**TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Thiết kế hệ thống chia tải cho mô hình game local network trong Unity**

**LÊ GIA ĐỨC**

duc0213266@huce.edu.vn

**Chuyên ngành Khoa học máy tính**

**Khoa Công nghệ thông tin**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Doãn Thị Ngân  Chữ ký của GVHD |
| **Bộ môn:** | Khoa học máy tính |
| **Trường:** | Đại học Xây dựng Hà Nội |
| **HÀ NỘI, 11/2024** | |

**Lời cảm ơn**

Lời cảm ơn đầu tiên mình xin được gửi tới cô Ngân, cũng là người hướng dẫn mình hoàn thiện đề tài này. Với sự góp ý của cô, mình đã hoàn thiện bài viết này và đề tài ở mức độ cao nhất có thể. Đây cũng sẽ là mô hình được tiếp tục hoàn thiện và tích hợp vào trong các trò chơi cá nhân hoặc các sản phẩm tương tự khác của bản thân trong tương lai. Rất cô trong thời gian qua đã đồng hành cùng mình!

Trong quá trình viết cuốn, có thể mình sẽ có những sai sót, mong mọi người thông cảm và góp ý để mình có thể sửa đổi hoàn thiện hơn. Rất trân thành cảm ơn!

**Tóm tắt nội dung đề tài**

Mình sẽ giới thiệu qua chút về trò Soul Kinght:

Đây là trò chơi thuộc thể loại Roguelike được phát hành bởi ChillyRoom vào năm 2017. Trò chơi có tính dối kháng và phối hợp đồng đội cao.

Với chế độ multiplayer, nhiều người chơi có thể kết nối với nhau thông qua một thiết bị mạng wifi nội bộ. Một thiết bị sẽ đóng vai trò là máy Host điều phối việc spawn các tham số trong suốt quá trình diễn ra hoạt động trò chơi. Câu chuyện xuất phát từ đây khi một thiết bị được chọn làm Host khá yếu và không đủ nhanh có thể dẫn tới trường hợp tất cả các người chơi kết nối với nhau được điều phối qua thiết bị trung gian này bị văng ra khỏi trò chơi.

Hiện nay với sự phát triển của công nghệ, chúng ta đã có những thiết bị thực sự vượt trội so với ngày trò chơi này được ra mắt và điều trò chơi này bị văng khi máy làm Host bị quá tải đã không còn xảy ra nữa. Nhưng đây vẫn là một bài toán thú vị để chúng ta có thể tạo ra giải pháp nâng cao khả năng tính toán cho tất cả các thiết bị trong cùng một mạng nội bộ như thế. Đề tài này là các mình thực hiện giải pháp này. Bằng cách chia việc cho các máy trên cùng mạng thay vì chỉ một máy phải làm toàn bộ quy trình này, ta có thể cân bằng được tài nguyên cho các máy, nhưng bên cạnh đó là thử thách về đồng bộ trong việc phân chia cho chúng như thế nào để vẫn đồng bộ được dữ liệu truyền tải giữa các máy với nhau.

**MỤC LỤC**

[CHƯƠNG 1. CÁC QUI ĐỊNH CHUNG 1](#_Toc21499666)

[1.1 Giới thiệu chung 1](#_Toc21499667)

[1.2 Sử dụng các định dạng văn bản theo qui định 1](#_Toc21499668)

[1.2.1 Qui định về căn lề văn bản 1](#_Toc21499669)

[1.2.1 Tạo lề cho văn bản in 2 mặt 3](#_Toc21499670)

[1.2.2 Tạo chương mới 3](#_Toc21499671)

[1.2.3 Tạo tiêu đề các cấp 3](#_Toc21499672)

[1.2.4 Định dạng phần nội dung các chương, mục 4](#_Toc21499673)

[1.2.5 Hình vẽ - Đồ thị 4](#_Toc21499674)

[1.2.6 Bảng biểu 6](#_Toc21499675)

[1.2.7 Phương trình 8](#_Toc21499676)

[1.3 Tạo tham chiếu chéo giữa các đoạn văn bản 11](#_Toc21499677)

[1.4 Tạo danh mục tài liệu tham khảo 11](#_Toc21499678)

[1.5 Cập nhật lại các chú thích và tham chiếu 15](#_Toc21499679)

[1.6 Tạo danh mục hình vẽ 15](#_Toc21499680)

[1.7 Tạo danh mục bảng biểu 16](#_Toc21499681)

[1.8 Tạo trang mục lục 16](#_Toc21499682)

[1.9 Qui cách đóng quyển 17](#_Toc21499683)

[CHƯƠNG 2. SỬ DỤNG CÁC BIỂU ĐỒ 19](#_Toc21499684)

[2.1 Giới thiệu về biểu diễn bằng đồ thị 19](#_Toc21499685)

[2.2 Đồ thị kiểu bánh 19](#_Toc21499686)

[2.3 Đồ thị kiểu thanh ngang 20](#_Toc21499687)

[2.4 Đồ thị kiểu cột đứng 20](#_Toc21499688)

[2.5 Đồ thị kiểu đường 21](#_Toc21499689)

[2.6 Đồ thị kiểu diện tích 21](#_Toc21499690)

[CHƯƠNG 3. KẾT LUẬN 23](#_Toc21499691)

[3.1 Kết luận 23](#_Toc21499692)

[3.2 Hướng phát triển của đồ án trong tương lai 23](#_Toc21499693)

[TÀI LIỆU THAM KHẢO 24](#_Toc21499694)

[PHỤ LỤC 25](#_Toc21499695)

**DANH MỤC HÌNH VẼ**

[Hình 2.1 Đồ thị kiểu bánh 20](#_Toc20580104)

[Hình 2.2 Đồ thị kiểu thanh ngang 20](#_Toc20580105)

[Hình 2.3 Đồ thị kiểu cột đứng 21](#_Toc20580106)

[Hình 2.4 Đồ thị kiểu đường 21](#_Toc20580107)

[Hình 2.5 Đồ thị kiểu diện tích 22](#_Toc20580108)

**DANH MỤC HÌNH VẼ**

[Bảng 1.1 Thống kê các thiết bị và giá thành 8](#_Toc20580109)

# TỔNG QUAN BÀI TOÁN

## Giới thiệu chung

Trò chơi Soul Knight là một trò chơi thành công, nhưng bên cạnh đó vẫn gặp một số lỗi khi có nhiều người cùng tham gia vào cùng một màn chơi. Cụ thể là việc bị văng game khi thiết bị được chọn làm Host yếu, làm cho việc tính toán, khởi tạo các thông số trò chơi trong quá trình hoạt động không đáp ứng được. Việc thiết kế mô hình truyền tải trong trò chơi này tương tự Clients – Server, đây là mô hình đơn giản và cũng dễ thiết kế nhất, nhưng nhược điểm cố hữu là khi thiết bị được chọn làm Server(Host) bị gặp trục trặc thì các Client (thiết bị của những người chơi khác) cũng không thể kết nối được với nhau.

Bài toán được lấy từ một vấn đề thực tiễn trong quá trình thiết kế giải pháp cho trò chơi điện tử và mang tính thực tiễn cao. Nó có thể áp dụng cho các bài toán khác tương tự trong cùng mạng nội bộ hoặc xa hơn có thể áp dụng cho các máy có chung vlan (được cấu hình sao cho mỗi máy đều kết nối với tất cả các máy khác trong mạng).

## Giải pháp (Mô hình Producers - Consumers)

Tại sao lại chọn mô hình (Producers – Consumers) ?

Mô hình này sẽ làm tăng tính liên lạc giữa các máy và không bị phụ thuộc vào một thiết bị trung chuyển mà tiêu biểu là mô hình Master - Slave ta vẫn hay thấy trong các hệ thống máy chủ phân tán hoặc mô hình đang áp dụng trong trò chơi hiện tại là Clients – Server.

Với việc áp dụng định lý CAP (Consistency – Availability - Partition Tolerance), ta có thể thấy trò chơi Soul Knight đang sử dụng mô hình tương tự Slave Master này sẽ gặp vấn đề về Availability (Tính sẵn sàng). Do đó, khi máy Host không thể đáp ứng được nhu cầu xử lý thì thiết bị của Player trong trò chơi ngay lập tức bị văng. Với việc ứng dụng một mô hình khác, cụ thể ở đây là Distributed Computing (Thiết bị của các Player vừa là Producer, vừa là Consumer). Với thiết kế này, ta sẽ tập trung vào Tính sẵn sàng (Availability) và Tính nhất quán (Consistency). Nó cũng sẽ giải quyết luôn vấn đề khi Player trên thiết bị Host thoát trò chơi thì các Player khác vẫn có thể chơi bình thường.

Tất nhiên với cách thiết kế này, thì việc tiêu tốn tài nguyên trên các máy cũng được cân bằng hơn. Các máy yếu hơn sẽ “dễ thở hơn” và sẽ không gặp phải tình trạng quá tải nữa.

## Thiết kế giải pháp

Do được thiết kế tối ưu hóa về Tính sẵn sàng (Availability) và Tính nhất quán (Consistency) nên cần một sự đồng bộ về dữ liệu cần xử lý giữa các máy với nhau.

Để giải quyết vấn đề này, ta sẽ cung cấp mỗi máy một hàng đợi tăng tính đồng bộ giữa các máy với nhau. Hàng đợi này tạm gọi là GlobalQueue, mỗi máy đều giữ một hàng đợi này, bên cạnh đó là một LocalQueue để đưa các nhiệm vụ cần xử lý vào. Mỗi nhiệm vụ xử lý được gọi là các Job (được chia ra với cấu trúc dữ liệu giống nhau cho mỗi nhiệm vụ). Các Job này có thể xử lý theo đặc trưng từng bài toán. Việc xử lý được diễn ra trong các Handler được cài vào trong LocalQueue, việc này làm tăng tính tùy biến và giảm phức tạp trong quá trình cấu hình cho Hanlder.

Tác dụng và hướng xử lý:

* Điều chỉnh các Job cần xử lý giữa các máy với nhau, các Job sau khi hoàn thành hoặc không xử lý được (có thể do trục trặc thiết bị hoặc lập tức bị ngắt kết nối) đều sẽ được nhả vào GlobalQueue để bốc đi và truyền vào các máy khác thông qua liên kết TCP
* Nhận các Job từ liên kết TCP về và được thêm vào GlobalQueue. Các Job khi đã ở trạng thái hoàn thành sẽ không được đẩy vào LocalQueue xử lý nữa.
* Với các Job được đưa vào LocalQueue từ GlobalQueue cho phép việc xác nhận xử lý diễn ra dễ dàng hơn và kiểm soát số lượng Job có thể nhận xử lý dễ dàng hơn. Từ đó, với mỗi Job được hoàn tất xử lý hoặc lỗi xử lý diễn ra ta có thể dễ dàng
* Việc xử dụng một hàng đợi được thiết kế bởi List (hoặc LinkedList) và các hàm tương tác tương tự như một Queue cho phép ghi đè bất cứ Job nào đã được hoàn tất với cùng Id tại trạng thái sau xử lý và tăng việc tuần tự hóa các Job đã được nhận xử lý tại LocalQueue.
* Khi Job được đưa vào LocalQueue, chúng sẽ được xử lý bởi các Hanlder cấu hình bên trong LocalQueue này. Các Job sau khi được LocalQueue nhả ra sẽ có các thông số lỗi hoặc các thông số hoàn thành xử lý tùy theo việc nó có lỗi hay không và được tiếp tục đưa vào GlobalQueue để được đồng bộ lại và chờ bốc gửi đi.
* Các Job được truyền qua máy khác (Dequeue tại GlobalQueue) nên sẽ mất khỏi máy truyền đi. (Các Job chỉ được gán Id tại thiết bị khởi tạo nên việc đồng bộ hóa chúng sẽ diễn ra đơn giản). Các Job đã được hoàn tất sẽ được ưu tiên ghi đè lên các Job chưa được xử lý hoặc các Job bị lỗi trong quá trình xử lý.

## Kiểu định dạng dữ liệu

Kiểu dữ liệu được thiết kế thông qua Json với các thành phần đóng gói là các Job. Trong các Job sẽ có trường lưu trữ các giá trị của dữ liệu cần xử lý.

Do đó người dùng có thể tùy biến các kiểu dữ liệu cần xử lý và đóng gói chúng lại. Thời gian xử lý vả phàn hồi sẽ tùy thuộc vào việc các Handler được cấu hình theo. Cấu hình dữ liệu trong các Job cần đồng nhất để có thể xử lý trong các Handler.

Các loại Queue chỉ đóng vai trò là các công cụ gửi và nhận các Job.

## Hướng giải quyết trong tương lai

Do giải pháp được xây dựng trong thời gian ngắn nên không thể không có các thiếu sót. Mình rất mong nhận được sự góp ý của mọi người để hoàn thiện hơn mô hình xử lý này, phần vì để nó hoàn thiện hơn, phần cũng để việc triển khai lên thực tế thuận lợi hơn.

Trong tương lai, có thể mình sẽ đưa thêm một vài trạng thái nữa vào các Job để có thể mở rộng tính đánh giá và thiết kế lại các hàng đợi để tăng tốc độ truy xuất. Việc mở rộng xử lý trên các máy là công việc khó kiểm soát, có thể mình sẽ thêm vào đó một hệ cơ sở dữ liệu để ghi lại Log.

Mọi đánh giá hoặc góp ý chân thành đều là thứ quý giá và sẽ được tác giả lắng nghe và sẽ cố gắng hoàn thiện nhất có thể. Chân trọng cảm ơn!

# NỘI DUNG CHÍNH

## Cấu hình cụ thể trong bài toán

Bài toán này được sử dụng C# và được thiết kế theo cách thức của ngôn ngữ lập trình hướng đối tượng (OOP)

### Job

Đây là thành phần quan trọng nhất trong việc lưu trữ dữ liệu xử lý và quản lý các vấn đề về xử lý.

Dưới đây là mô hình Job mình thiết kế:

A close-up of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer screen

Description automatically generated

Cấu hình đối tượng Job:

* Được implement từ một interface (IjobComponent) gồm các thuộc tính

+ Properties:

* + DataTye (có thể là Bytecode hoặc String, trường hợp này đang được sử dụng String).
  + Error: Mô tả của lỗi nếu có phát sinh lỗi trong quá tình xử lý.
  + IsCompleted: Nếu Job được xử lý hoàn tất thì trả về true (phân biệt với IsError lỗi trong quá trình lý).
  + IsLocked: Nếu Job đã được truyền vào trong LocalQueue thì truy cập trường này trả về true.
  + IsProcessing: Nếu Job đang trong quá trình xử lý thì sẽ trả về true.
  + JobLockedTime: theo dự tính trước đây thì trường này được thêm vào trong quá trình log lại từ thời gian được lấy từ GlobalQueue vào trong LocalQueue và cho tới hết khi hoàn tất xử lý và được truyền ngược lại từ LocalQueue vào GlobalQueue (thường chỉ được dùng trong quá trình thử nghiệm để tính độ trễ khi truyền).
  + Message: Thông báo trả ra nếu có khi quá trình xử lý diễn ra hoàn tất (trong trường hợp IsCompleted trả về true).
  + ProcessTime: Thời gian xử lý tính từ thời điểm bắt đầu khởi chạy Hanlder cho tới khi nó kết thúc.
  + ReceiveFromQueueTime: thời điểm được lấy từ GlobalQueue vào trong LocalQueue.
  + ReceiveTime: Thời điểm GlobalQueue nhận Job tthông qua giao tiếp TCP.
  + SendTime: Thời điểm Job đượcgửi từ GlobalQueue thông qua giao tiếp TCP tới các IpAddress khác.

+ Fields:

* + HandledBy: Được xử lý bởi máy thuộc IpAddress nào?
  + ReceivedBy: Được nhận từ máy có IpAddress nào?
  + req: mã yêu cầu tự sinh.

+ Methods:

* + Clone: Được tạo ra nhằm mục đích sao chép các tham số từ properties của các Job (đối với các tham số mang tính bất biến) để không bị dùng chung bộ nhớ gây xung đột (nếu thời gian xử lý của Job dài sẽ gây ra vấn đề này).
  + Receive: Khi Job được nhận, gọi tới method này để cấu hình các tham số có liên quan tới thông tin việc nhận Job từ hàng đợi.
  + Send: Khi Job được gửi đi từ hàng đợi, gọi hàm này để cấu hình các tham số cần thiết cho quá trình gửi đi.
  + ToString: Được ghi đè để chuyển trực tiếp đối tượng sang Json.
  + FromString: Phân giải Json thành đối tượng.

### Queue

Đây là cấu hình tiêu chuẩn cho một hàng đợi được xử dụng trong kiến trúc này.

**Đây là Interface tiêu chuẩn của một Queue bất kỳ:**

A screenshot of a computer

Description automatically generated

Cấu hình Interface IQueueBasic:

* Được implement từ một interface (IJobComponent) gồm các thuộc tính

+ Methods:

* + Clear: Hủy toàn bộ Job nhận vào trong Queue.
  + Count: Trả về số lượng Job hiện tại đang được chứa trong Queue.
  + Dequeue, Enqueue: Là 2 hàm với chức năng lần lượt là Trả về và xóa, Nhận vào hàng đợi các Job theo thứ tự First in first out.
  + Peek: Tương tự như Dequeue nhưng không xóa đi đối tượng được lấy ra khỏi hàng đợi.

**Đây là Interface tiêu chuẩn của một Queue bất kỳ:**

A screenshot of a computer

Description automatically generated

Với LocalQueue, ta sẽ cấu hình thêm một Interface nữa là ILocalQueue :

* Được implement từ một interface (IJobComponent) gồm các thuộc tính

+ Properties:

* + IsCompletedSetup và Status: Là các trạng thái cấu hình của LocalQueue khi được khởi tạo
  + Cancel: Khá giống với Clear trong cấu hình của IJobComponent nhưng cho phép cấu hình thêm các chức năng chuyên biệt trong LocalQueue. (Hiện tại chúng đang giống nhau, sau này có thể cấu hình thêm, có thể chứa chức năng liên quan đến bộ đệm Cache).

**Dưới đây là sơ đồ khối biểu thị mối quan hệ của các thành phần chính (LocalQueue và GlobalQueue)**

A screenshot of a computer

Description automatically generated

A screenshot of a phone

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

BasicQueue là thành phần tiêu chuẩn cho cả GlobalQueue và LocalQueue được implement từ IQueueBasic và chứa thêm 2 hàm khác đặc trưng:

* Được implement từ IQueueBasic và chứa thêm 2 hàm khác đặc trưng

+ Methods:

* + GetIndexJobByName: Hàm lấy ra chỉ số của Job với tên định danh truyền vào.
  + GetJobByName: Lấy ra đối tượng Job với tên định danh được truyền vào.

+ Properties:

* + Jobs: Là thành phần lưu trữ các Job mà thông qua các hàm của Queue, ta có thể truy cập được chúng.

GlobalQueue là thành phần chứa các Job được nhận về từ liên kết TCP, với các Job chứa trạng thái được chỉ định nhận vào, đây là nơi chứa các Job có thể đẩy vào tiếp cho LocalQueue để đến bước xử lý.

* Được kế thừa từ BasicQueue, nó sẽ có toàn bộ tính chất của một Queue và các thông số mở rộng khác gồm

+ Methods:

* + Clone: Cho phép sao chép trực tiếp Job cần thực hiện thành một bản mới với đầy đủ các thông số cơ bản khác không thay đổi để phục vụ cho việc xử lý để không bị chồng chéo tài nguyên trong quá trình xử lý. Hàm này cũng có thể dùng trong việc gửi đi song song thành nhiều bản, việc này cũng làm giảm khả năng lỗi khi có đồng thời nhiều luồng xử lý cùng truy câp vào Job để lấy thông tin truyền đi.

+ Properties:

* + JobChange: Thay đổi trạng thái cho các Job khi nhận về, khi đẩy sang LocalQueue, khi nhận từ LocalQueue, và truyền đi các IpAddress khác từ GlobalQueue.
  + MaxCloneJob: Số bản Clone tối đa (áp dụng trong việc gửi). Việc này sẽ làm hạn chế số lượng IpAddress được nhận. Với mỗi Job, GlobalQueue sẽ chỉ Clone chúng ra với tối đa số bản được thiết lập.
  + TransferJob: Chuyển đổi kiểu dữ liệu của Job để phù hợp trong việc truyền đi hoặc nhận về.

LocalQueue là thành phần chứa các Job cần được xử lý được nhận về từ GlobalQueue. Sau khi việc xử lý hoàn tất, các Job tiếp tục được đẩy ngược lên GlobalQueue, GlobalQueue sẽ sét trạng thái các Job này và gửi các Job này đi sang các IpAddress khác.

* Được kế thừa từ BasicQueue, nó sẽ có toàn bộ tính chất của một Queue. Tuy nhiên đây là hàng đợi chứa Job để xử lý nên về mặt lý thuyết thì nó không quá khác biệt so với GlobalQueue.

### QueueManager

Đây là cấu hình tiêu chuẩn cho một hàng đợi được xử dụng trong kiến trúc này.

**Đây là Interface tiêu chuẩn của một Queue bất kỳ:**

A screenshot of a computer

Description automatically generatedA screenshot of a phone

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Với QueueManager, việc quản lý và tương tác giữa các Job và hàng đợi trở nên dễ dàng hơn:

* QueueManager chứa các tham số kết nối và hàng đợi, bộ khởi tạo Job, thông số quản lý trạng thái thiết bị

+ Properties:

* + ExternalHandler và Spawner lần lượt là 2 đối tượng (giải pháp này gắn liền với C# nên đây là 2 Action được thêm vào bên trong) với mục đích xử lý và spawn các Job.

### Qui định về căn lề văn bản

Nội dung phần chữ chọn căn đều hai bên:



Căn lề phía trên, dưới, trái, phải của văn bản như sau:





Cỡ giấy: chọn cỡ A4 trong tab “Paper”.

### Tạo lề cho văn bản in 2 mặt

Với văn bản yêu cầu in hai mặt cần điều chỉnh phần Page Layout như sau:



Chọn Pagelayout 🡪 Margins 🡪 Custom Magins 🡪 chọn mục Multiple pages 🡪 chọn Mirror margins 🡪 chọn OK.

Do trang đầu tiên là mẫu bìa của đồ án, nên khi in chế độ hai mặt có thể cần chèn (insert) 1 trang trắng sau trang bìa để đảm bảo trang “Lời cảm ơn” sẽ là trang lẻ.

Hiện tại mẫu này đang mặc định ở chế độ in một mặt.

### Tạo chương mới

Các chương đều phải bắt đầu từ một trang mới bằng cách bấm tổ hợp phím “Ctrl+Enter” tại vị trí muốn chuyển sang chương mới.

Đánh máy tiêu đề chương và chọn “CHUONG” từ thanh công cụ:



Khi đó tên chương sẽ được tự động đánh số và định dạng (tự động thêm CHƯƠNG 1, CHƯƠNG 2…). Tên chương dùng chữ viết hoa (UPPERCASE).

### Tạo tiêu đề các cấp

Sử dụng tối đa 3 cấp tiêu đề (ví dụ: 1 hoặc 1.1 hoặc 1.1.1), nếu vẫn muốn tạo thêm các mức khác thì dùng a, b, c… hoặc a), b), c)…hoặc các gạch đầu dòng.

Để tạo tiêu đề với cấp mong muốn: đánh tiêu đề cần tạo, chọn bằng cách bôi đen và chọn “Cap 1” hoặc “Cap 2” hoặc “Cap 3” tương ứng từ thanh công cụ. Khi đó tiêu đề sẽ được tự động đánh số và định dạng.



### Định dạng phần nội dung các chương, mục

Người sử dụng đánh máy nội dung cần trình bày, sau đó chọn toàn bộ đoạn văn bản đó 🡪 bấm chọn Normal trên thanh công cụ. Khi đó phần văn bản vừa tạo sẽ được định dạng đúng theo format yêu cầu của phần nội dung đồ án tốt nghiệp.



Lưu ý: người sử dụng không thay đổi đặc tính của các kiểu style (Normal, Cap 1, Cap 2, Cap 3…) đã được tạo sẵn.

### Hình vẽ - Đồ thị

Hình vẽ hoặc đồ thị (gọi tắt là hình vẽ) có hiệu quả cao khi sử dụng để minh họa cho các nội dung cần tóm lược, do vậy nên được sử dụng để tránh việc đưa các thông tin quá dài.

Hình vẽ có kích thước chiều rộng không quá 75% của chiều rộng nội dung phần chữ, căn lề giữa (trừ các trường hợp đặc biệt có thể rộng hơn hoặc sử dụng trang ngang kiểu Landscapse ).



Chú thích của hình vẽ đặt dưới hình, căn lề giữa, thứ tự hình vẽ theo số thứ tự của chương tương ứng. Để tạo chú thích cho hình vẽ thực hiện như sau:

1. Bấm chọn hình vẽ và bấm phải chuột chọn Insert Caption



1. Chọn New Label 🡪 đánh chữ “Hình” 🡪 chọn OK



1. Sau đó bấm Numbering để tạo định dạng thứ tự cho các chú thích (Hình 1.1, Hình 1.2…) 🡪 tích chọn “Include chapter number” 🡪 chọn Heading 1 🡪 chọn “period”



1. Sau đó đánh nội dung chú thích vào sau chữ Hình….



Với các hình vẽ tiếp theo không cần thực hiện lại các bước trên, chỉ cần chọn hình vẽ 🡪 bấm phải chuột 🡪 Chọn Insert Caption 🡪 bấm Label và chọn Hình 🡪 OK.



### Bảng biểu

Tương tự như hình vẽ, bảng biểu nên có chiều rộng không quá 75% chiều rộng phần chữ của nội dung. Tiêu đề bảng biểu đặt phía trên bảng với cách tạo định dạng tương tự. Bảng biểu nên bố trí để nằm trọn vẹn trong một trang, tránh việc cùng một bảng bị ngắt sang trang khác.



1. Chọn toàn bộ bảng biểu và bấm phải chuột chọn Insert Caption
2. Chọn New Label 🡪 đánh chữ “**Bảng**” 🡪 chọn OK



1. Tại ô Position: chọn Above selected item
2. Bấm Numbering để tạo định dạng thứ tự cho các chú thích (Hình 1.1, Hình 1.2…) 🡪 tích chọn “Include chapter number” 🡪 chọn Heading 1 🡪 chọn “period”



1. Sau đó đánh nội dung chú thích vào sau chữ Bảng….

Bảng . Thống kê các thiết bị và giá thành

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TT** | **Hạng mục** | **Số lượng** | **Đơn giá** | **Thành tiền** |
| 1 | Đèn bàn | 10 | 10 | 100 |
| 2 | Quạt trần | 10 | 100 | 1000 |
| 3 | Quạt bàn | 10 | 50 | 500 |
| 4 | Bàn học | 10 | 120 | 1200 |
| 5 | Ghế văn phòng | 10 | 70 | 700 |

Với các bảng biểu tiếp theo không cần thực hiện lại các bước trên, chỉ cần chọn hình vẽ 🡪 bấm phải chuột 🡪 Chọn Insert Caption 🡪 bấm Label và chọn Bảng 🡪 OK.

### Phương trình

Để tạo đánh số tự động cho các phương trình thực hiện như sau:

1. Chèn một bảng gồm {1 dòng & 3 cột} tại vị trí muốn có phương trình;

Chỉnh chiều rộng cột 1 khoảng 15% của bảng, chỉnh chiều rộng cột 3 khoảng 15-20% của bảng; còn lại sẽ là cột 2 (khoảng 70% của bảng)

|  |  |  |
| --- | --- | --- |
| Cột 1 (15%) | Cột 2 (~65-70%) | Cột 3 (15-20%) |

1. Bấm chuột vào ô của cột 2 🡪 chọn Insert trên thanh công cụ 🡪 Chọn Equation 🡪 chọn Insert New Equation.



Khi đó sẽ có bảng như sau:



Bấm vào “Type equation here” 🡪 sau đó chọn căn lề trái trên thanh công cụ, mục đích để các chữ trong cột 2 sẽ căn lề trái.



1. Chọn toàn bộ bảng 🡪 Bấm phải chuột chọn Insert Caption 🡪 New Label và tạo nhãn mới có chữ “PT” (tương tự như khi tạo chú thích với Hình và Bảng ở mục 1.2.5 và 1.2.6) 🡪 sẽ tạo ra chú thích với cấu trúc ví dụ như “PT 1.1”
2. Cắt và dán toàn bộ phần chú thích “PT 1.1” vào cột 3, sẽ có bảng như sau:

|  |  |  |
| --- | --- | --- |
|  |  | PT . |

1. Chọn toàn bộ bảng 🡪 chọn nút mở rộng của All Border trên thanh công cụ 🡪 chọn No Border



Kết quả sẽ là một bảng không có đường biên dùng để soạn phương trình:



1. Chọn lại toàn bộ bảng này và chọn Insert trên thanh công cụ 🡪 chọn Equation 🡪 “Save Selection to Equation Gallery…”



Khi đó sẽ hiện ra hộp thoại yêu cầu nhập tên của mẫu phương trình vừa tạo, người dùng có thể đánh chữ tạo tên là “Phuong trinh” để dễ nhớ và bấm OK.



Như vậy trong thư viện của Equation đã có một mẫu soạn thảo phương trình với số thứ tự của phương trình tự động thay đổi.

1. Đánh máy phương trình cần tạo vào bảng vừa tạo tại vị trí "Type equation here"
2. Để tạo phương trình tại các vị trí mong muốn khác:

Bấm Insert 🡪 Equation 🡪 kéo thanh trượt xuống dưới và chọn Phuong trình (theo tên vừa đặt)



Khi đó sẽ hiện ra bảng không có đường biên để người dùng soạn phương trình tiếp theo, số thứ tự của phương trình sẽ tự động tăng. Đánh máy phương trình vào ô “Type equation here”

|  |  |  |
| --- | --- | --- |
|  |  | PT . |

Thực hiện tương tự cho các phương trình khác.

## Tạo tham chiếu chéo giữa các đoạn văn bản

Khi viết đồ án nếu cần tham chiếu tới một mục khác hoặc hình vẽ hoặc bảng biểu hoặc phương trình…thì có thể thực hiện tự động. Tham khảo ví dụ sau:

“*Giá trị trung bình của các kết quả thí nghiệm đã được mô tả ở bảng……..; các đánh giá ở mục…. hoàn toàn phù hợp với kết quả được thể hiện ở hình…..*”

Ở ví dụ này cần tham chiếu đến đầu mục, bảng, hình vẽ sẽ thực hiện như sau:

Bấm chuột vào chỗ cần chèn tham chiếu 🡪 chọn Reference 🡪 chọn Cross-reference 🡪 hiện bảng thoại 🡪 chọn mục tương ứng của Reference type:

1. Tham chiếu tới chương, mục 🡪 chọn "Heading" 🡪 với mục “Insert reference to” chọn loại tương ứng là “Paragraph number” 🡪 chọn đầu mục tương ứng trong "For which numberred item:" 🡪 OK
2. Tham chiếu tới hình vẽ, bảng biểu: chọn mục "Reference type" tương ứng với kiểu tham chiếu và thực hiện tương tự như tham chiếu đầu mục.



## Tạo danh mục tài liệu tham khảo

Lưu ý: tài liệu tham khảo là các tài liệu được trích dẫn trong đồ án, không phải là các tài liệu đã đọc. Cách thức trích dẫn và tạo danh mục tài liệu tham khảo theo các bước sau:

* Bước 1: nhập thông tin chi tiết của từng tài liệu tham khảo.
* Bước 2: trích dẫn tài liệu tại các mục cần thiết.
* Bước 3: tạo danh mục tài liệu tham khảo

1. Bước 1: nhập thông tin chi tiết của từng tài liệu tham khảo

Chọn "Reference" trên thanh công cụ 🡪 "Manager Sources" 🡪 hiện hộp thoại "Source Manager" 🡪 chọn "New" để tạo chỉ mục cho tài liệu mới



Thực hiện tương tự các bước trên khi có nhiều tài liệu tham khảo, kết quả sẽ là một cơ sở dữ liệu của các tài liệu dự tính dùng để tham khảo như ví dụ sau:



1. Bước 2: trích dẫn tài liệu tham khảo trong nội dung đồ án

Đặt chuột tại vị trí cần chèn tài liệu tham khảo 🡪 Chọn "Reference" trên thanh công cụ 🡪 chọn kiểu trích dẫn tài liệu trong mục Style là IEEE 🡪 sau đó chọn "Insert Citation" 🡪 chọn tài liệu mong muốn.



Kết quả:

Tóm tắt nội dung của đồ án tốt nghiệp Tóm tắt nội dung của đồ án tốt nghiệp Tóm tắt nội dung của đồ án tốt nghiệp [1] Tóm tắt nội dung của đồ án tốt nghiệp Tóm tắt nội dung của đồ án tốt nghiệp Tóm tắt nội dung của đồ án tốt nghiệp [2] Tóm tắt nội dung của đồ án tốt nghiệp.

1. Bước 3: tạo danh mục tài liệu tham khảo của đồ án

Chuyển tới trang muốn tạo danh mục "TÀI LIỆU THAM KHẢO" và thực hiện theo hướng dẫn sau:

* Tạo trang TÀI LIỆU THAM KHẢO
* Đặt chuột tại ví trí đầu trang 🡪 chọn tab Reference
* Chọn Style kiểu "IEEE"
* Bấm vào Bibliography 🡪 "Work Cited".



Trong trường hợp phần số thứ tự của các tài liệu tham khảo bị lệch dòng so với phần chữ như sau:



Khi đó bấm chuột vào chữ bất kỳ trong danh mục tài liệu tham khảo đang có 🡪 bấm phải chuột và chọn "Paragraph":



Cửa sổ Paragraph sẽ hiện ra 🡪 chọn Special trong mục Indentation 🡪 bấm mũi tên xuống và chọn "(none)" 🡪 chọn OK.



Khi đó toàn bộ các số thứ tự sẽ được căn thẳng hàng với phần chữ của tài liệu tham khảo.



## Cập nhật lại các chú thích và tham chiếu

Trong quá trình soạn thảo đồ án người viết có thể thêm/bớt các đầu mục, xóa các đoạn văn không cần thiết, thêm trang, bớt trang… khi đó cần phải cập nhật lại danh mục các tham chiếu chéo. Các bước cập nhật tham chiếu chéo thực hiện như sau:

* Bước 1: bấm vào chỗ bất kỳ trong văn bản và bấm đồng thời Clt-A để lựa chọn toàn bộ văn bản (hoặc từ tab Home chọn Select rồi chọn Select All).
* Bước 2: bấm phải chuột tại chỗ bất kỳ của văn bản đã được lựa chọn sau đó chọn "Update Field" hoặc bấm phím F9.



* Bước 3: chọn Update entire table và bấm OK.

Có thể sử dụng tùy chọn "Update page number only" nếu quá trình soạn thảo chỉ làm thay đổi số thứ tự các trang.

## Tạo danh mục hình vẽ

Tạo một trang trắng tại vị trí muốn đặt "Danh mục hình vẽ", bấm chọn vị trí đầu trang và thực hiện các thao tác sau:

* Chọn tab "References" 🡪 bấm chọn "Table of Figures":



* Chọn "Hình" trong mục "Caption label" và bấm OK



Danh mục các hình vẽ sẽ được tạo tại trang mong muốn.

Để cập nhật lại đánh số trang, tên hình vẽ…thực hiện như sau:

* Đưa chuột vào vị trí danh mục hình vẽ
* Bấm phải chuột và chọn "Update Field" sau đó tùy chọn chỉ cập nhật số trang hoặc cập nhật cả trang và cả tên, thứ tự hình vẽ.

## Tạo danh mục bảng biểu

Tạo một trang trắng tại vị trí muốn đặt "Danh mục bảng biểu ", bấm chọn vị trí đầu trang và thực hiện tương tự như mục 1.6; tuy nhiên sẽ chọn mục "Bảng" trong mục "Caption label" và bấm OK.



Danh mục các bảng biểu sẽ được tạo tại trang mong muốn. Việc cập nhật lại tên bảng biểu, số trang…tương tự như với danh mục hình vẽ.

## Tạo trang mục lục

Tạo một trang trắng tại vị trí muốn đặt "Mục lục ", bấm chọn vị trí đầu trang và thực hiện các thao tác sau:

* Chọn tab "References" 🡪 bấm chọn "Table of Contents":



* Chọn kiểu "Automatic Table 1".
* Mục lục sẽ được tạo tại trang mong muốn.

Việc cập nhật lại mục lục thực hiện tương tự như với danh mục hình vẽ.

## Qui cách đóng quyển

Phần bìa trước chế bản theo qui định; bìa trước và bìa sau là giấy liền khổ. Sử dụng keo nhiệt để dán gáy khi đóng quyển thay vì sử dụng băng dính và dập ghim.





Phần gáy ĐATN cần ghi các thông tin tóm tắt sau:

Kỳ làm ĐATN - Ngành đào tạo - Họ và tên sinh viên - Mã số sinh viên

Ví dụ:

**2019.1 - VẬT LÝ KỸ THUẬT - NGUYỄN VĂN A - 20141234**

Qui cách ghi chữ phần gáy như hình sau:



# SỬ DỤNG CÁC BIỂU ĐỒ

## Giới thiệu về biểu diễn bằng đồ thị

Trong rất nhiều lĩnh vực cần phải trình bày, giới thiệu các thông tin liên quan tới con số, thống kê hay các dữ liệu khác. Các dữ liệu đo đạc, tính toán thường được thu thập dưới dạng bảng biểu; tuy nhiên bảng biểu chỉ thích hợp khi trình bày các số lượng nhỏ các số liệu, đồng thời không cung cấp các đánh giá trực quan về xu hướng của dữ liệu thu được.

Đồ thị có khả năng cung cấp hình ảnh trực quan, dễ hiểu giúp người đọc nhanh chóng nắm bắt được ý tưởng muốn nhấn mạnh, muốn trình bày. Người trình bày cần lựa chọn đúng loại đồ thị và không nên sử dụng các đồ thị quá màu mè; lựa chọn tên đồ thị ngắn gọn, dễ hiểu. Các loại đồ thị thường gặp là:

* Kiểu bánh (Pie charts)
* Kiểu thanh ngang & dọc (kiểu cột) (Horizontal & Vertical bar charts)
* Kiểu đường & Kiểu phân bố (Line charts & Scatter diagrams)
* Kiểu diện tích (Area charts)

   

Phần tiếp theo sẽ khuyến cáo về phạm vi sử dụng của từng loại đồ thị này.

## Đồ thị kiểu bánh

Phạm vi sử dụng:

* Dùng để biểu thị tỷ lệ phần trăm (%)
* Biểu diễn mối liên hệ tương quan tỷ lệ
* Không nên dùng quá nhiều miếng (tối đa 6 miếng) trong một đồ thị



Khi muốn nhấn mạnh một đại lượng:

* Để diễn tả phần quan trọng: đặt phần quan trọng này ở phía trên, bên phải, tính từ vị trí 1 giờ
* Khi cần nhấn mạnh: có thể kéo phần nhô này ra khỏi đồ thị (Hình 2.1 nhấn mạnh về tỷ trọng phần trăm của ngô là nhỏ nhất)



Hình . Đồ thị kiểu bánh

## Đồ thị kiểu thanh ngang

Phạm vi sử dụng:

* Khi muốn so sánh độ lớn hoặc kích thước
* Không nên dùng quá 5 thanh trong một đồ thị

Khi muốn nhấn mạnh một đại lượng:

* Sử dụng vị trí các thanh hợp lý để diễn tả ý muốn nhấn mạnh; không nên đặt các thanh ngẫu nhiên vì có thể gây phân tán suy nghĩ và không diễn tả được ý.
* Dùng các màu khác biệt nhiều để diễn tả đại lượng quan trọng.

So sánh 02 đồ thị trong Hình 2.2 sẽ thấy đồ thị bên trái biểu diễn được ý tưởng muốn nhấn mạnh.



Hình . Đồ thị kiểu thanh ngang

## Đồ thị kiểu cột đứng

Phạm vi sử dụng:

* Khi muốn diễn tả sự thay đổi theo thời gian
* Không nên dùng quá 5 cột trong một đồ thị

Khi muốn nhấn mạnh một đại lượng:

* Khi trình bày nên giản lược đồ thị, bỏ những dữ liệu không cần thiết
* Xem xét dùng đồ thị con để diễn tả sự đóng góp của các thành phần vào sự thay đổi theo thời gian
* Tô màu, gạch chéo hoặc dùng mũi tên để diễn tả những điểm đặc biệt

 

Hình . Đồ thị kiểu cột đứng

## Đồ thị kiểu đường

Phạm vi sử dụng:

* Biểu diễn xu hướng biến đổi của dữ liệu
* Có tác dụng so sánh nhiều dữ liệu theo thời gian
* Không nên dùng quá 3 đường dữ liệu trong một đồ thị

Khi muốn nhấn mạnh một đại lượng:

* Dùng đường nét đậm
* Đồ thị có nhiều đường: dùng nét đậm và màu nổi bật



Hình . Đồ thị kiểu đường

## Đồ thị kiểu diện tích

Phạm vi sử dụng: phù hợp để biểu diễn so sánh sự thay đổi về số lượng theo thời gian

Các lưu ý:

* Phần đáy đồ thị nên dành cho đại lượng có giá trị lớn nhất (Hình 2.5 nhân mạnh mục tư vấn đóng góp tăng trưởng mạnh nhất so với các mục khác)
* Dùng màu đậm nhất cho phần diện tích đáy; màu đậm sẽ có tác dụng tạo hiệu ứng như “neo” đồ thị, người đọc sẽ nhìn thấy và cảm thấy chắc chắn, hợp mắt
* Các tên chú thích nên để nằm ngang cho dễ đọc
* Đồ thị kiểu này cần nhiều thời gian để quan sát, do vậy nếu sử dụng cho poster thì cần dành nhiều thời gian cho độc giả tìm hiểu.



Hình . Đồ thị kiểu diện tích

# KẾT LUẬN

## Kết luận

Nội dung phần kết luận này tùy thuộc vào từng đồ án. Lưu ý trong phần kết luận không nên có bất cứ phương trình, biểu đồ hay bảng biểu nào. Cần trình bày rõ nội dung đồ án tốt nghiệp đã đáp ứng đầy đủ các yêu cầu của đề bài hay chưa. Trình bày về ý nghĩa của các kết quả thu được, các đánh giá nhận xét về tính khả thi, tính chính xác của kết quả, tính thực tế của đồ án…Cần lưu ý hạn chế sử dụng các tính từ, trạng từ mạnh trong khi miêu tả kết quả đạt được, cần đảm bảo tính trung thực của các kết luận.

Trình bày các kiến thức mà sinh viên đã đạt được sau khi thực hiện đồ án tốt nghiệp. Đồng thời trình bày về các kỹ năng đã học được (kỹ năng tự tìm kiếm tài liệu, tổng hợp thông tin, kỹ năng chế bản, kỹ năng trình bày, viết báo….).

## Hướng phát triển của đồ án trong tương lai

Nêu tóm tắt hướng mở rộng của đề tài trong tương lai nếu có. Đây là mục tùy chọn vì phụ thuộc vào loại đề tài.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | Trần Bách, Lưới điện và hệ thống điện, Nhà xuất bản Khoa học Kỹ thuật, 2004. |
| [2] | Abe Masayuki, “A Practical Approach to Accurate Fault Location on Extra High Voltage Teed Feeders,” *IEEE Transaction on Power Delivery,* pp. 159-168, 1995. |
| [3] | Microsoft, "Add citations in a Word document," 2017. |

# PHỤ LỤC

1. **Chi tiết số liệu thí nghiệm**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có).

1. **Chi tiết các bước tính toán**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có).

1. **Chi tiết sơ đồ mô phỏng**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trìn