# 「最終制作 ~CPU 温度から推測する PC への負荷~」

## 1． 筆者が PC を扱っていて悩むこと（スライド p1-2）

ネットワーク情報学部生は、MacBook を使って勉強することが多いと考えられる。そして、筆者もその一人であり、MacBook をフル活用して、大学での学びを得ている。

そのような中で、 MacBook は消耗品であるため、日々負荷がかかっており、過剰な負荷によって破損する可能性も考えられる。だが、大学生にとっては、MacBook の買い替えはとても大きな出費であり、なるべく避けたいところではある。

## 2． PC に負荷がかかるケース（スライド p3-5）

そこで、考えられる負荷がかかる場合というのは、PC が熱くなっている、すなわち CPU に過剰に負荷がかかっていると時であると言える。そして、筆者の体験談上、そのようなことが起きるのは、たくさんのソフトを同時に使用しているとき、または過剰な処理が必要になるソフトを扱っている時である。

## 3． 今回の課題（スライド p6）

そして、今回は、たくさんのソフトを同時に使用している場合を選択し、この課題を乗り越えられるようなサービスを考えていきたい。そして、もう一度課題の本質を整理すると、PC に過剰な負荷をかけずに、長く使えるようなサービスを作りたいということである。

## 4． 課題と想定していることをテスト（スライド p7-9）

今回挙げている、たくさんのソフトを同時に使用している場合に過剰な負荷がかかるのは、本当なのかテストする必要があると考え、集中講義を通じて学びを得た、ThingSpeak を利用して、様々なソフトが重なるごとに、CPU の温度が上昇するかを確かめることにする。そこで、ソフトを同時に使用する場合があると想定しても、MacBook を持っている人間が全員そのような使い方をするかと言ったら、そうではないため、ペルソナは、筆者と同じ、専修大学情報ネットワーク学部生とする。このペルソナであれば、様々なソフトを同時に利用することが考えられる。

そして、CPU の温度は、通常時 50 度前後、高い負荷がかかっている状況で 80 度を超えるとされている。だが、この温度の目安には、様々な意見が挙げられているため、あくまで参考の温度として考える必要がある。

## 5． テストの結果（スライド p10-12）

ソフトの利用状況のデフォルトを、Affinity Designer とターミナルを利用している状況

として、そこから、使用するソフトの数を増やし、計測するといった形で行った。どの状況でも 45 秒ごとに計 5 回の CPU 温度の計測を行いソフトの重なり別の温度を平等に計測した。

結果としては、重なる数が比例して温度が上昇することはなく、温度の上昇にはばらつきがあったと言える。だが、デフォルト+atom +word +music の一番負荷をかけたとされた状況の際は、ＣＰＵ温度は他と比べ高く、想定している結果が得られた。

そして、データから分かることとしては、ソフトの重なる数と CPU の温度は関係性が浅く、一つずつにかかる処理数が負荷を生んでいると考えらえた。そして、さらにグラフからわかったのは、ソフトの起動時には、必ず CPU 温度は上昇しており、起動時に負荷がかかっていることがデータからわかった。

## 6． 課題を解決するには（スライド p13）

今回の結果から課題を解決するには、一定の温度以上に達した時に、これ以上ソフトを新たに開かないで、負荷をかけないように、通知を送る、利用者に知らせることが大事であると考えた。

## 7． 考案したサービス（スライド p14）

そこで、考案したのが、LINE Notify を利用して、 CPU の温度が一定以上に達した時に通知を行うシステムである。LINE は利用率も高く、専修大学の学生の場合、incampus の通知を LINE Notify を利用して得ているため、今回のペルソナに適しているサービスであると考える。

## 8． サービスを使用する上で必要なもの（スライド p15）

サービスを利用する上で必要なものは、 CPU の温度を計測する PC 本体、そして Python、LINE Notify、LINE を受け取る PC や、スマホが必要となってくる。

## 9． システム構造図について（スライド p16）

今回のシステムの構造図は、PC の温度を、Python を通じて、取得し、その温度が一定以上になった際に、LINE Notify を通じて、通知を送るというものである。実際に温度の上昇があった場合のみ、通知は送られるため、温度の上昇が見られない場合は、通知は送られない。

## 10． 制作結果（スライド p17-18）

そして、制作期間を通じて、挑んだ制作は未完成として終わった。Python コードをうまく組み立てることができず、CPU 温度と LINE の連携を行うことができなかったことにより、システム構造図のようなシステムは構築することができなかった。

実行できなかったPythonコードはスライドのような感じになっており、簡単に説明すると、上部に書かれているソースコードがLINE Notifyを送信するためのコードであり、下部に書かれているのが、取得したCPUの温度に合わせて、通知が送られるようにするコードである。だが、そこのCPUの温度に合わせて送信する際にエラーが発生し、システムを構築することができなかった。

## 11．未完成だができたこと（スライドp19-20）

今回は完成できなかったシステムだが、授業を通じて得た理解を活かし、CPUの温度の取得はでき、LINE Notifyを利用したラインの送信もPythonのコードを通じて、行うことができた。CPU温度の取得は、前半でグラフを作ることもできたので、割愛して、LINEを実際に送った時の画面について紹介する。

スライドの画面のように、CPU温度の通知を行うことができた。テキストに書かれているように、温度の上昇によって送信されたわけではないが、このような通知が送られるシステムの構築は行うことができた。そして、このような通知が送られてきた場合、CPUの処理状況が確認しやすいとも考えられた。

## 12．完成するには（スライドp21-22）

今回のシステム構築を完成させるには、CPU温度の取得とLINE Notifyを利用したラインの送信の二つを、繋ぐ的確なPythonのコーディングが必要であると考えられる。そのため、改善点は、Pythonコードであり、それと伴い、LINE Notifyの理解が浅かったため、その点での理解を深めることは大きな改善点として挙げられると感じる。

## 13．最終制作を通じて得られたこと（スライドp23）

今回の最終制作、そして集中講義の5日間を通じて、様々な理解、学びを得ることができ、具体的には、　ThingSpeakの理解、Pythonの理解、テストの結果から得られたソフトとCPU温度の関係性、LINE Notifyの活用法など、多角的に学びを得ることができたと感じる。

## 14．最終制作の感想（スライドp24-25）

この5日間、多くの学びが得られて、学習として、とても有意義で楽しい時間であったと感じる。多くの学びを、日本語で説明できるよう、様々な理解を結び合わせて学習できたことはとても貴重な学びであり、今まで知ることができていなかった、IoTの仕組みついて、知ることができたと感じる。