B-8　モジュール間の関係に注目したプログラム疲労の概念の拡張に関する提案

A proposal about the extension of program fatigue concept notice to the relations between Modules.

○梶谷　義行**　箭内　直樹**　荒　伸太郎**
Yoshiyuki Kajitani　Naoki Yanai　Shintarou Ara
金子　正人*　武内　惇*　藤本　洋*
Masato Kaneko　Atsushi Takeuchi　Hiroshi Fujimoto

1．はじめに

仕様変更に伴うプログラムの特性の変化を定量的に明確にすることは重要である．その理由は，新しいシステムを設計するとき，新しくプログラムを作成する場合と，既存のプログラムを変更する場合のどちらの方が時間・コストなどの面で効率的であるかの判断が重要である．しかし，この判断を定量的に行う科学的な方法がないため，プログラムの変更作業の工数や開発期間が増大する問題が起こる．

このため，「プログラム疲労」という考えを導入し，判断するための仕組み，および尺度を作る．[1]
しかし，昨年までの研究で「プログラム疲労」は関数内の制御構造の変化といった考慮範囲の狭い疲労モデルであった．

この課題を解決するため，「プログラム疲労」を見直した．本稿では実世界での疲労モデルとして「金属の疲労」の疲労の考え方を見直し拡張した．すなわち，「モジュール間構造」，「ひずみ」，「不純物」，「素性」という視点を加えて「プログラムの疲労」の考え方を拡張しその課題について述べる．

2．疲労の考え方

2．1　金属疲労モデル

「プログラムの疲労」を実世界にある「金属の疲労」に対応させて考えた．さらに実世界にある「金属の疲労」を例に疲労の特性を考えた．

金属は繰り返し外部からの力を受けることで形が変形する．変形したところに金属内部に生じる力が集中する．この金属が金属内部に生じる力に耐えることができなくなるとき金属が破壊されてしまう．これが金属の疲労していく過程である．すなわち，「金属の疲労」とは「金属の形が変形したところに金属内部に生じる力が集中する現象」のことである．[2]

金属の疲労の原因は「ひずみ」の蓄積と空孔の膨張である．ひずみの蓄積は，外部からの力を受けると金属にできる微小なひび割れが蓄積し，金属の強度が低下してしまうため疲労が起こる．

空孔の膨張は金属内部に空気などの不純物があると，外部からの力や，温度変化で膨張し，金属粒子を破壊することで金属自体をもろくしてしまう．

*日本大学工学部
**日本大学大学院工学研究科

また，金属は金属強度などの素材の特性によって変形の仕方に差異が生じる．この変形の仕方による差異は金属の疲労に大きく関係してくるため，金属疲労を考える場合は金属の素材の特性を考慮する必要がある．

「金属自体の素材の特性」を「素性」，「金属に出来る微小なひび割れが発生する現象」を「ひずみ」，「金属内部の空孔」を「不純物」と呼ぶことにした．この金属の「素性」と金属の疲労の原因である「ひずみ」の蓄積，「不純物」の拡張が金属疲労の特性を示すものである．

2．2　金属疲労と「プログラム疲労」との対応付け

プログラムの疲労となる原因を考え金属の疲労原因との対応付けを行った．金属の疲労モデルと「プログラム疲労」の対応付けを表１に示す．

表１．金属疲労と「プログラム疲労」の対応付け

金属疲労	プログラム疲労
外部からの力	プログラムの変更作業
金属の変形	変更されたプログラム
金属の破壊	プログラムの変更ができない
金属の「素性」	プログラムが持っている本来の処理の難しさ
金属の「ひずみ」	変更作業による，プログラムの組み立ての悪い場所※
金属の「不純物」	プログラムで論理的に実行，評価されないコードが変更作業により増える

※組み立てとはプログラムの手続きの構造，式の構造のことをいう．

表１に示す用語を用いて，プログラムの疲労していく過程を説明する．変更作業によってプログラムの組み立ての悪い場所や，プログラムで論理的に実行や評価されないコードが発生する．さらに，その変更作業の繰り返しにより，プログラムの組み立ての悪い場所やプログラムで論理的に実行，評価されないコードが増大し，プログラムの変更が難しくなる．これがプログラムが疲労していく過程である．

プログラムの疲労特性となる「プログラムのひずみ」，「プログラム内の不純物」，「プログラムの素性」について述べる．

（1）プログラムのひずみ

プログラムは，モジュール強度，モジュール結合度

といったモジュール同士の間の組み立て，連結，分岐，繰り返しといったプログラムの流れを制御する「制御構造」と個々の文や式を表現する「式構造」を把握することでプログラム全体のことを示したことになる．
変更作業により，「モジュール間構造」，「制御構造」，「式構造」が変化し，プログラムの質が変化することを「プログラムのひずみ」とした．

（2）プログラムの不純物
プログラム内で実行されないコードおよび論理的に評価されないコードを「プログラムの不純物」とした．

（3）プログラムの素性
プログラムがはじめから持っている処理の難しさを「プログラムの素性」とした．「プログラムの素性」の違いはプログラムを変更したときに受ける悪い影響の大きさ（プログラムのひずみやすさ，不純物の入りやすさ）が違う．
プログラムの素性はプログラムを作成した時点で，はじめから持っているプログラム特性ということができる．

3．プログラム疲労の測定の考え方

疲労の測定方法のモデルを図１に示す．図１で縦軸をプログラムが持っている特性，横軸を版数とする．最初に作られたプログラムが持っているプログラムの特性の大きさを「素性」とし，改版したときにプログラムの「ひずみ」と「不純物」が増減することで，プログラム特性が変化する．この変化量を疲労した量「疲労の変化量」とした．その変化量を測ることでプログラムの疲労を測定する．

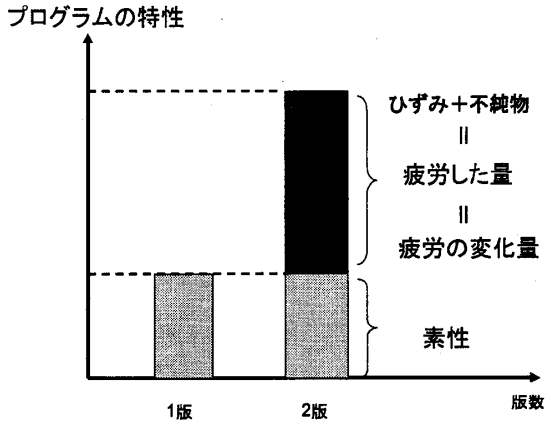

図１．疲労の測定方法

4．プログラム疲労の原因

プログラム疲労の原因を「ひずみ」，「不純物」に分類した．分類に従って C 言語での「プログラム疲労」になる原因を抽出した．抽出した結果を表２に示す．

表２．「プログラム疲労」になる原因項目

疲労の分類		疲労の原因
ひずみ	モジュール間構造	モジュール強度が低くなること（機能的強度，情報的強度，連絡的強度，手順的強度，時間的強度，論理的強度）
		モジュール結合度が強くなること（データ結合，スタンプ結合，制御結合，外部結合，共通結合，内容結合）
	制御構造	ネストが深くなること
		ｉｆ文の数が多くなること
		ｇｏｔｏ文の数が多くなること
		再帰関数の数が多くなること
		完結されていないｉｆ－ｅｌｓｅ文が多くなること
		ｓｗｉｔｃｈ－ｃａｓｅ文の数が多くなること
		フラグの数が多くなること（フラグの追加，または多用）
	式構造	外部と内部の変数名が同じものが多くなること
		１行に複数の記述文が書かれることが多くなること
		ｉｆ文の中の条件式が多くなること（＆＆や｜｜が多い）
		ｉｆ，ｗｈｉｌｅ，ｆｏｒ文の本文が括弧で囲まれていないこと
		マジックナンバーの数が多いこと
不純物		キャスティングされていない演算が多くなること
		不要なコードが残っている数が多いこと

5．終わりに

今回は「ひずみ」，「不純物」，「素性」の分類を行い「プログラム疲労」の概念を見直した．また，モジュール間の構造を考え「プログラム疲労」の拡張を行った．「プログラム疲労」は静的解析ツール[3]を使用して測定が行えるという見通しを得ている．今後はプログラム疲労を測定するための尺度を決定し，「疲労の変化量」を定量的に測るために尺度の重み付けを検討していくことにより，プログラム疲労の評価法の確立を目指す．

参考文献
[1] 滝ほか：“Ｃソースプログラムの疲労度の一考察”，情報処理学会第６０回全国大会，2P－05，3，2000
[2]Charlie R. Brooks：“金属の疲労と破壊”，内田老鶴圃，1999
[3]㈱東陽テクニカ：ＱＡＣ，Ｃソースコード静的解析ツール，2000