Parallel Universe Computing: 仕事の集中と再開を支援するシステムの実装と評価

刈川陽平[†] 近藤秀樹[††] 小出洋[†††]

ひとつの作業環境の下で複数の仕事を並行して切り替えながら取り組む際に, 一定の集中を維持することは難しい. 特に, プログラミングなどの高度に創造的な活動にはまとまった時間での集中が必要である. 頻繁に仕事を切り替えざるを得ないことにより生じる集中の妨害は, このような集中を要する仕事の妨げとなっている. 我々はこの問題を解決するための作業環境:Parallel Universe Computingの概念に基づき, 実験的な評価を行う.

An Implementation and Evaluation of Parallel Universe Computing, An Environment Supporting Concentration on and Switch to Tasks

Youhei Karikawa,[†] Hideki Kondo[††] and Hiroshi Koide [†††]

It is difficult to keep a level of concentration on tasks when we have to do multiple tasks concurrently on an environment. Especially, highly creative tasks like programming require concentration during some time period to us. Though people often have to switch multiple tasks very frequently, it prevents us concentrating on such a task. We evaluate a user environment based on an idea of Parallel Universe Computing to solve this problem in this paper.

1. はじめに

本研究の目的は，計算機を利用した複数の仕事に取り組む際に, 仕事ごとに専用の作業環境をそれぞれ用意し，環境を切り替えることによって，ひとつの仕事に集中できるようなシステムの提案とその評価を行うことである．まとまった集中を要する仕事に対してシステムを適用し，適用しない場合との仕事の成果を比べることで，このシステムによって集中が得られるのかどうかを実験的に確かめた．

情報機器を用いた人の活動をソフトウェアごとに整理するのではなく、活動をベースに整理しようとする「活動ベースのコンピューティング[1)]」が提案されている。3D デスクトップ環境スケジューラの Cosmo Scheduler D[2)3)] にも，スケジュール項目ごとにまとまった環境を提供しようとする試みがある．また，PC 上での仕事ごとに専用の作業環境を用意して複数の仕事を支援する提案[4)] もある．しかし，これらの定量的な評価は行われておらず，実際に利用した際の仕事内容への影響は明らかにされていない．そこで本研究では実際に複数の仕事を，切り替えて利用できる環境を設計実装し，その評価を行った．

具体的には，複数の仕事ごとに仮想計算機を割り当てることにより，仕事間の干渉を意図的に避ける実験システム:Parallel Universe Computing Environment(PUCE[4)]) を実現し，複数の特徴的な仕事を行うによって実験的に確かめた．

† 九州工業大学大学院 情報工学府 情報科学専攻
†† 九州工業大学 情報工学部 情報通信技術教育センター
††† 九州工業大学大学院 情報工学研究院

2. 実　　験

2.1 実験概要

実験では被験者に対し 3 つの仕事を行ってもらい，PUCE を利用するグループ (PUCE グループ) と利用しないグループ (non-PUCE グループ) に分けて実験を行った．実験では被験者の活動をビデオカメラで撮影した動画と，計算機の画面全体のスクリーンショット，各仕事の成果物，そして事後アンケートの結果を被験者ごとにまとめた．その後の分析で，仕事の成果と仕事の切り替え回数，連続して仕事に取り組んでいた期間の長さを両グループで比較した．

2.2 実験システム:PUCE

実験環境の外観を図 1 に示す．PUCE を実行する計算機をホスト計算機，仕事を割り当てる仮想計算機をゲスト計算機と呼ぶ．ユーザはホスト計算機に接続されたスイッチ (図 2) を押すことで図 3 に示す PUCE GUI を呼び出し，パネルをマウスでクリックして選

図 1 実験環境の外観.
Fig. 1 Evaluation of PUCE.

図 2 SunSPOT.
Fig. 2 SunSPOT.

択することで仕事の切り替えを行う．また，パネル上には各仕事の直近のスクリーンショットが縮小表示される．図 4 に各仕事に対応する仮想計算機と PUCE GUI との間の状態遷移を示す.

仕事の切り替えに使うスイッチはホスト計算機に接続された SunSPOT[5] を使用した．その理由は，PUCE では作業環境として仮想計算機を用いているため，マウスやキーボードなどの仮想計算機の管理する入力方式を使って PUCE GUI を呼び出すそうとすると，ホスト計算機より先にゲスト計算機がそのイベントを拾ってしまい，結果的に PUCE GUI の制御ができなくなるためである．そこで，仮想化ソフト（VMM）が管理しないハードウェアである SunSPOT を使用することにより，ホスト計算機上で動く PUCE の制御が可能となる.

今回は仕事をスムーズに切り替えられるよう，仕事に割り当てられたゲスト計算機は全てホスト計算機のバックグラウンドで実行されている．また，PUCE は Java と AppleScript によって構成されている．SunSPOT のスイッチの入力を受けた PUCE が，AppleScript によって VMM を制御し，仮想計算機のウィンドウを最前面に表示させることで仕事の切り替えを実現している.

実験のために，本システムでは以下のようなユーザの活動情報を記録した.

全体の作業スクリーンショット:

10 秒ごとにホスト計算機に表示されている画面全体のスクリーンショットを撮影する．スクリーンショットの撮影時刻をファイル名として画像を保存し，時刻とスクリーンショットを参照することでユーザがどの時間にどのような作業をしていたのかが分かる．また，10 秒ごとにスクリーンショットを記録しているので，実験後の分析時にユーザの活動を 10 秒間隔で大まかにチェックし，その後に詳細な分析が必要になった際にはスクリーンショットの撮影時刻から動画をチェックするといった事後解析のために利用できる．さらに，ビデオ画像では詳細には読み取ることのできない画面上の文字の読み取りも可能となる.

図 3 切り替えインタフェースのイメージ.
Fig. 3 Screenshot of switching interface of PUCE.

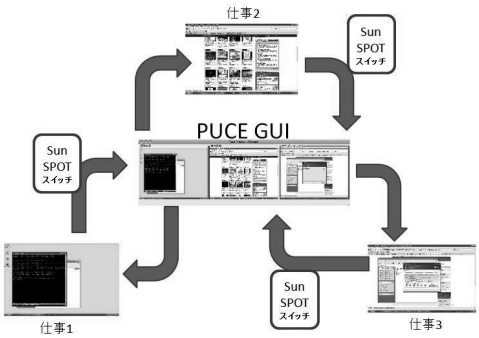

図 4 仕事の切り替え.
Fig. 4 State Transition Diagram of PUCE.

仕事の切り替えの履歴:
仕事を切り替えるときに，その時の仕事名と，切り替えた先の仕事名を記録する．記録を参照することで，仕事の切り替え時刻からユーザがその仕事に充てていた時間が分かる．

仕事の切り替え時のスクリーンショット:
仕事を切り替える際に，切り替え元の仕事の仮想計算機のスクリーンショットを撮る．撮影したスクリーンショットは PUCE GUI(図 3) 上のパネルに縮小表示される．PUCE GUI を表示したとき，ユーザが仕事の切り替え前のスクリーンショットを見ることによって，切り替え以前の仕事の状態を把握することができる．

実験に用いた計算機の諸元は表 1 の通りである．

2.3 事 前 準 備

今回の被験者の数は 6 人とした．また，被験者の間で仕事の遂行能力に大きな差が出ないよう，以下の条件を満たす者を被験者として採用した．

(1) OS の基本的な操作ができる
(2) プログラミングの経験が 4 年以上ある

条件 (1) の基準は，実験で行う仕事の内容が，キーボードで標準的な速度以上で文字を書いたり，Web ブラウジングなどの OS の基本的な操作が必要となるためである．条件 (2) は，何らかのメジャーなプログラミング言語によるプログラミングの経験があり，小規模でもソフトウェアを完成させたことのある人物を対象とした．また，理工系大学で 4 年生までに習得することのできる程度のプログラミングの知識[7] を必要とする仕事を扱うため，経験年数を 4 年以上とした．

2.4 実 験 内 容

2.4.1 実験の流れ

6 人の被験者を 3 人ずつ，PUCE グループと，non-PUCE グループに分け，実験を行った．実験は以下の手順で行った．

(1) 実験開始:
実験開始と同時にビデオカメラによる録画を開始し，被験者の背面越しにモニタの様子を撮影する．同時に，10 秒ごとにスクリーンショットの撮影を開始する．

表 1 各計算機の諸元.
Table 1 Specification.

ホスト計算機

OS	Mac OS X 10.6.1
VMM	VMware Fusion 2[6]
CPU	Intel Core 2 Duo 2.66GHz
RAM	8GB

各ゲスト計算機

OS	Windows XP SP3
CPU	Intel Core 2 Duo 2.66GHz
RAM	2GB

(2) 練習:

本実験の前に，システムの操作や作業環境に慣れてもらう目的で本実験を模した練習を行う．具体的には難易度の易しい仕事を 3 つ行う．また，実験者は PUCE グループの被験者に対して，ひとつの仮想計算機上で複数の仕事をするのではなく，複数の仮想計算機を切り替えながら仕事をしているかどうかを確認し，もしそうでなかった場合は誘導をする．約 20 分の経過を目安に練習を打ち切り，その後本実験を開始する．

(3) 本実験:

後述する本実験用の 3 つの仕事をやってもらう．本実験は実験開始から 6 時間経過した時点で打ち切る．本実験中，被験者はそれぞれ自由に休憩を取ることができるが，休憩中の時間も全体時間に含まれる．実験者は被験者の入退室と作業の中断/再開の時刻を記録しておく．

(4) アンケート:

本実験の終了後に，以下の項目でアンケートをとる．

- 作業環境について (計算機の性能など)
- 集中できたかどうか
- 仕事の難易度について
- 普段作業している環境との違いについて
- 全体の感想

2.4.2 仕事内容

実験では以下の 3 つの仕事を行う．

仕事 1 Go Lauguage:

プログラミング言語 Go について簡単なレポートを書く．レポートには Go 言語のメリットとデメリットの分かりやすい説明と，どのようなプログラミングに向いているのかとその理由，プログラムの実行方法を記述する．

仕事 2 面白い動画:

ニコニコ動画と YouTube の 2 つのサービスに掲載される動画を見て，その URL リストを作る．その中から自分が面白いと思う動画を 5 本ずつ選び，それぞれについて見所を含めた短いレビューを書く．

仕事 3 twitter bot:

Java か Ruby を用いて，twitter に自動的に発言するボットと呼ばれる種類のプログラムを作成する．ボットの仕様は，public timeline と呼ばれる全ての twitter ユーザの発言一覧から特定のキーワードを拾い，予め決めておいたキーワードがマッチすれば特定の言葉を twitter で発言する，そうでなければランダムな発言をするというものである．

2.4.3 仕事の性質

今回は性質の異なる 3 つの仕事を用意した．具体的な各仕事の性質は以下の通りである．

仕事 1:

- Web 上に断片的に点在する技術情報を探し出し，レポートにまとめる．
- 専門的だが課題の粒度は細かく，仕事の各段階や集めた技術情報の間の依存関係も多くない．
- 技術情報を探し出し，それらをレポートにまとめる際に集中を要する．

仕事 2:

- 主に娯楽性の高い動画を調査し，レポートにまとめる．
- 専門的ではない． 課題の粒度は細かく，個々の動画の間に依存関係もない．
- 動画を見ることにはそれほど集中する必要はないが，感想をまとめる際に軽度の集中を要する．

仕事 3:

- 開発環境から実行環境まで各自で用意し，プログラミングを行う．
- 専門的かつ高度な知識が多く必要で，依存関係が複雑である．また，課題の仕様に合わせるために知識や情報を応用する必要がある．
- 知識を応用する際にまとまった集中を要する．

仕事 3 の課題は他の仕事と比較して，長い時間まとまった集中を要するものである．仕事ごとに比較すると，仕事 3 が最も作業量が多く，依存関係が複雑で，多くの知識と情報を応用しなくてはならない．

2.5 事後解析

実験終了後，被験者 6 人分のスクリーンショット約 12,000 枚を一定の基準によって「仕事 1」,「仕事 2」,「仕事 3」,「中断中」の 4 つからなるカテゴリに分類した．分類には独自で開発した専用のツールを用いた．これについては 3.4 で述べる．なお，スクリーンショット間の前後関係は考慮せず，スクリーンショット 1 枚のみを見て，どの仕事に取り組んでいる時のスクリーンショットなのかを一定の基準に基づいて判断した．また，スクリーンショットの分類と同時にそのスクリーンショットの撮影時刻をひとつのログにまとめるようにした．これにより，各カテゴリごとにいつからいつまで仕事に取り組んでいたかを時間軸上で分析することが可能になった．

2.6 実験結果

以下では，実験で得た仕事の切り替え回数とそのタイミング，各仕事の成果について述べる．

2.6.1 仕事の切り替えの回数

図 5 は各被験者が仕事を切り替えた回数をグラフ化したものである．グラフより，PUCE グループと non-PUCE グループの被験者の仕事の切り替え回数を比べると，PUCE グループのほうが仕事の切り替え回数が少なくなっていることが分かる．

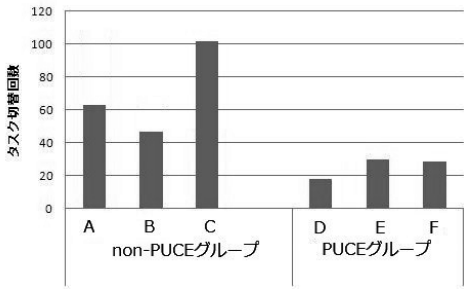

図 5 切り替え回数.
Fig. 5 Number of switching tasks.

2.6.2 仕事の切り替えのタイミング

図 6，7 は各被験者の仕事の切り替え状況を仕事ごとに時間軸上に表現したものである．各色は各仕事と中断していた時間に対応している．左端が実験の開始で，右端が実験の終了となる．この期間を約 6 時間とする．また，PUCE グループの被験者をそれぞれ A，B，C，non-PUCE グループの被験者をそれぞれ D，E，F と分類した．

図 6，7 より，non-PUCE グループの被験者は頻繁に仕事の切り替えを行っており，連続した期間で仕事に取り組めていない．それに比べ，PUCE グループの被験者の仕事の切り替え頻度は低く，連続した期間で仕事に取り組めているのが分かる．また，図 6 の被験者 B の後半部分は一見すると連続した期間で仕事に取り組めているように見える．しかし，これは被験者 B が既に実験の前半で仕事 1,2 をほぼ完了したものと誤解して，仕事 3 のみに取り組んだ結果である．つまり，被験者 B が後半部分に連続した期間で仕事に取り組めているのは，後半でやるべき仕事は残りひとつしかないので仕事を切り替える必要が無いと判断していたためだと考えられる．このことを考慮した上で被験者 B の後半部分を除く前半部分を見ると，頻繁に切り替えが発生しており，連続した期間で仕事に取り組めていないことが分かる．

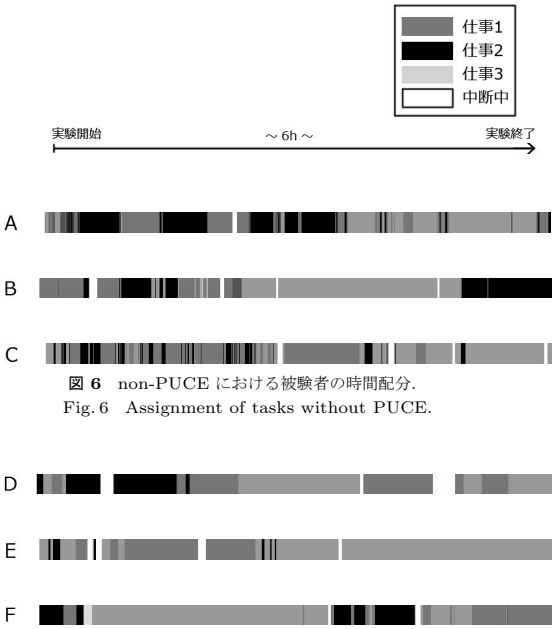

図 6 non-PUCE における被験者の時間配分.
Fig. 6 Assignment of tasks without PUCE.

図 7 PUCE を利用する条件における被験者の時間配分.
Fig. 7 Assignment of tasks with PUCE.

2.6.3 仕事の成果

実験終了後に被験者が使用していた仮想計算機環境から課題の成果物を取り出し，課題の中でいくつの評価項目を達成できているかどうかを検証する．

検証の結果は表 2 に示す．表の網掛け部分は特に集中を要する項目である．これについては評価 3.2 で考察する．

3. 評価と考察

3.1 集中の持続

2.6.1 の結果から，PUCE グループと non-PUCE グループの仕事の切り替え回数を比べると，PUCE グループの方が少ないことが分かった．また，2.6.2 の結果から，PUCE グループの被験者は，non-PUCE グループの被験者に比べて連続した期間で仕事を進められていることが分かった．このことから，PUCE を利用することで作業の切り替え回数が減り，連続した期間で仕事を進めることができるという傾向が読み取れる．

次に，PUCE を利用したことで，なぜ切り替え回数が少なく連続した期間で仕事を進められているのかについて考察する．

3.1.1 隔離された環境としての仮想計算機の利用

要因のひとつに，PUCE で用いた仮想計算機が他

表 2 各被験者の課題達成リスト.
Table 2 Achievement of each subject.

		non-PUCE グループ			PUCE グループ		
	被験者	A	B	C	D	E	F
仕事 1: 技術的な調査 Go Language について調査	全体的な特徴の記述があるか	○	○	×	○	×	○
	Go のメリットの記述があるか	○	○	○	○	×	○
	Go のデメリットの記述があるか	○	○	×	○	×	○
	どのような用途に適しているかの記述があるか	×	○	×	×	×	△
	何故そのような用途に適しているかの記述があるか	×	○	×	×	×	△
	実際に実験的に動かしてみるには どうすればよいかの記述があるか	○	×	×	×	×	○
仕事 2: 流行調査 おもしろい動画について調査	YouTube 上の URL が 10 件挙げられているか	× (0)	× (0)	○ (10)	× (0)	○ (10)	× (0)
	ニコニコ動画上の URL が 10 件挙げられているか	○ (10)	× (0)	○ (10)	○ (10)	○ (10)	○ (10)
	YouTube について何本レビューしているか	× (0)	× (4)	○ (5)	× (0)	○ (10)	× (0)
	ニコニコ動画について何本レビューしているか	○ (5)	○ (6)	○ (5)	○ (5)	○ (10)	○ (5)
仕事 3: プログラミング Twitter ボットの開発	twitter について調査した	×	○	○	○	○	○
	twitter ライブラリ (自作を含む) を導入する	×	○	○	○	○	○
	Timeline から発言を取ってくる	×	×	○	×	○	○
	Tweet をつぶやくことができる	×	○	○	×	○	○
	キーワードを探して反応を返すロジックが 出来ている	×	×	×	×	○	○
	ランダム発言を行うロジックが出来ている	×	×	×	×	△	○

の仕事と隔離された環境であったことが考えられる．non-PUCE ではひとつの環境で複数の仕事が進行していたため，仕事同士が互いに干渉し合い，妨害関係になっている可能性がある．例えば，図 8 の non-PUCE グループの被験者のスクリーンショットで，他の仕事に妨害されている可能性が見て取れる．この図では，仕事 3 の作業中に，バックグラウンドのウィンドウで仕事 2 の YouTube の動画が同時に表示されていた．そのため，YouTube の動画がきっかけとなり，ウィンドウのフォーカスが仕事 3 で使う Web ブラウザから，仕事 2 で使うテキストエディタへと切り替わってしまっている可能性が高い．このように，ひとつの作業環境で複数の仕事のウィンドウが画面に並ぶことにより，互いが妨害関係になっている可能性が考えられる．この問題を解決するために，仕事で用いる作業環境を他の仕事と隔離することを考える．

既存の環境として，仮想デスクトップというものがある．前述の問題はこの仮想デスクトップを用いることで隔離された環境を実現し，解決できるようにみえるが，必ずしもそうではない．仮想デスクトップでは，擬似的にウィンドウやファイルのアイコンが見えなくなっているだけで，これらの実体は同一の作業環境内に混在している．この状態では完全に隔離された環境とは言い難い．なぜなら，同一の作業環境内ではウィンドウ，タスクメニューなどのプロセスは共有して管理されているので，アプリケーションのアラートなど，ユーザの意図しない形で別の仕事のプロセスが見えてしまう可能性があるからである．

それと比べて，PUCE では仕事ごとに作業環境が存在するので，各仕事がもつ，ウィンドウやタスクバー，ファイルのアイコンやアプリケーションの設定などは全て隔離された環境内に収まり，仕事同士が互いに干渉することは少ない．よって PUCE を用いることで，進行中の仕事は他の仕事に妨害されることがなく，まとまった集中を得やすいと考えられる．

3.1.2 仕事の切り替えにかかる適度な負荷

もうひとつの要因に，仕事の切り替えにかかる負荷が被験者に対して適切に働いていたことが考えられる．ここで言う切り替えにかかる負荷とは，切り替えるための手段の煩雑さのことを指す．例えば,「背後に隠れたウィンドウをクリックしてアクティブウィンドウにする」場合，ウィンドウを探してきてクリックをするときの手間が切り替えのときの負荷にあたる．このような仕事の切り替えにかかる負荷の大小は，そのときの切り替え方法によって様々である．例えば，Windows 系 OS ではタスクメニューのクリックや，Alt + Tab キーを押すなどの手段で仕事の切り替えを行っている．これらの機能を利用することで仕事の切り替えをスムーズに行うことができるが，必ずしもメリットばかりであるとは限らない．あまりにも切り替えにか

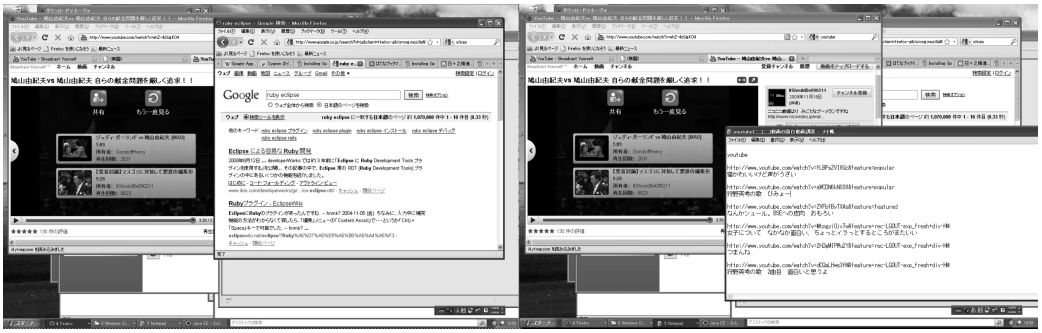

図 8 切り替え前と切り替え後.
Fig. 8 Before switching and after switching.

かる負荷が軽いと，他の仕事で使うウィンドウが見えていただけでクリックをしてしまう可能性があり，簡単に仕事を切り替えることができてしまうので，ひとつの仕事に集中できなくなる可能性がある．

PUCE ではホスト計算機に外付けされたスイッチを押すことによって仕事の切り替えを行う．このように，マウスでもキーボードでもなく，外付けされたスイッチを押すという非標準的な切り替え方法で負荷をかけることによって，気軽に仕事を切り替えられないようにし，仕事への集中が途切れないようにしている．

3.2 仕事の成果の比較

次に，仕事の成果について考察する．

仕事 3:プログラミングで達成すべき項目のうち，「キーワードを探して反応を返す」と「ランダムに発言をする」は特に集中を要するものである．なぜなら，これらを達成するためには，前提条件として public timeline などの twitter の構造を理解しておかなければならず，さらに twitter で発言をするという機能を先に実装しておく必要がある．また，twitter ライブラリなどの既存の技術を用いる場合に，課題の仕様を実現するためにこれらの技術の応用を考えなくてはならない．このように，解決すべき依存関係が複雑になり，ある一定の長時間まとまった集中を要する．

表 2 より，PUCE グループと non-PUCE グループとの仕事の成果を比較すると，仕事 3 でやるべき事項の「キーワードを探して反応を返す」と「ランダムに発言をする」を達成できているのは PUCE グループの被験者のみである．ここで重要なのは，仕事 3 はまとまった集中を要する，ということである．これらを達成できているのが PUCE グループの被験者のみであるということは，PUCE を利用することで，まとまった時間で集中して仕事に取り組むことができる可能性があると考えられる．

逆に，2.6.2 より，non-PUCE グループの被験者は連続して仕事に取り組むことが少なく，他の仕事に切り替える頻度が高いのが分かる．仕事 1 と仕事 2 の性質上，これらの課題は粒度が小さく依存関係も少ないので，まとまった集中を必要としない．その結果，non-PUCE グループの被験者は仕事 1 と仕事 2 に関しては断片的に作業をし，ある程度の成果物を上げることができるものの，まとまった集中を要する仕事 3 に関しては，集中が持続せず，連続して仕事に取り組むことができていないと考えられる．

3.3 プログラミングでの集中の必要性

実験から，プログラミングという仕事には集中が重要であることが分かった．一般的に創造的な仕事は集中して取り組むことが必要であるとされており，プログラミングもそのひとつとして扱われることが多い．断片的な時間を単に組み合わせるだけでは充分ではなく，連続して課題に取り組むことで達成される課題であることが分かった．

3.4 分析ツールの効果

本研究では実験データの分析のために，人間の長所を生かす小さなツールを開発した．例えば，スクリーンショットの分類ツールによって，人間の持つ認識能力をうまく利用して,「活動内容」のような曖昧な概念を扱う研究が効率的に可能になった．以下にその効果を説明する．

2.5 で述べたように，本研究ではユーザの活動内容によってスクリーンショットを分類する必要があった．分類の基準はあらかじめ可能な限り明瞭なレベルで記述したが，たとえば「アクティブウィンドウに”go” ”language” ”言語”が含まれていたら，技術調査の仕事とする.」というような基準である．人間であれば充分高精細な画像を読み取ることで容易に判断でき，かつ判断の内容にぶれが生じることは少ないが，計算

機で処理することは現実的ではない．逆に，見るべき画像を間違えないように見て，判断した画像を間違いなく所定の場所へ仕分けし，適切な記録を作るという操作は人間には大変苦痛であるが，計算機にとっては自動的に行えることである．

そのため，一枚ずつ画像を掲示し，それを人間が見て判断して，合致する判断基準となるボタンをクリックすることで，計算機が自動的に仕分けと記録を行い，次の画像を掲示する，というツールを開発した．開発は 3 時間程度で終了し，約 12,000 枚のスクリーンショットを，一人の人間が休憩を含む 10 時間程度で仕分けることができた．ツールの開発期間は 3 時間程度であり，それを合算しても，約 4 秒に 1 枚の画像を分類できたことになる．一般に流通しているアプリケーションやツールではこの効率は達成できなかった可能性が高い．

4. まとめと今後の課題

複数の仕事ひとつひとつに仮想計算機を割り当てることで，集中を支援するシステムである PUCE を提案，実装し，その評価を行った．その結果，PUCE を利用することで，複数の仕事に取り組むときに，個々の仕事にまとまった時間で集中して取り組むことができるということが分かった．

今後の課題として，長期の中断からの再開についての評価についても行いたい．例えば，1 週間前に中断した仕事を再開しようとするときに，他にも複数の仕事を抱えていた場合，作業環境の中に残った別の仕事の痕跡が枷となり，1 週間前にやっていた内容を思い出す作業に時間を費やしてしまうことがある．そこで PUCE のように複数の仕事ごとに作業環境を用意することでこの問題を解決できるのかというと，必ずしもそうであるとは言い切れない．なぜなら，今回の PUCE を用いた評価は，6 時間という短い期間の中での評価であり，1 日や 1 週間，1 年といった長期の仕事に対して PUCE のようなシステムを適用した場合，どのような影響があるのかは明らかではない．そこで今後は，長期の中断が仕事にどのような影響があるのか評価をし，解決できるような手法を考えていきたい．

もうひとつの課題に，仕事の切り替えを容易にすることの検討がある．3.1.2 では切り替えが容易に行えることによって，仕事の切り替え回数が増え，集中の妨害に繋がる可能性があることが分かった．しかし，場合によって切り替えが容易に行えることが良い影響をもたらす可能性も考えられる．PUCE の実装では VMM のウィンドウを制御することによって切り替えを実現しているが，これには現在のところ秒単位のオーバーヘッドがかかる．これはユーザにとって負担となっていることがアンケートからも読み取ることができた．ユーザの負担を減らすために，切り替えを容易にすることが良い結果をもたらすのかどうかは明らかではないが，評価をする価値は充分あると考えている．今回の「切り替えの回数が少ないと，まとまった集中を得ることができる」という評価に対し，この方針は性質が異なるものなので，充分な比較が必要であると考える．今後は，切り替えを容易にした実験的なシステムで新たに評価を行い，今回の評価と比較し，本手法をさらに改善する予定である．

謝　　辞

本研究は科研費 (18100001) の助成を受けたものである．

参　考　文　献

1) Donald A.Norman(著), 岡本明 (訳), 安村通晃 (訳), 伊賀聡一郎 (訳) : パソコンを隠せ、アナログ発想でいこう!–複雑さに別れを告げ、“ 情報アプライアンス ” へ,pp.108-112, 新曜社 (2000).
2) Kouji Yakushiji, Yoshifumi Maeda, Hirokazu Minamisako, Hiroshi Koide : *Cosmo Scheduler D*, http://www.klab.ai.kyutech.ac.jp/cosmo/, 2009 年 11 月 29 日確認
3) 薬師寺浩二，前田良史，南迫博和，小出　洋 : 新しい 3 次元アプリケーションを作るための CosmoAPI の設計と評価，日本ソフトウェア科学会第 22 回大会論文集 CD-ROM，2B-1 (2005).
4) 近藤秀樹, 刈川陽平, 小出洋 : 予定ベースの計算機フロントエンド, IPSJ Programming Symposium, (2009).
5) Sun Microsystems, Inc., *Sun SPOT*, http://sunspotworld.com/, 2009 年 11 月 29 日確認
6) VMware, Inc., *VMware workstation - Run Multiple OS Including Linux on Windows on Virtual Machines*, http://www.vmware.com/products/ws/, 2009 年 11 月 29 日確認.
7) 情報処理学会，大学の理工系学部情報系学科のためのコンピュータサイエンス教育カリキュラム J97, http://www.ipsj.or.jp/12kyoiku/J97-v1.1.pdf, 2009 年 11 月 29 日確認