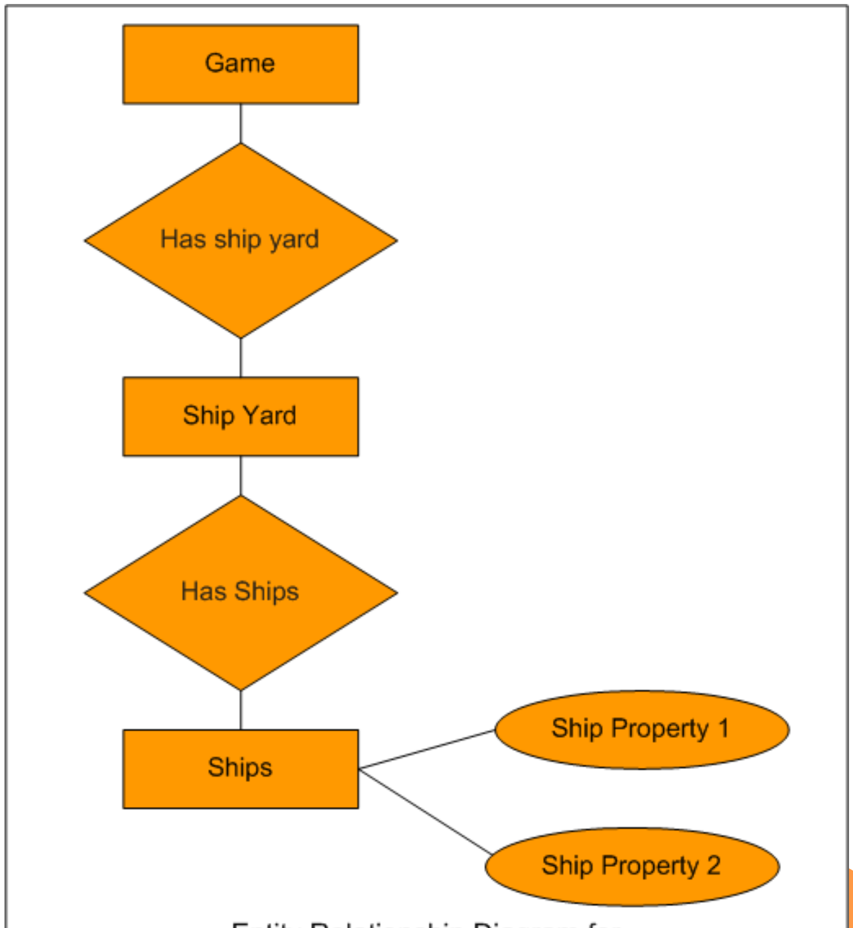
ENTITY RELATIONSHIP NOTATIONS- 1


<I/p Cardinality>
<NAME>
<O/p Cardinality>
A relationship between entities.
<Name>
An Entity
<Name>
An entity or relation attribute
An inheritance relationship
An entity inherits the attribute of its related entity

Entity Relationship Model - 1

- Example for

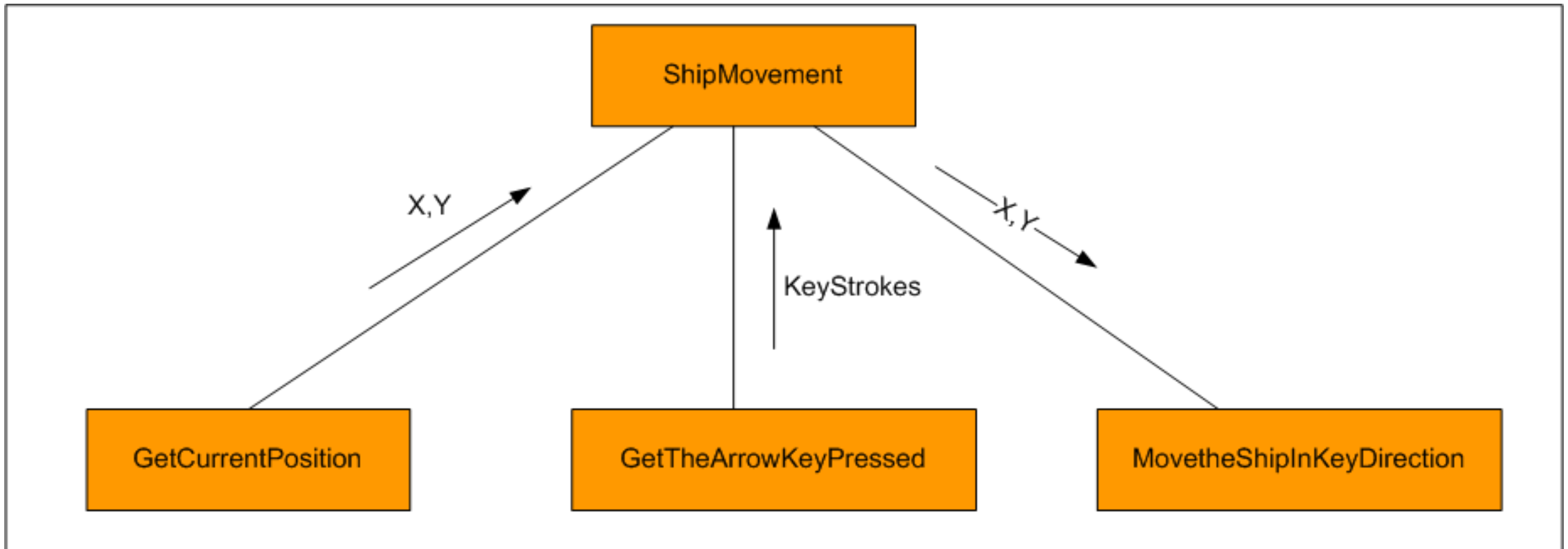
Entity Relationship Model



Entity Relationship Diagram for
NULL Space Game Shooter

STRUCTURAL MODEL - 1

- Example for structural model



Structural Model of
NULL Space Game Shooter

Software Design Strategies

Design Strategies-1

- Function Oriented
  - Design is decomposed into set of interacting units where each unit has clearly defined function
  - Conceals the details of an algorithm in a function but system state information is not hidden.
- The activities of this strategy;
  - Data-flow design
  - Structural decomposition
  - Detailed design description

Design Strategies-2

- Object-oriented design
  - Is based on the idea of information hiding.
  - System is viewed as a set of interacting objects, with their own private state.
  - Dominant design strategy for new software systems.
  - Objects communicate by calling on services offered by other objects rather than sharing variables. This reduces the overall system coupling.
  - Message passing model allows objects to be implemented as concurrent processes.
- Kinds of concurrent object implementation
  - Passive objects
  - Active objects

Science Of Design

- **Don Batory** argues that a fundamental problem in software engineering is the abject lack of a science for software design.
- In October 2003, he attended a *National Science Foundation (NSF)* workshop in Virginia on the “Science of design”.
- **Fred Brookes,** “We don’t know what we’re doing, and we don’t know what we’ve done”
- Software design process is an *art* or an *inexact science.*
- If it is purely a mechanical process by which a specification is translated into a design of an efficient program, then this process follows an *exact* or *deterministic science*.

Design Quality-1

- Budgen(1993) describes some of the ‘ilities’ that form a group of quality factors that need to be considered when making any attempt to assess design quality.
  - Reliability
  - Efficiency
  - Maintainability
  - Usability.

Design Quality-2

- For the systems to be reliable, the designers should count on completeness, consistency and robustness.
- Efficiency of a system can be measured through its use of resources such as processor time, memory, network access and so on.
- By separating the concerns, designers can help the future maintainers to gain clear understanding of their original ‘mental models’.
- Design of user interface forms an important component and will influence other design decisions.

Design Quality-3

- 10 heuristics of user interface design;
  - Visibility of system status
  - Match between system and the real world
  - User control and freedom
  - Consistency and standards
  - Error prevention
  - Recognition rather than recall
  - Flexibility and efficiency of use
  - Aesthetic and minimalist design
  - Help users recognize, diagnose, and recover from errors
  - Help and documentation

Software Tools

- The productivity of engineering designers is improved when they are supported by CAD systems which take over tedious drawing chores and which check for errors and omissions.
- The term CASE (Computer Aided Software Engineering) is generally accepted as the name for this automated support for engineering process.

Software Design Failure.

- Software design flaws leading to mishap.
- The Ariane 5 was not flight tested because there was so much confidence on part of the management team. The error which ultimately led to the destruction of the Ariane 5 launcher was clearly identified in the report of the investigating committee: a program segment for converting a floating point number, representing a measurement, to a signed 16 bit integer was executed with an input data value outside the range representable by a signed 16 bit integer.
- This is not a software issue, but a design flaw at a much deeper level.

References

- Software engineering techniques: design for quality By Krzysztof M. Sacha
- Software Design (2nd Edition) by D. Budgen
- High-Integrity System Specification and Design (Formal Approaches to Computing and Information Technology (FACIT)) by Jonathan P. Bowen and Michael G. Hinchey
- Software Engineering (7th Edition) (Hardcover) Ian Sommerville
- A Science of Software Design. 3-18 Don Batory
- J. Christopher Jones, *Design Methods: Seeds of Human Futures(Jones 1970.)*