ソフトウェア工学 107－4
（1996．1．18）

# バグ作り込みにおけるストレス影響度の設計法による差の分析

古山恒夫　新井義夫　飯尾和彦
furuyama@slab.ntt.jp
日本電信電話（株）ソフトウェア研究所
〒180　武蔵野市緑町3-9-11

ソフトウェア品質の向上には、ストレスの軽減などの作業環境の改善が重要である。しかし、実際には納期切迫などの過度のストレスが不可避なことが多い。本論文では、ふたつのチームに心理的ストレスをかけながら機能的設計法と構造化設計法という異なる設計法で同じソフトウェアを開発させた実験結果を報告する。分析の結果、ストレス起因のバグ発生率は、FD工程では両設計法で差がないが、DD工程では構造化設計法の発生率が機能的設計法の半分であることがわかった。このことから、作り込みバグを減らすためには、より構造化された設計法を用いてソフトウェアを開発することが、心理的ストレスの不可避な環境でも重要であることがわかった。

# ANALYSIS OF THE DIFFERENCE OF FAULT GENERATION CAUSED BY MENTAL STRESS BETWEEN TWO DESIGN METHODOLOGIES

Tsuneo Furuyama, Yoshio Arai, and Kazuhiko Iio

NTT Software Laboratories, 3-9-11 Midori-cho, Musashino-shi, Tokyo 180, Japan

This paper describes the difference of faults generation caused by mental stress between two design methodologies, functional design methodology and structured design methodology. In the FD phase, Team A using functional design methodology and Team B using structured design methodology had the same generation rate of faults caused by mental stress. In the DD phase, however, even under the same stress conditions, Team B generated only a half of faults caused by mental stress that Team A generated. Therefore, it is important not only to establish better circumstances, but also use more structured design methodology for improving reliability.

## 1．はじめに

ソフトウェアの開発管理において信頼性の評価／向上技術は最も重要な技術のひとつである。信頼性の評価／向上技術は、ソフトウェア開発の歴史の初期には開発の下流工程すなわち試験段階でいかにしてバグをとり、残存バグを推定するかが重視されていた。その後、上流工程で信頼性を評価／向上することの重要性が認識されるようになり、フォーマルインスペクションなどの設計レビューが脚光をあびるようになってきた。

ソフトウェアの信頼性を更に向上するためには、このようなバグ摘出技術だけでなく、開発者によるバグ作り込みを減らす技術が重要である。機械工学分野においては、システムの信頼性に最も大きな影響を及ぼす要因としてヒューマンファクタに関する議論が盛んに行われている[1][2]。その中でも、Rusmussen [3]らは、認知科学の手法を取り入れたヒューマンエラーに関するモデルを構築し、ヒューマンエラー発生のメカニズムの枠組みを提案している。しかし、ソフトウェア開発の分野においては、システムの複雑さや設計情報の伝達・管理の面からの議論は行われている[4][5][6]が、ヒューマンファクタの面からの議論はほとんど行われていない。我々は、ソフトウェア開発プロセスの中心が人間であることを重視したモデル、すなわち、バグ作り込みの原因となったエラー要因とそれによって発生したエラー現象を組み合わせたバグ作り込みの抽象モデルを提案した（図1）。それをベースにバグの原因を徹底的に究明して、ストレスと人間の作業特性がバグ作り込みに大きな影響を与えていることを定量的に明らかにした[7]。

また、ストレスがどの程度かかったときにバグ作り込みにどの程度の影響を与えているかを、実験により定量的に明らかにした[8]。その実験では、２つのチームに同じソフトウェアを開発させ、片方のチームにだけストレスをかけた。ストレスのかかり具合を、心理的な状態を測る内観メトリクス（本人の申告による主観的な尺度）を用いて測定し、内観メトリクスがストレス起因のバグ数予測に有効であることを明らかにした。

一方、Takahashiらは、同じ仕様のソフトウェアを２つのチームにそれぞれ構造化設計法および機能的設計法で開発させて信頼性と生産性を比較した結果、構造化設計法の方が機能的設計法より開発したソフトウェアの信頼性が向上することを明らかにした[9]。しかし、その実験では特にストレスなどの心理的な開発環境については考慮していない。

今回、ストレス起因バグの信頼性に与える影響が、設計法によりどのように違うかを明らかにするために実験を行った。

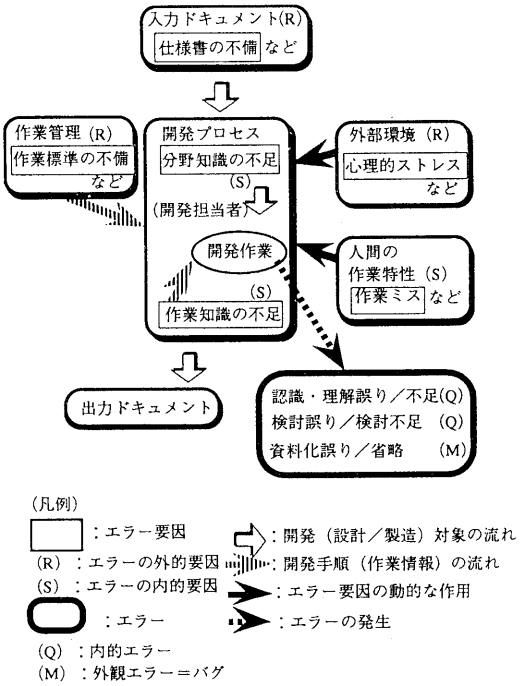


図１　バグ作り込みの抽象モデル [1]

## 2．実験方法

### 2．1　概要

技術力に差のない２つのチームに同じ要求仕様のソフトウェアを、設計法以外は同じ条件で開発させた。２つのチームとも、開発期間の短縮と要求仕様変更の心理的ストレスをかけた。この開発過程で、開発者のストレス度、作り込みバグ数、作業状況などのデータを収集した。

### 2．2　開発方法

（１）開発システム

C言語で約４ksの、図書の貸出、返却、検索、登録機能などを支援する図書管理システムを開発した。

（2）設計方法

機能的設計法と構造化設計法を採用した。2つの設計法の比較を表1に示す。構造化設計法では、機能的設計法より多くの情報を構造化した図として明示するところに特徴がある。

表1　構造化設計法と従来設計法の比較

| 工程 (*1) | 構造化設計法 | 従来設計法 |
|---|---|---|
| FD | 機能仕様書<br>データフロー図<br>データ構造図 | 機能仕様書<br>-<br>- |
| DD | ストラクトチャート(*2)<br>HCPチャート | モジュール構造図<br>HCPチャート |

(*1) FD：機能設計、DD：詳細設計
(*2) モジュール間のデータフローを加えたモジュール構造図

（3）チーム編成

各チームは2人1組で構成した。チーム間で技術力の差がないように、開発者の設計能力を評価し、技術力の比較的高い開発者と低い開発者で1チーム（Bチーム）、中程度の2人で1チーム（Aチーム）を構成した。Aチームは機能的設計法、Bチームは構造化設計法をそれぞれ用いて上記システムを開発した。実験では、思考過程のない清書作業以外は、設計作業を分担しないで、2人で一緒に考えるようにさせた。

（4）作業環境

作業環境は、チーム間に差がなく、且つ相互干渉が起きないように配慮する必要がある。このため、各チームには作業室の広さや照明、騒音の程度などに差のない作業室を別々に割り当てた。

（5）開発工数

開発期間は約3ケ月である。上記2つの設計方法では工程毎の開発工数の分布に差がある[10]ため、それらの違いを考慮した工程を標準工程として設定した。すなわち、構造化設計法の方が、FD工程で50%、DD工程で30%、機能的設計法より開発期間が長い。表2に実際の開発工数を示す。

## 2．3　ストレス環境とその測定

（1）　ストレスのかけ方

ソフト開発作業においては、納期の時間的切迫や仕事量の多さなどがストレスの大きな要因である[8] [11]。この要因の影響を分析するために、今回の実験ではFD、DDのそれぞれの工程において、A、B両チームに対し、次のようなストレスをかけた。

(a) 設計期間を標準の7割に短縮
(b) FD工程で3回の要求仕様の変更

表2　開発工数（人時）

| 工程 | Aチーム | Bチーム |
|---|---|---|
| FD | 119 | 192 |
| FDレビュー | 98 | 70 |
| DD | 250 | 321 |
| DDレビュー | 100 | 59 |
| M/DB | 180 | 158 |
| 合計 | 747 | 800 |

（2）ストレス測定メトリクスの選定

心理的ストレスの測定方法には、生理学的手法（脳波、心電図、心拍数、呼吸量など）、生化学的手法（尿、血液中のカテコールアミンなどの分析）、心理的手法（被験者が質問用紙に回答する形式でストレス状態を書かせる）などがあるが、このうち心理的手法が簡便かつ有効である[8]。「線表の厳しさ」という評価尺度(内観メトリクス）が心理的ストレスの検出能力が高く、「仕事量の多さ」がバグの発生と相関が高いという結果に基づき、今回の実験ではこの2つをストレス測定メトリクスとして選んだ。

（3）ストレス値の測定法

今回の実験では、その日に感じたストレス程度を「普通」を基準に前後合わせて6段階で記入できるようにした（図2）。これらのストレス値は毎日、作業終了時に測定した。

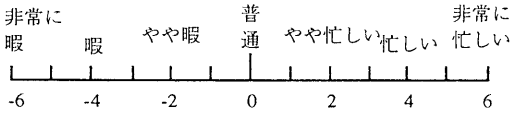


図2 内観メトリクス（仕事量が多い）の測定シート

## 2．4　バグの発生状況の記録

バグの発生時刻と発生原因を正確に把握できるように、開発担当者に毎日詳細な作業日誌を記録させた。作業日誌には、バグ内容、バグを生みだしたエラー原因、バグ発生日などを記録する。また、作業には、エラー要因とストレスの強さの関係を容易に明らかにできるようにするために、そなお、単なる誤字や脱字はバグから除外した。

## 3．実験結果の分析と考察

### 3．1　ストレスのかかり方

（1）心理的ストレスメトリクス値

4人の被験者（開発担当者）が受けた心理的ストレスと身体的ストレスの強さを図3に示す。ただし、この実験プロジェクトを開始する前の数日間の安静時の状態をストレスレベル0（普通）とした。

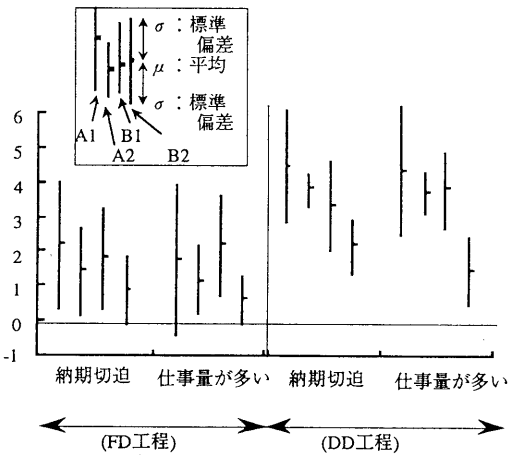


図3 メトリクス毎の4人の平均と標準偏差

図3および両チーム間のストレスレベルの平均値の差の検定結果（表3）の結果から次のことが言える。ただし、同じメトリクス値は被験者によらず同じ程度ストレスがかかっているものと解釈している。

(a) FD工程では両チームに同程度のストレスがかかっている。

(b) DD工程ではAチームの方がBチームより多くのストレスがかかっている。ただし、開発担当者B2を除くと、両チーム間に差は見られない。

### 3．2　バグの分類

発生したバグは、エラー原因により、静的バグと動的バグに分ける。静的バグは、図1のモデルにおいて、入力情報（what）に起因するバグ、作業管理（how）に起因するバグ、知識不足（what, how）に起因するバグの総称である。動的バグは、ストレスに起因するバグおよび人間の作業特性（以下単に人間特性と呼ぶ）に起因するバグの総称である。

バグの分類にあたっては、静的バグ＞ストレス起因バグ＞人間特性起因バグの順で優先順位をつけている。すなわち、高い心理的ストレスがかかっている状態で起きたバグでも、もともと仕様書が誤っていた場合は静的バグに分類した。また、静的バグ以外のうちストレスに起因していることが明確なものをストレス起因バグとして、残りを人間特性起因バグとした。

表3　心理的メトリクスのAチームとBチームの平均値の差の検定

| 工程 | メトリクス | Aチーム 平均 | Aチーム 分散 | Aチーム 観測数 | Bチーム 平均 | Bチーム 分散 | Bチーム 観測数 | 自由度 | T値 |
|---|---|---|---|---|---|---|---|---|---|
| FD | 納期の切迫 | 1.81 | 2.544 | 32 | 1.35 | 1.721 | 52 | 56.3 | 1.39 |
| | 仕事量が多い | 1.47 | 2.838 | 32 | 1.39 | 1.888 | 52 | 55.9 | 0.24 |
| DD | 納期の切迫 | 4.16 | 1.547 | 62 | 2.75 | 1.534 | 68 | 126.8 | 6.48** |
| | 仕事量が多い | 4.03 | 1.999 | 62 | 2.62 | 2.449 | 68 | 128.0 | 5.42** |

**：95％有意

### 3．3　作り込みバグ

チーム（設計法）別、工程別、エラー原因別の作り込みバグ数の分布を表4に示す。

表4　発生バグ数の分布

| 工程 | エラー原因 | Aチーム（機能的設計法） | Bチーム（構造化設計法） | 合計 |
|---|---|---|---|---|
| FD | 静的バグ | 14 | 15 | 29 |
| | ストレス起因バグ | 20 | 14 | 34 |
| | 人間特性起因バグ | 22 | 15 | 37 |
| | （小計） | 56 | 44 | 100 |
| DD | 静的バグ | 8 | 13 | 21 |
| | ストレス起因バグ | 85 | 42 | 127 |
| | 人間特性起因バグ | 10 | 23 | 33 |
| | （小計） | 103 | 78 | 181 |
| 合計 | | 159 | 122 | 281 |

#### 3．3．1　全体傾向

(a) 両チームともDD工程のバグがFD工程のバグの約2倍である。

(b) FD、DD工程とも機能的設計法チームのバグ

が構造化設計法チームのバグより25-50%程度多い（FD＝61：41、 DD＝104：82）。なお、両チームの違いについては、4。2の設計法による心理的ストレス影響度の違いで詳細に分析する。

3．3．2　エラー原因別分析

(a) 静的バグ数はFD工程では両設計法間で有意差はない（A: B＝14：15）が、DD工程では有意差が見られる（A: B＝8：13）。Bチームで発生した静的バグの多く（6件）は設計法（HCPチャート記法）の知識不足であり、固有事情と考えられる。

(b) 静的バグの原因は、チームによらず、FD工程では分野知識不足（A: B＝11／14：11／15）が、DD工程では作業知識不足（A: B＝5／8：10／13）が主な原因となっている。これは、FD工程では分野知識の占める割合が大きいのに対し、DD工程では、分野知識より設計法の知識が必要であることを示している。

(c) 両設計法ともストレス起因バグの占める比率が高い。FD工程では1／3以上（機能的設計法＝36%（20件）、構造化設計法＝32%（14件））を、DD工程では1／2以上（機能的設計法＝82%（85件）、構造化設計法＝54%（42件））を占める。

(c) 人間特性起因バグは、FD工程ではストレス起因バグと同程度であり、DD工程ではストレス起因バグより少ない。

## 4．分析

図4の分析モデルに従って設計法による心理的ストレス影響度の違いを分析した。

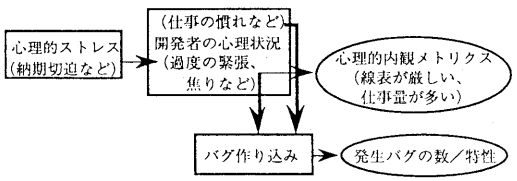


図4　分析モデル

### 4．1　バグ発生率の計算

しばしば残業が発生したこと、および午前と午後の半日毎の作業時間が大きく異なることから、単純に半日を単位として作り込まれたバグ数同士を比較するのは適切ではない。そこで、われわれはストレスレベル毎に、累積発生バグ数／累積作業工数を計算して、単位工数あたりの発生バグ数を求めた（表5）。なお、表5のストレスレベルは「仕事量が多い」の値を用いている。

ストレスレベル毎のバグ発生率を求めるには2つの考え方がある。ひとつは実際の工数で発生バグ数を割るものであり、他の方法は正規化された工数で発生バグ数を割るものである。後者は、ファンクションポイント法と類似の概念に基づいており、作られた機能あたりのバグ数と言える。言い替えると、同じ機能を作るのにもともとそれぞれの設計法で必要な標準単位工数で割るべきである、という考えに基づいている。本論文では後者を採用した。具体的には、Aチームは実工数で計算しているが、Bチームの場合は、ストレスレベル毎に実工数でバグ発生率を計算後に、AチームとBチームのFD工程の作業工数比（191.5／119）で正規化（掛け算）している。

### 4．2　チーム間のバグ発生率の比較

表5に示した結果から、次のことがわかる。

(a) FD工程におけるAチームの心理的ストレス起因バグ（単位時間あたり0.17件）はBチーム（単位時間あたり0.07件）よりも多い。しかし、この差はAチームにおけるストレスレベル6でのバグの多発（Bチームではストレスレベル6に達したことはなかったのでこのレベルでのバグはない）によるものであり、これを除くとFD工程では正規化したバグ発生率は両チーム間で差はない。

(b) DD工程では、Aチームのバグ発生率は、Bチームのバグ発生率よりも高い。この差はストレスレレベル4で、両チーム間に大きな差があるためである。

(c) FD工程における人間特性起因バグの発生率は、レベル3での差を除いて両チーム間で有意な差はない。レベル3では、Aチームのバグ発生率は非常に高いが、Bチームで該当するデータがないため比較できない。

(d) DD工程における人間特性起因バグに関しては、Aチームのバグ（10件）がBチームのバグ（23件）に比べて非常に少ない。

(e) Aチームの動的起因の総バグ数が、ストレスレベルの違いを考慮してもBチームより非常に多

い。このことは、機能的設計法よりも構造化設計法を用いる方が、高橋らが指摘しているようなストレスのない環境の場合[10]だけでなく、過度のストレスのある環境の場合も開発したソフトウェアの信頼性が高まることを示している。

表5　ストレスレベル毎の動的バグ発生数

| 工程 | ストレスレベル | Aチーム 作業時間 | Aチーム バグ数 ストレス起因バグ | Aチーム バグ数 人間特性起因バグ | Bチーム 作業時間 | Bチーム バグ数 ストレス起因バグ | Bチーム バグ数 人間特性起因バグ |
|---|---|---|---|---|---|---|---|
| FD | 0 | 41.25 | 2 (0.05) | 6 (0.15) | 49.75 | 0 (0.00) | 3 (0.10) |
| | 1 | 24.75 | 1 (0.04) | 2 (0.08) | 66.50 | 1 (0.02) | 5 (0.12) |
| | 2 | 16.50 | 1 (0.06) | 2 (0.12) | 31.25 | 2 (0.10) | 0 (0.00) |
| | 3 | 9.50 | 2 (0.21) | 6 (0.63) | 0.00 | -- | -- |
| | 4 | 19.00 | 9 (0.47) | 6 (0.32) | 44.00 | 11 (0.40) | 7 (0.26) |
| | 5 | 0.00 | -- | -- | 0.00 | -- | -- |
| | 6 | 8.00 | 5 (0.63) | 0 (0.00) | 0.00 | -- | -- |
| 合計 | | 119.00 | 20 (0.17) | 22 (0.19) | 191.50 | 14 (0.07) | 15 (0.08) |
| DD | 0 | 4.25 | 0 (0.00) | 0 (0.00) | 21.75 | 0 (0.00) | 1 (0.06) |
| | 1 | 2.00 | 0 (0.00) | 0 (0.00) | 54.75 | 11 (0.26) | 7 (0.16) |
| | 2 | 28.00 | 7 (0.25) | 0 (0.00) | 83.75 | 15 (0.23) | 5 (0.08) |
| | 3 | 18.00 | 1 (0.06) | 0 (0.00) | 31.75 | 1.5 (0.06) | 1 (0.04) |
| | 4 | 144.50 | 65 (0.45) | 6 (0.04) | 107.25 | 13.5 (0.16) | 7 (0.08) |
| | 5 | 0.00 | -- | -- | 8.75 | 0 (0.00) | 0 (0.00) |
| | 6 | 53.00 | 12 (0.23) | 4 (0.08) | 13.25 | 1 (0.10) | 2 (0.19) |
| 合計 | | 249.75 | 85 (0.34) | 10 (0.04) | 321.25 | 42 (0.17) | 23 (0.09) |

( ): バグ発生率（バグ数／人時）。Bチームの発生率は、工程毎にAチームに対するBチームの工数比率で正規化してある。

4．3　ストレスレベルとバグ発生率

ストレスレベルと動的バグ発生率の相関係数を表6に示す。

(f) FD工程では、ストレスレベルとストレ起因バグの発生率は比例する。回帰係数は、Aチームが0.109、Bチームが0.104でほとんど同じである(表7、図5）。

(g) DD工程におけるストレスレベルとストレ起因バグの発生率の相関係数は、両チームとも有意でない。

(h) FD工程におけるストレスレベルと人間特性起因バグについては、両チームともストレスレベルが高くなるにつれて発生バグ数が増加する傾向が見られたが、特に有意な相関はなかった。

(i) DD工程におけるストレスレベルと人間特性起因バグの間には、Bチームでは相関が見られなかったにもかかわらず、Aチームで高い相関が見られた。

表6 ストレスレベルと動的発生バグ数の相関係数

| 工程 | チーム | ストレス起因バグ | 人間特性起因バグ |
|---|---|---|---|
| FD | A | 0.94** | 0.01 |
| | B | 0.96* | 0.58 |
| DD | A | 0.63 | 0.89* |
| | B | -0.22 | 0.09 |

*：99%有意　　**：95%有意

表7　ストレス度とストレス起因バグの回帰分析（FD工程）

| チーム | | 自由度 | 2乗和 | F値 | 寄与率 | 回帰係数 | 切片 |
|---|---|---|---|---|---|---|---|
| A | 回帰 | 1 | 0.278 | 32.6* | 0.891 | 0.109 | -0.048 |
| | 残差 | 4 | 0.034 | | | | |
| B | 回帰 | 1 | 0.095 | 24.6** | 0.925 | 0.104 | -0.050 |
| | 残差 | 2 | 0.008 | | | | |

*：99%有意　　**：95%有意

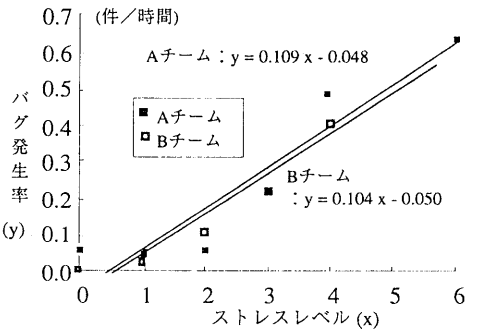


図5　ストレスレベルと動的バグ発生率（FD工程）

図5　ストレスレベルと動的バグ発生率（FD工程）

4．4　考察

（1）FD工程では両方のチーム間で正規化されたバグ発生率に差がないという結果から、両方

の設計方法のパラダイムが類似であると、ある機能を作成するのに必要な基本的な作業量（思考ステップ）も類似のものとなり、たとえ作成するドキュメントが異なっていても、バグは作業量に依存して発生するものと推測される。

一方、DD工程では構造化設計法の方がストレス起因バグが少なかった。Aチームで顕著なストレスレベル6のバグ数を考慮対象外としても、ストレス起因バグはAチームとBチームで73：41となり、Aチームの方が約2倍多い。この理由は、Bチームの方が、構造化された十分なドキュメントが入力として与えられるため、詳細設計がしやすくなり、ストレスの影響も受けにくいものと思われる。

（2）設計工程全体を通してみると、動的エラー要因に起因するバグは、Bチームの方がAチームより少なかった。Bチームに多く見られたストレスレベル6の状態のデータを無視すると、AチームとBチームの動的バグ全体の比率は、74：62となる。すなわち、少なくとも16%、Bチームの方が動的バグが少なかった。したがって、**構造化設計法の方が同程度のストレス環境下においても、機能的設計法より信頼性の高いソフトウェアを開発することができる、と言える。**その主な理由としては、構造化設計法を用いた開発担当者の方が、きちんとした手順で、より理解しやすいドキュメントを設計工程の早期から行うことによるものと思われる。

（3）FD工程における1時間あたりストレスレベルあたりのバグ発生率0.109は、前回の実験結果の一日当たりのバグ発生率0.86と非常に近い値である[8]。このことは、**一般に再現性が低いと言われている人間要因に関しても、このようなコントロールされた実験では、再現性が高いということを思わせる。**

（4）DD工程では、両チームともFD工程ほどストレスとの相関がなかった理由は、DD工程がFD工程より作業がより手順化されているためではないと推測される。特にBチームで、ストレスとの相関がなかった。この理由のひとつとして、DD工程で利用される入力ドキュメントが、FD工程で厳密に記述され、作業がより手順化されていることが考えられる。

（5）表2で示したように、BチームがAチームより多くの時間を設計工程で費やしているけれども、全体の工数はAチームと同等である。今回の実験では、ほとんどのバグは実験者によってDD工程末で摘出されており、その摘出／修正に必要と想定される残存バグの修正工数もそれにしたがって減っている。もし、デザインレビューによるバグ摘出率が両チームで同じであると仮定すると、Aチームの残存バグの方が多いことになり、より多くの工数が後工程で必要になったと思われる。このことは、ソフトウェアを構造的に設計することは、信頼性を向上するだけでなく、工数の削減にも重要な役割を果たすことを示唆している。

## 5．結論

心理的ストレスのかかった環境のもとで構造化設計法と従来設計法の両方で開発実験を行い、設計工程におけるバグ原因、ストレスのかかり具合を収集・分析して、次のことを明らにした。

1）機能設計工程では、機能設計法を用いて開発したAチームと、構造化設計法を用いて開発したBチームの間には、バグ発生率という点で差はなかった。

2）一方、詳細設計工程では、Bチームの方がAチームより心理的ストレスに起因するバグの発生率が低く、同程度のストレスレベルを想定した場合でも約半分であった。

3）動的バグ全体では、同程度のストレスレベルを想定した場合、Bチームの方がAチームより少なくとも16%、発生バグ数が少ない。

これらのことから、心理的ストレスを減らすなど開発環境をよくするだけでなく、より構造化された設計法を用いることが、たとえ過度のストレスのある環境でも信頼性向上に寄与すると言える。

## 参考文献

[1] K. W. Lee, F. A. Tillman and J. J. Higgins: A Literature Survey of the Human Reliability Component in a Man-Machine System, IEEE Trans. Reliability, Vol. R-37, No.1, pp.24-34 (1988).

[2] J. Reason: Human Error, p. 302, Cambridge Univ. Press (1990).

[3] Edited by J. Rasmussen, K Dunkan and J. Leplat: New Technology and Human Error, p. 354, John Wiley & Sons Ltd. (1987).

[4] V. R. Basili and B. T. Perricone: Software Errors and Complexity: An Empirical Investigation, Comm. ACM, Vol. 27, No. 1, pp.

42-52 (1984).
[5] M. Takahashi Y. and Kamayachi: An Empirical Study of a Model for Program Error Prediction, Proc. 8th ICSE, pp. 330-336 (1985).
[6] T. Nakajo and H. Kume: A Case History Analysis of Software Error Cause-effect Relationship, IEEE Trans. Software Eng., Vol. SE-17, No. 8, pp. 830-837 (1991).
[7] T. Furuyama, Y. Arai and K.Iio: Fault Generation Model and Mental Stress Effect Analysis, J. Systems and Software, Vol. 26, No. 1, pp. 31-42 (1994).
[8] T. Furuyama, Y. Arai and K.Iio: Metrics for Measuring the Effect of Mental Stress on Fault Generation during Software Development, Int. J Reliability, Quality and Safety Eng., Vol. 1 No. 2, pp. 257-275 (1994)
[9] K. Takahashi, A. Oka, S. Yamamoto, and S. Isoda,"A Comperative Study of Structured and Text-Oriented Analysis and Design Methodologies," J. Systems and Software, Vol. 28, No. 1, pp. 69-75 (1995).
[10]岡　敦子、山本修一郎、磯田定宏：ソフトウェア開発実験に基づく構造化分析／設計手法の評価、情処論文誌、第34巻、第12号、pp. 2543-2551 (1993)
[11] 藤垣裕子: ソフトウェア技術者の職業性ストレス, p. 177, 労働科学研究所出版部（1992）