ソフトウェアプロセスの自己改善は自学自習でも可能なのか？

梅田政信
九州工業大学
umerin@ci.kyutech.ac.jp

片峯恵一
九州工業大学
katamine@ci.kyutech.ac.jp

荒木俊輔
九州工業大学
araki@ci.kyutech.ac.jp

橋本正明
九州工業大学
masaaki.hashimoto@kyudai.jp

日下部茂
長崎県立大学
kusakabe@sun.ac.jp

要旨

パーソナルソフトウェアプロセス (PSP) は，ソフトウェア技術者のための自己改善プロセスである．*PSP for Engineers* コースは，*SEI* が提供する *PSP* トレーニングコースの一つであり，受講者は，講義と演習とを通じて，高品質ソフトウェアの開発に必要なスキルを習得できる．同コースの講義資料等は，*2018* 年 *10* 月より *Creative Commons* ライセンスに基づき入手可能となり，*PSP* を自学自習する環境は整ってきた．しかし，改善提案の根拠となるプロセスデータの正確性と精密性とを自己で判断することは必ずしも容易ではなく，自学自習によるソフトウェアプロセスの自己改善の有効性は明らかになっていない．本論文では，九州工業大学における *PSP for Engineers* コースの *2013* 年～*2018* 年の実施結果に基づいて，*PSP* インストラクタが果たしている役割を定量的に分析し，自学自習によるソフトウェアプロセスの自己改善の課題を明らかにする．

1. はじめに

情報通信技術の発展に伴い，情報システムの重要性は増すばかりであり，ソフトウェアの品質は，ビジネス上の成功や安心・安全な社会を実現する上で不可欠な要素となっている．そのため，ソフトウェアの品質や開発生産性の向上を目的として，形式手法やモデル駆動開発等の様々な手法やツールが提案されている [1]．しかし，どのような手法やツールを用いたとしても，ソフトウェアの品質を最終的に担保するのはソフトウェア技術者であり，ソフトウェアの品質改善にソフトウェア技術者のスキル向上が不可欠なことに変わりはない．

パーソナルソフトウェアプロセス (PSP) [2, 3] は，米国カーネギーメロン大学ソフトウェアエンジニアリング研究所 (SEI) の Watts S. Humphrey により開発されたソフトウェア技術者のための自己改善プロセスである．PSP for Engineers コースは，企業の実務経験者を主な対象として，SEI が提供する PSP トレーニングコースの一つである．受講者は，PSP インストラクタによる講義とプログラム開発の演習，および自己分析レポートを通じて，高品質ソフトウェア開発に必要なスキルを習得できる．

従来，同コースに関する講義資料等は，SEI とのパートナー契約に基づいて利用可能な他，一定の条件下で Web サイト [4] からも一部入手可能であった．しかし，2018 年 10 月より Creative Commons ライセンス [5] の基に SEI Digital Library [6] から無償で入手可能となっている．そのため，ここより入手した講義資料と市販の書籍等とを組合せて PSP を自学自習する環境が整ったとも言えるが，これらの資料等に基づいて演習を行うだけでは，自己改善のスキルを習得することは必ずしも容易ではない．実際，自学自習により改善効果が得られないことから，PSP の有効性に疑問を呈する意見もある [7]．

PSP for Engineers コースにおいて，PSP インストラクタは，講義により単に知識を伝達するだけでなく，演習結果に対する指導や助言を通じて，効率的で効果的なソフトウェアプロセスの改善を促すことが期待されてい

る．しかし，ソフトウェアプロセスの自己改善において，PSP インストラクタが果たしている役割を定量的に分析，評価した例は知られていない．そこで本論文では，九州工業大学の大学院生を対象とした PSP for Engineers コースの実施結果に基づいて，PSP インストラクタが果たしている役割を定量的に分析し，自学自習によるソフトウェアプロセスの自己改善の課題を明らかにする．

以下では，まず 2 節で PSP for Engineers コースの概要を述べ，3 節で九州工業大学の大学院教育に導入した PSP コースとそれによるプロセス改善結果について述べる．次に，4 節で PSP コースの実施結果に基づいて PSP インストラクタの役割を定量的に分析し，5 節で自学自習によるソフトウェアプロセスの自己改善の課題について考察する．

2. PSP for Engineers コースの概要

2.1. PSP for Engineers コースの構成

ソフトウェアの品質は，それを構成する最低品質の部品によって決まり，各部品の品質は，それを開発した個人とその時に用いたプロセスの品質によって決まる．そのため，ソフトウェアの品質改善には，個人のスキル改善が不可欠である．

PSP は，ソフトウェア技術者のための自己改善プロセスである．図 1 は，PSP におけるプロセスの発展，および PSP とチームソフトウェアプロセス (TSP)[8, 9] との関係を図示したものである．同図中，PSP0，PSP0.1 においては，欠陥記録，時間記録，規模測定，改善提案，および，これらを確実に実施できる規律の重要性を学ぶ．PSP1，PSP1.1 においては，要求の実現に必要な部品の同定と，これに基づく規模と時間の見積り，開発計画と進捗追跡を学ぶ．PSP2，PSP2.1 においては，品質計画，設計とコードのレビュー，および設計テンプレートを用いた設計と検証とを学ぶ．

PSP for Engineers コースは，企業の実務経験者を主な対象として，SEI が提供する PSP トレーニングコースの一つであり，PSP-Planning（PSP0～PSP1.1）とそれに続く PSP-Quality（PSP2～PSP2.1）の 2 つのコース（各 5 日間）からなる．各コースの受講者は，半日の講義後に数 100 行程度の小規模なソフトウェアを開発する演習を行う．また，最終日には講義後に自己分析レポートが課される．受講者は，自身が行った演習に関するプロセスデータを定量的に分析し，それに基づいて自己のソフトウェアプロセスに対する改善提案を行い，次の演習に適用する．この過程を継続的に繰り返すことにより，高品質ソフトウェア開発に必要なスキルの習得を目指す．

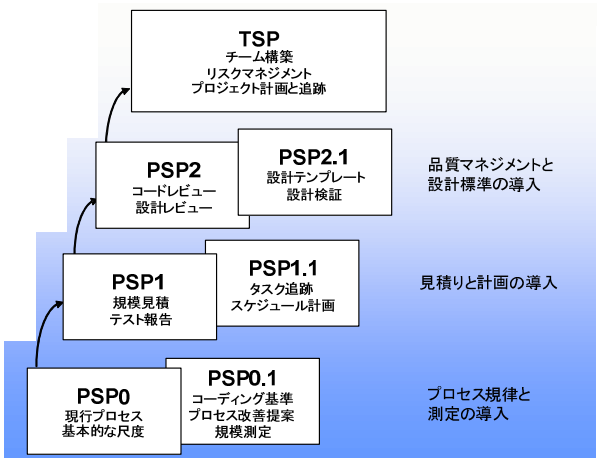


図 1. PSP プロセスの発展

2.2. PSP for Engineers コースの流れ

PSP for Engineers コースを構成する 1 日の講義と演習の大まかな流れを図 2 に UML アクティビティ図として示す．PSP インストラクタは，書籍による事前学習を前提として講義を行い，その中でプログラム開発の演習課題を課す．受講者は，プロセス（PSP0～PSP2.1）に従って，ソフトウェア開発の計画を立案し，計画のレビューを PSP インストラクタに依頼する．PSP インストラクタは，計画の内容を確認し，必要に応じて指導や助言を行う．計画に不備がなければ，受講者は，その計画に基づいてプログラムを開発する．開発が完了すると，受講者は，演習レポートのレビューを PSP インストラクタに依頼する．PSP インストラクタは，演習レポートの内容を確認し，必要に応じて指導や助言を行う．受講者は，必要に応じて演習レポートを修正し，再度のレビューを依頼する．

受講者は，一つの演習課題に着手した時点から演習レポートが受理されるまでの一連の活動の中で，欠陥や時間，規模等に関するプロセスデータを逐次記録する．図 3 と 4 に欠陥記録と時間記録の例を示す．演習中に記録したこれらのプロセスデータは，自己のソフトウェアプロセスに対する改善提案を行うための基礎的なデータであり，もし表 1 に示す欠陥型 [3] の選択誤りや時間記録

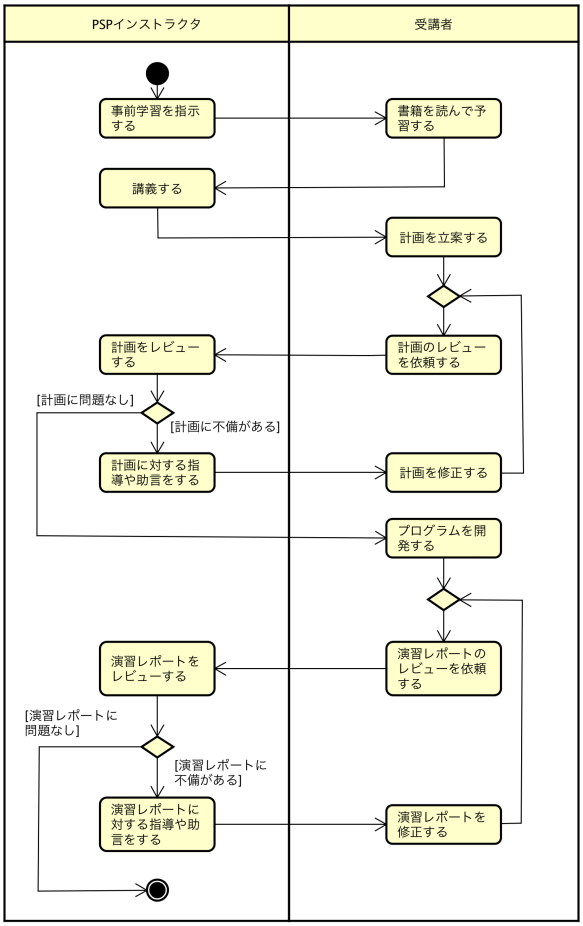
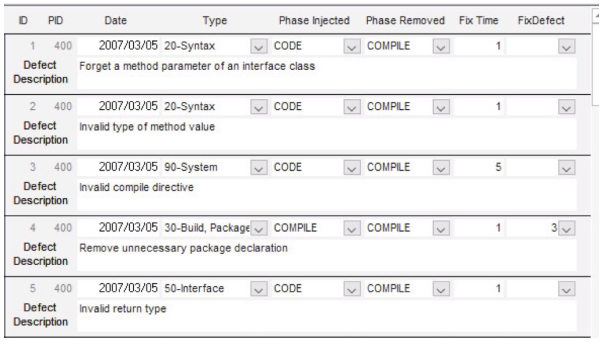
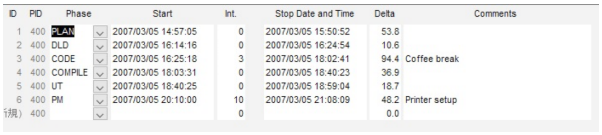


図 2. PSP for Engineers コースにおける講義と演習の流れ



ID	PID	Date	Type	Phase Injected	Phase Removed	Fix Time	FixDefect
1	400	2007/03/05	20-Syntax	CODE	COMPILE	1	
Defect Description	Forget a method parameter of an interface class						
2	400	2007/03/05	20-Syntax	CODE	COMPILE	1	
Defect Description	Invalid type of method value						
3	400	2007/03/05	90-System	CODE	COMPILE	5	
Defect Description	Invalid compile directive						
4	400	2007/03/05	30-Build, Package	COMPILE	COMPILE	1	3
Defect Description	Remove unnecessary package declaration						
5	400	2007/03/05	50-Interface	CODE	COMPILE	1	
Defect Description	Invalid return type						

図 3. 欠陥記録の例



ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments
1	400	PLAN	2007/03/05 14:57:05	0	2007/03/05 15:50:52	53.8	
2	400	DLD	2007/03/05 16:14:16	0	2007/03/05 16:24:54	10.6	
3	400	CODE	2007/03/05 16:25:18	3	2007/03/05 18:02:41	94.4	Coffee break
4	400	COMPILE	2007/03/05 18:03:31	0	2007/03/05 18:40:23	36.9	
5	400	UT	2007/03/05 18:40:25	0	2007/03/05 18:59:04	18.7	
6	400	PM	2007/03/05 20:10:00	10	2007/03/05 21:08:09	48.2	Printer setup
計規)	400			0		0.0	

図 4. 時間記録の例

表 1. PSP の欠陥型標準

番号	名前	説明
10	Documentation	Comments, messages
20	Syntax	Spelling, punctuation, typos, instruction formats
30	Build, Package	Change management, library, version control
40	Assignment	Declaration, duplicate names, scope, limits
50	Interface	Procedure calls and references, I/O, user formats
60	Checking	Error messages, inadequate checks
70	Data	Structure, content
80	Function	Logic, pointers, loops, recursion, computation, function defects
90	System	Configuration, timing, memory
100	Environment	Design, compile, test, or other support system problems

の漏れ等の正確性や精密性を欠くデータが含まれていると，これがソフトウェアプロセスの改善を阻害する恐れがある．そのため，PSP インストラクタは，演習レポートのレビューにおいて，欠陥記録や時間記録，規模記録等がソフトウェアプロセスの改善に支障がないと判断されるまで，繰り返し指導や助言を行う．

このように，PSP インストラクタは，受講者に対して，講義を通じて単に PSP に関する知識を伝達するだけでなく，演習結果に対する指導や助言を通じて，効率的で効果的なソフトウェアプロセスの改善を促すことが期待されている．

3. 九州工業大学におけるソフトウェアプロセスの改善結果

3.1. 九州工業大学の PSP コースの概要

九州工業大学は，文部科学省「先導的 IT スペシャリスト育成推進プログラム」の一環として，2007 年より

SEI と連携し，PSP と TSP とを大学院教育に取り入れ，情報化社会を牽引する高度情報通信技術者の育成に取り組んでいる [10, 11, 12, 13, 14]．この大学院科目群は，PSP for Engineers コースに基づく PSP コースと，教育向けに設計された Introductory TSP (TSPi) に基づく TSP コースとからなる．

PSP コースは，PSP-Planning と PSP-Quality とに対応する二つの演習科目からなり，PSP for Engineers コースと同様の内容を実施している．ただし，受講者が同時期に他の科目を履修しても演習時間を十分確保できるように，1 日分の内容を一週間で実施し，全体を半期で終えるスケジュールとなっている．

PSP コースの運営は，SEI 認定の PSP インストラクタ資格を有する教員が，SEI のライセンスに基づいて行なっている．このため，同コースの修了者には，SEI が実施するコースと同様に，PSP for Engineers コースの修了証が授与される[1]．

3.2. ソフトウェアプロセスの改善結果

PSP コースは，2007 年度から 2018 年度までに，PSP-Planning とそれに続く PSP-Quality とを，それぞれ 111 名とその内の 99 名とが履修し，95 名と 19 名とが修了[2]している．以下では，PSP-Planning を受講した 111 名とそれに続くに PSP-Quality を受講した 99 名とについて，PSP コースを通じてソフトウェアプロセスがどのように改善されたのか，その概略を示す．

3.2.1 規模と時間の見積り誤差

図 5 は，規模の見積り誤差の最小値，平均値，および最大値の推移を表したものである．横軸は演習課題の番号，縦軸は見積り誤差を表す．この図から，当初，過小見積りの傾向が見られるものの，コースの進捗に伴い誤差が小さくなり，過小と過大とにバランス良く見積れるように変化していることが分かる．

同様に，時間の見積り誤差も，図 6 に示すように，過小と過大とにバランス良く見積れるように変化している．

見積り誤差は，小さいに越したことはないが，複数の部品からなるシステム全体を見積る際には，部品の見積り誤差を互いに相殺できるように過小と過大とにバランスすることの方が重要である．

3.2.2 欠陥密度

図 7，8 は，コンパイルとテストにおいて除去した欠陥の密度（1000 行当たりの欠陥数）の推移を四分位で表したものである．横軸は演習課題の番号，縦軸は欠陥密度を表す．これらの図より，コンパイル欠陥密度は，コースの進捗に伴い着実に減少していることが分かる．また，テスト欠陥密度は，一部の例外はあるものの，コースの進捗に伴い減少傾向を示し，課題 8 における第 3 四分位の欠陥密度は，課題 1 における第 1 四分位の欠陥密度と比べて低くなっている．すなわち，コース受講当初に多数の欠陥を作り込んでいた受講者の多くは，コース受講当初の中では優れたとされる品質と比較して，それを超える改善が見られる．このことは，図 9 に示すように，コンパイル前までの欠陥除去率 (プロセスイールド) が最終的に平均で 70 %を超えていることからも確認できる．

図 10，11 は，開発時間に占めるコンパイル時間とテスト時間の割合の推移を四分位で表したものである．コンパイル時間の割合は，コンパイル欠陥密度の減少に伴い低下している．一方，テスト時間の割合は，テスト欠陥密度の減少程に低下しているとは必ずしも言えない．これは，テストにおいて一つの欠陥を除去する時間にばらつきが大きいことが一因である．

これらの結果は，企業の実務経験者を対象に実施された PSP for Engineers コースの結果と大差なく，小規模なプログラミング演習経験しか有しない大学院生であっても，ソフトウェアの品質向上に必要なスキルを習得可能であり，そのツールとして PSP コースが有効なことを示している．

4. PSP インストラクタの役割分析

4.1. 分析方法

2.2 節で述べたように，PSP インストラクタは，計画に対するレビュー時と演習レポートに対するレビュー時との主に二つの場面で，指導や助言の機会がある．ここでは，演習レポートに対するレビューにおいて，その指導や助言の前後で受講者のプロセスデータが変化することに着目し，その変更記録からソフトウェアプロセスの

[1] 2018 年 10 月より Creative Commons ライセンスに移行するため 2018 年度までとなる．

[2] 修了は，四つの演習課題とその後の自己分析レポート課題を全て終えていることを言う．

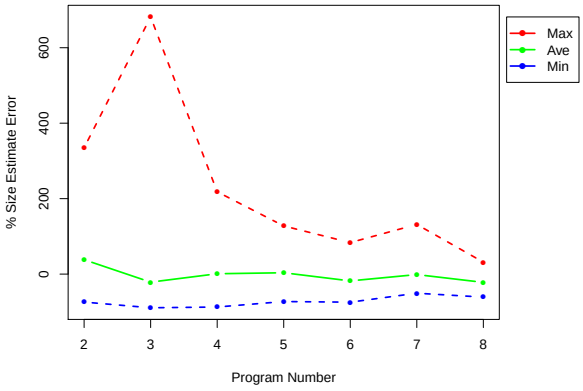


図 5. 規模見積り誤差の推移

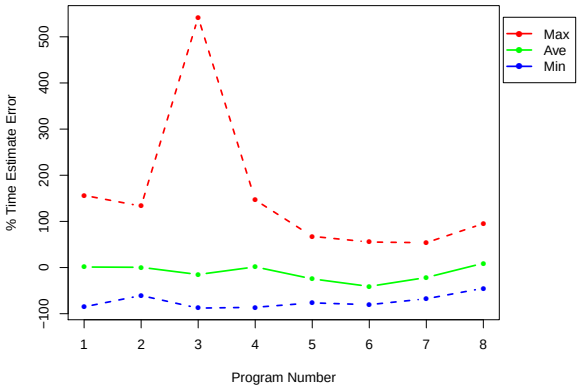


図 6. 時間見積り誤差の推移

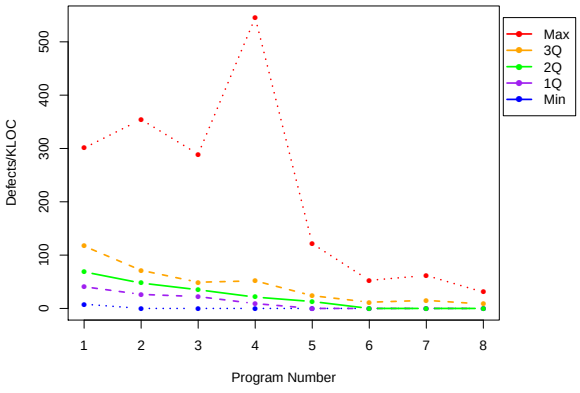


図 7. コンパイル欠陥密度の推移

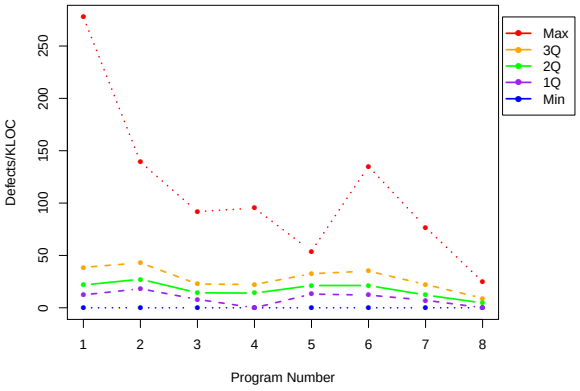


図 8. テスト欠陥密度の推移

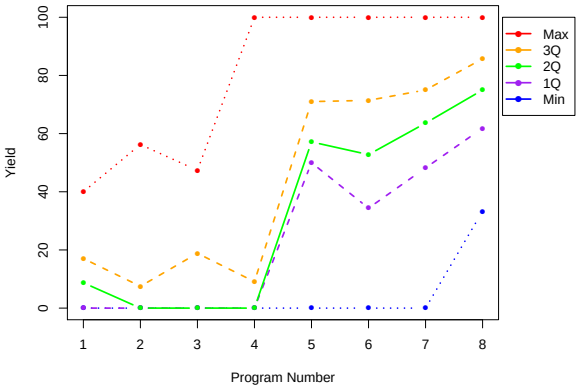


図 9. プロセスイールドの推移

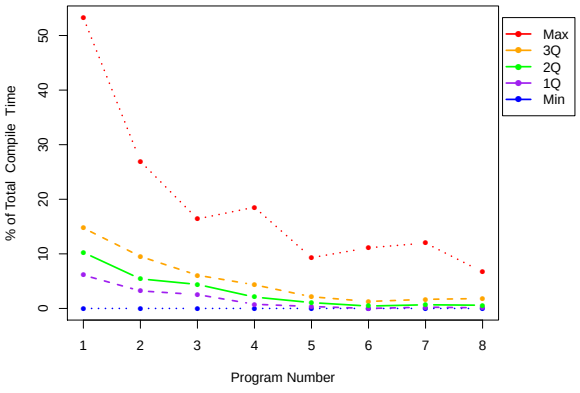


図 10. コンパイル時間割合の推移

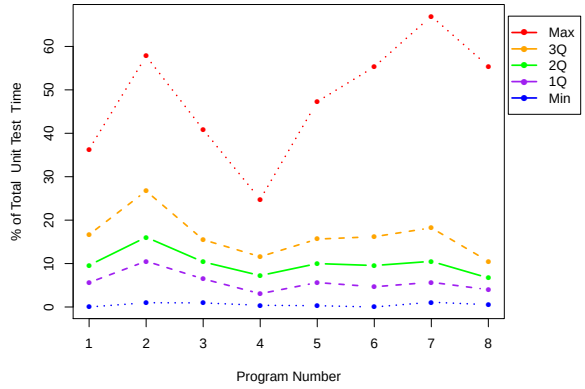


図 11. テスト時間割合の推移

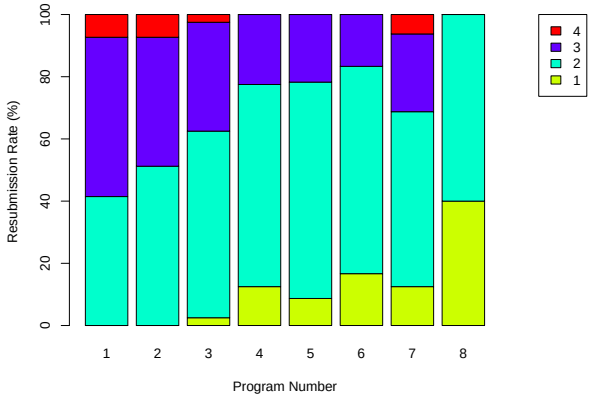


図 12. 演習レポートのレビュー回数の推移

改善に PSP インストラクタが果たしている役割の一部を明らかにする．

分析対象のプロセスデータは，2013 年～2018 年の PSP コース受講者の中で，演習レポートのレビュー依頼時に，その都度プロセスデータの提出を受けた 39 名分である．これらには，レビュー依頼時に間違ったファイルを提出した等の理由により，プロセスデータに不足があるものは含まれていない．

4.2. 分析結果

演習レポートのレビュー回数　図 12 は，一つの演習課題当りの演習レポートのレビュー回数の割合の推移を表したものである．レビュー回数は，最大 4 回[3]であり，コースの進捗に伴い次第に減少している．演習レポートの再提出に至る理由は様々であるが，後述するように，欠陥型の間違いなど，プロセスデータの正確性や精密性に問題がある場合が大半である．このことから，プロセスデータを適切に記録するスキルがコースの進捗に伴い次第に定着していることが確認できる．

欠陥型の修正　図 13 は，欠陥記録において，欠陥型の変更を伴う修正の割合の最小値，平均値，および最大値の推移を表したものである．また，図 14 は修正前の欠陥型別の欠陥数とその累積割合を，図 15 は修正前の欠陥型に対する修正後の欠陥型の型別個数を，それぞれ表したものである．図 13 より，欠陥型の修正は，当初，平均約 35.7%であったものがコースの進捗に伴い減少し，最終

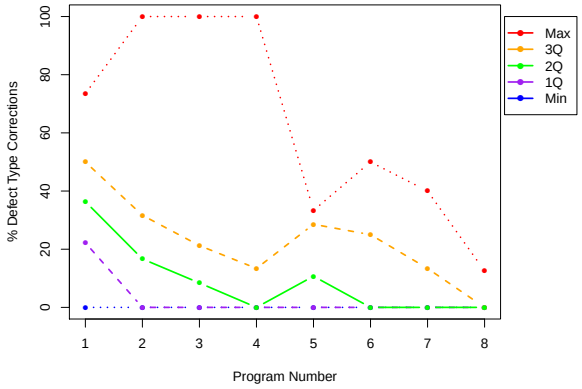


図 13. 欠陥型の修正割合の推移

的に高々12.5%にまで低減している．この結果は，適切な欠陥型を判断するスキルがコースの進捗に伴い定着したことを示している．また，図 14 より，修正前の欠陥型は，20 番 (Syntax) が約 25.7%，70 番 (Data) が 22.8%，40 番号 (Assignment) が約 16.8%を占めており，これらだけで全体の約 65%を占めている．一方，図 15 より，欠陥型 20 番 (Syntax) の誤りの多くは，50 番 (Interface) と 80 番 (Function) であり，たとえば，関数やメソッドの引数の順序やデータ型の誤り等をタイプミス等の構文誤りと取り違えている．

欠陥の埋込フェーズの修正　図 16 は，欠陥記録において，欠陥の埋込フェーズの変更を伴う修正の割合の最小値，平均値，および最大値の推移を表したものである．また，図 17 は，修正前の埋込フェーズに対する修正後の埋込フェーズのフェーズ別個数を表したものである．図

[3]演習レポートを最大 3 回再提出していることになる．

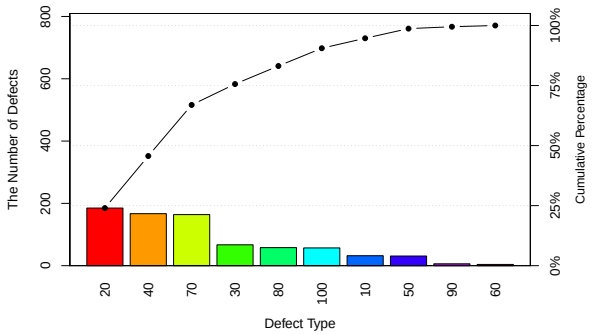


図 14. 修正前の欠陥型の割合

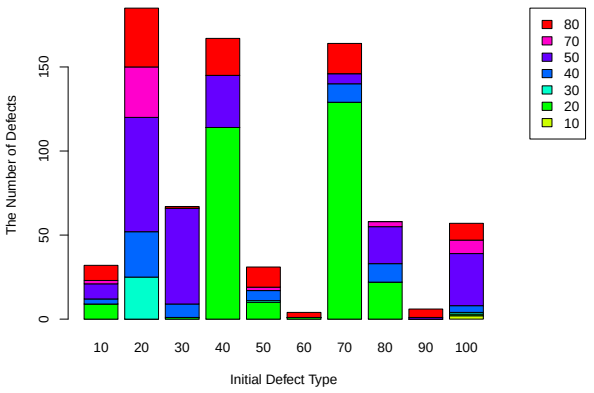


図 15. 修正前の欠陥型に対する修正後の欠陥型

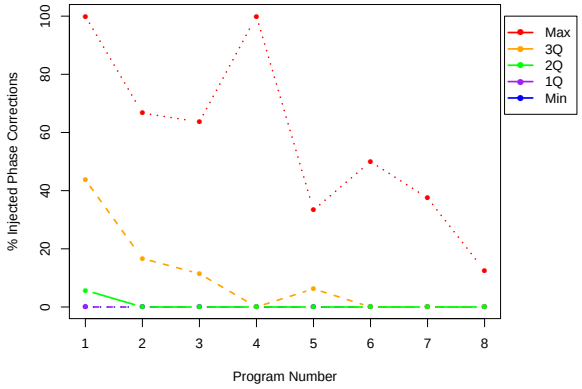


図 16. 埋込フェーズの修正割合の推移

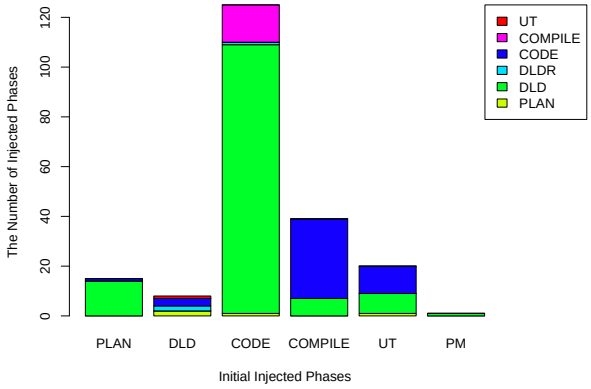


図 17. 修正前の埋込フェーズに対する修正後の埋込フェーズ

16 より，埋込フェーズの修正は，当初，平均約 20.3%であったものがコースの進捗に伴い減少し，最終的に高々 12.5%にまで低減している．欠陥型の場合と同様に，当初，欠陥の埋込フェーズを誤って判断する場合が少なからずあり，適切な埋込フェーズを判断するスキルがコースの進捗に伴い定着したことを示している．また，図 17 より，欠陥を CODE(コーディング) フェーズで埋め込んだとする誤りが最も多く，その内の 85.5%は DLD(詳細設計) フェーズとすべきものである．

欠陥の除去フェーズの修正　図 18 は，欠陥記録において，欠陥の除去フェーズの変更を伴う修正の割合の最小値，平均値，および最大値の推移を表したものである．また，図 19 は，修正前の除去フェーズに対する修正後

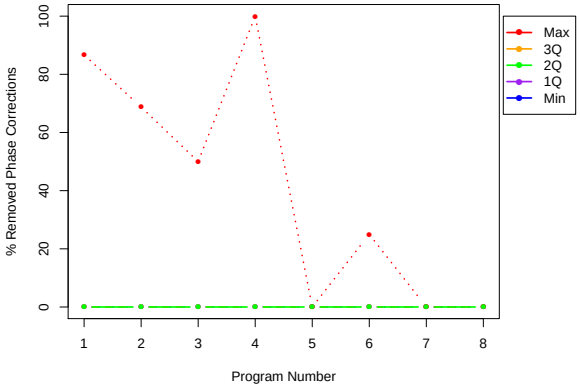


図 18. 除去フェーズの修正割合の推移

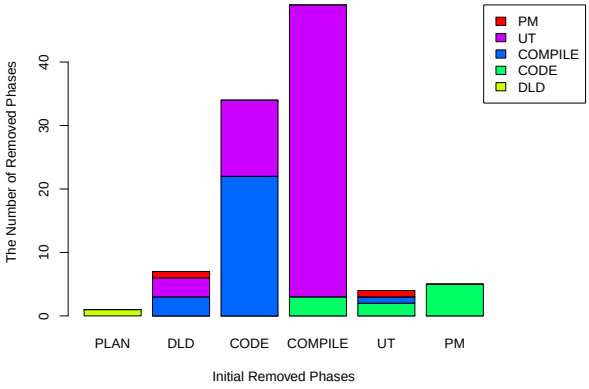


図 19. 修正前の除去フェーズに対する修正後の除去フェーズ

の除去フェーズのフェーズ別個数を表したものである．図 18 より，除去フェーズの修正は，埋込フェーズ程に多くはないものの，一定数あり，適切な除去フェーズを判断するスキルがコースの進捗に伴い定着したことが分かる．また，図 19 より，欠陥を COMPILE(コンパイル) と CODE(コーディング) フェーズで除去したとする誤りが最も多く，それぞれ UT(単体テスト) フェーズと COMPILE(コンパイル) フェーズとすべきものである．これらの誤りの多くは，フェーズと作業内容（行為）との混同[4]である．

[4]たとえば，UT(単体テスト) フェーズにおいて，欠陥を除去するためにプログラムを修正し，コンパイルしても，欠陥を除去したフェーズは CODE や COMPILE ではなく UT である．

5. 自学自習によるソフトウェアプロセス自己改善の課題

5.1. 自学自習による自己改善の限界

欠陥型は，欠陥の埋込に対する対策や欠陥の除去方法を考案する上で重要な手掛りとなる．たとえば，欠陥型の 20 番 (Syntax) に対しては，変数のタイプミスを防止するために，変数一覧を用意し，そこからコピー＆ペーストする等の対策が考えられる．一方，50 番 (Interface) に対しては，関数引数の順序やデータ型の間違いを上流工程で除去するために，設計レビュー用のチェックリストを更新する等の対策が考えられる．このように，欠陥の防止や除去のための対策が欠陥型により異なるため，欠陥型の誤りは，自己改善の阻害要因となる可能性が高い．

一方，欠陥の埋込フェーズは，欠陥の埋込に対する対策をどのフェーズで行うべきか，また，欠陥の除去フェーズは，欠陥の除去をどのように行うべきか，またそもそも欠陥の埋込はどのように防ぐべきだったのかを，それぞれ判断する上で重要な手掛かりとなる．欠陥の埋込に対する対策は，たとえば，設計時とコーディング時とでは一般的に異なる．このため，欠陥型と同様に，埋込フェーズや除去フェーズの誤りは，自己改善の阻害要因となる可能性が高い．

このように，欠陥型や欠陥の埋込フェーズ等のプロセスデータは，自己改善の手がかりとなるものであり，正確性や精密性を欠けば自己改善に支障を来す恐れがある．しかし，4.2 節の分析結果は，プロセスデータの正確性や精密性を受講者が単独で判断することが容易ではないこと，また，プロセスデータの妥当性を判断するスキルが PSP インストラクタからの指導や助言を通じて向上し定着していることを示している．これらの結果は，ソフトウェアプロセスの自己改善を PSP インストラクター等の第三者の介入なく自学自習のみで行うことの限界を示している．

5.2. PSP 教育コースと PSP インストラクタの品質保証

PSP の自学自習は，Creative Commons ライセンスに移行した講義資料等と市販の書籍とを用いて可能になった．そのため，自学自習によりソフトウェアプロセスの自己改善に取り組む事は，これまで以上に容易である．しかし，前述したように，PSP インストラクタによる指

導や助言は，自己改善に重要な役割を果しており，自学自習のみによる自己改善には限界があることも明らかとなった．したがって，効率的で効果的な自己改善には，PSP for Engineers コースと PSP インストラクタによる指導や助言とが不可欠である．

一方，Creative Commons ライセンスへの移行に伴い，SEI による PSP インストラクタの認定制度も終了した．このため，PSP for Engineers コースの内容や実施方法，PSP インストラクタの役割等も自由化された．しかし，効率的で効果的な自己改善には，PSP for Engineers コースと同等な PSP 教育コースとそれを実施する PSP インストラクタの品質保証は欠かせない．この品質保証の枠組み作りは，PSP/TSP コミュニティで取り組むべき今後の大きな課題である．

6. おわりに

本論文では，PSP for Engineers コースに基づく九州工業大学の PSP コースを対象として，ソフトウェアプロセスの改善結果を示し，次に，その改善結果を導いた PSP インストラクタの役割をプロセスデータの変更記録に基づいて分析した．その結果，欠陥型や欠陥の埋込フェーズ等のプロセスデータの正確性と精密性とを自己で適切に判断することは必ずしも容易ではなく，ソフトウェアプロセスの自己改善に，PSP インストラクタによる指導や助言が重要な役割を果していることを定量的に明らかにした．

PSP/TSP に関連する講義資料等が Creative Commons ライセンスに移行したことに伴い，Team Process Community of Practice (TP CoP) [15] において PSP/TSP の今後の展開について議論されている．関係各位の積極的な参加を期待したい．また，俊敏さと高品質とを兼ね備えたソフトウェア開発手法の重要性は益々高まっており，今後は，TP CoP とも連携しながら，PSP/TSP に Agility を取り入れた手法の開発に取り組む必要がある [16, 17]．

また，PSP コースは，演習が途中で停滞してコースを修了できない受講者が一定数あり，修了率は 20%〜30% 程度に止まっている．新しいスキルが定着するまでコースを継続できるかどうかは，受講者の動機づけが深く関わっていると考えられるため，動機づけに着目した教育手法の改善も今後の重要な課題である [18, 19, 20, 21, 22, 23]．

謝辞

九州工業大学への PSP/TSP 関連科目の導入は，文部科学省「先導的 IT スペシャリスト育成推進プログラム」の支援によるものである．PSP/TSP 関連科目の導入を主導された秋山義博客員教授，ならびに日頃よりご支援頂く関係各位に感謝申し上げます．

参考文献

[1] 栗田太郎：モバイル FeLiCa のソフトウェア開発における品質確保のための構造と実践-抽象度の制御やコミュニケーションの活性化に向けて-, 情報処理学会デジタルプラクティス, Vol. 1, No. 3, pp. 148–157 (2010).

[2] Humphrey, W. S.: *A Discipline for Software Engineering*, Addison-Wesley (1995), (邦訳:松本 正雄 監訳, ソフトウェア品質経営研究会訳:『パーソナルソフトウェアプロセス技法』, 共立出版, 1999 年).

[3] Humphrey, W. S.: *A Self-Improvement Process for Software Engineers*, Addison-Wesley (2005), (邦訳:秋山 義博 監訳, JASPIC TSP 研究会訳:『PSP ガイドブック ソフトウェアエンジニア自己改善』, 翔泳社, 2007 年).

[4] Software Engineering Institute, : PSP Academic Material, http://www.sei.cmu.edu/tsp/tools/academic (2013).

[5] Creative Commons, : クリエイティブ・コモンズ 表示 4.0 国際パブリック・ライセンス, https://creativecommons.org/licenses/by/4.0/legalcode.ja (2019).

[6] Software Engineering Institute, : Team Software Process (TSP) and Personal Software Process (PSP) Materials, https://www.sei.cmu.edu/go/tsp (2019).

[7] 海谷治彦, 小島彰, 海尻賢二：大学におけるプロセス改善教育のあり方について –PSP 法実践の経験をもとに–, in *Software Symposium 2001 Proceedings*, pp. 137–142 (2001).

[8] Humphrey, W. S.: *Introduction to the Team Software Process*, Addison-Wesley (1999), (邦訳:秋山義博 監訳, JASPIC TSP 研究会訳:『TSPi ガイドブック』, 翔泳社, 2008 年).

[9] Humphrey, W. S.: *TSP Leading a Development Team*, Addison-Wesley (2005).

[10] 秋山義博, 片峯恵一, 梅田政信, 橋本正明, 乃万司：九州工業大学におけるパーソナルソフトウェアプロセス教育－ソフトウェア品質向上のためのスキル修得－, *SEC Journal*, Vol. 6, No. 3, pp. 118–125 (2010).

[11] Katamine, K., Umeda, M., Hashimoto, M. and Akiyama, Y.: Changing Software Management Culture from Academic, in *TSP Symposium 2011 Proceedings*, pp. 12–18 (2011).

[12] Katamine, K., Umeda, M., Hashimoto, M. and Akiyama, Y.: A STRATEGY IN EFFECTIVE TEACHING OF SOFTWARE ENGINEERING PROCESS FOR GRADUATE STUDENTS, in *The Proceedings of IADIS International Conference Information Systems 2012*, pp. 259–266 (2012).

[13] 梅田政信, 片峯恵一：PSP/TSP による実践的な ICT 人材育成の取り組み, 情報処理学会誌, Vol. 53, No. 10, pp. 1084–1087 (2012).

[14] Umeda, M., Katamine, K., Hashimoto, M. and Akiyama, Y.: Improving Introductory TSP for Creating High Performance Student Teams, in *TSP Symposium 2013* (2013).

[15] TP CoP Working Groups, : TEAM PROCESS COMMUNITY OF PRACTICE, https://www.tpcop.org (2019).

[16] 片峯恵一, 梅田政信, 橋本正明：PSP/TSP を基礎とした軽量なソフトウェア開発手法に関する取り組み, プロジェクトマネジメント学会 2017 年度秋季研究発表大会予稿集, pp. 131–132 (2017).

[17] 片峯恵一, 梅田政信, 荒木俊輔, 橋本正明：TSP の俊敏性向上に関する取り組みと評価, ソフトウェア品質シンポジウム 2017 (SQiP 2017), pp. B3–2 (p.6) (2017).

[18] Ishibashi, K., Hashimoto, M., Umeda, M., Katamine, K., Yoshida, T. and Akiyama, Y.: A Preliminary Study on Formalization of Motivation Process in Personal Software Process Course, in *The Proceedings of the 10th Joint Conference on Knowledge-Based Software Engineering*, pp. 128–137 (2012).

[19] Umeda, M., Katamine, K., Ishibashi, K., Hashimoto, M. and Yoshida, T.: Motivation Process Formalization and its Application to Education Improvement for the Personal Software Process Course, *IEICE Transactions on Information and Systems*, Vol. E97-D, No. 5, pp. 1127–1138 (2014).

[20] 田頭薫, 片峯恵一, 梅田政信, 石橋慶一, 橋本正明：動機づけプロセスの状態遷移モデルを用いた PSP コース受講生の分析, プロジェクトマネジメント学会 2016 年度秋季研究発表大会予稿集, pp. 247–251 (2016).

[21] 日下部茂, 梅田政信, 片峯恵一, 石橋慶一：ソフトウェアプロセス教育向け動機づけモデルをシステム理論に基づく STAMP/STPA により効果的に活用する手法の提案, プロジェクトマネジメント学会 2017 年度秋季研究発表大会予稿集, pp. 301–306 (2017).

[22] Kusakabe, S., Umeda, M., Katamine, K. and Ishibashi, K.: Managing Personal Software Process Education Course Based on Motivation Process Model by Using System-Theoretic Method STAMP/STPA, in *Proceedings of the 11th International Conference on Project Management (ProMAC2017)*, pp. 1066–1072 (2017).

[23] 日下部茂, 梅田政信, 片峯恵一, 石橋慶一：システム理論に基づくモデリングと質的研究を併用したソフトウェアプロセス教育の動機づけシナリオ改善, ソフトウェア・シンポジウム 2018 論文集, pp. 100–107 (2018).