東京学芸大学紀要　自然科学系　69: 301－309，2017

研究室におけるコードの知識消失防止のためのドキュメント記載項目の提案

倉　田　和香菜*1・櫨　山　淳　雄*2

情報科学分野

（2017年5月29日受理）

KURATA, W. and HAZEYAMA, A.: Proposal of Document Description Item to Prevent Knowledge Loss of Code in Laboratory. Bull. Tokyo Gakugei Univ. Div. Nat. Sci., 69: 301–309. (2017) ISSN 1880–4330

Abstract

Currently, various systems are operated, and they become complicated by addition and modification of functionality. In addition, due to changes in persons in charge, there are many situations where people who didn't participate in the development work on maintenance work. A laboratory in a university is an example as a high liquidity organization of human resources.

It is said that the task with the highest time cost among the entire process of software maintenance is to read and understand the code. However, in the maintenance process, since information on concepts and relationships considered by developers is rarely recorded, maintenance personnel are reading documents to infer concepts and relationships. In addition, in the laboratory there is a situation that students with less maintenance experiences have to maintain the system developed by beginners of software development. Programs created by beginners have problems for example unique coding styles and unnecessary descriptions, which are considered to be more time-consuming. In addition, because a laboratory has high human resources liquidity, it is difficult to ask developers directly about system concepts and relationships, which leads to the loss of knowledge.

It is therefore important to keep knowledge as a document to prevent the loss of knowledge. However, there is a problem that it is not clear what should be left in the document.

In this research, we propose items which should be described in the document on code. By utilizing documents that satisfy the proposed items, we make it possible to prevent knowledge loss of code in the laboratory. We also conducted an experiment on the time cost of the maintenance process and the validity of the items proposed. From the results, we found some items that do not appear in other literature.

Keywords: software maintenance, documentation

Department of Information Sciences, Tokyo Gakugei University, 4-1-1 Nukuikita-machi, Koganei-shi, Tokyo 184-8501, Japan

要旨： 現在，様々なシステムが運用され，仕様変更，追加修正によりシステムが複雑化している。また，担当者の変更などにより，開発時に関わらなかった人が保守作業を行う場面も多い。このような場面が多く現れる人材流動性の高い組織として大学の研究室がある。

＊1　東京学芸大学2016年度卒業生
＊2　東京学芸大学 技術・情報科学講座 情報科学分野（184-8501　小金井市貫井北町4-1-1）

ソフトウェア保守の全工程の中で最も時間的コストの高い作業は，コードを読み理解することであると言われている。しかし，保守工程では開発者が考慮した概念や関係の情報がほとんど残されていないために保守者がドキュメントを読んで概念や関係を推測することが行われている。また，研究室ではソフトウェア開発初学者が開発したシステムを保守経験の少ない学生が保守しなければならないという状況がある。初学者の作成したプログラムは，コーディングスタイルが独特であることや，不要な記述があるといった問題もあり，さらに時間的コストがかかると考える。加えて，研究室は人材流動性が高いので，開発者にシステムの概念や関係を直接質問することが難しく，これは知識が消失することに繋がる。そこで知識の消失を防ぐために，ドキュメントとして知識を留める。しかし，ドキュメントに何を残すべきか不明確という問題がある。

本研究では，コードについてのドキュメントに記載すべき項目を提案する。提案項目を満たすドキュメントの活用により，研究室におけるコードの知識消失防止を可能にする。保守工程の時間的コスト，ドキュメント記載項目の妥当性に関する実験を行った。実験の結果，ソフトウェア開発時に残すべき項目を提案する他の文献やツールには現れないドキュメント記載項目を発見することができた。

1．はじめに

本節では，本研究の背景と目的について述べる。

1．1　背景

現在，様々なシステムが運用され，仕様変更，追加修正によりシステムが複雑化している［1］。また，担当者の変更などにより，開発時に関わらなかった人が保守作業を行う場面も多い［1］。このような場面が多く現れる人材流動性の高い組織として大学の研究室がある。研究室という組織では，毎年度学生の出入りがあり，何年も同じ人材が滞在するということは稀である。つまり，研究室で運用されているシステムは開発に関わらなかった保守経験の少ない学生が保守していかなければならない。

ソフトウェア開発全体における総コストの50～90%が保守に費やされている［2］。そして保守の全工程の中で最も時間的コストの高い作業は，コードを読み理解することであると言われている［3］。しかし，保守工程では開発者がドキュメント作成時に考慮した概念や関係の情報がほとんど残されていないために保守者がドキュメントを読んで概念や関係を推測すること［4］が行われている。また，仕様を理解する上で必要なドキュメントはそろっていないことが多く，システムのソースプログラムを直接参照して，システムの仕様を分析しなければならない場合が多い［1］。これらは時間的コストがかかる要因と言えることは明らかである。また，研究室ではソフトウェア開発初学者が開発したシステムを保守経験の少ない学生が保守しなければならないという状況がある。初学者の作成したプログラムは，コーディングスタイルが独特であることや，不要な記述があるといった問題もある［5］。この要因からさらに時間的コストがかかると考える。

加えて，研究室は人材流動性が高いので，開発者にシステムの概念や関係を直接質問することが難しく，これは知識が消失することに繋がる。知識の消失を防ぐため，ドキュメントとして知識を留める研究やツールが開発されている。本研究ではV字モデルに基づくソフトウェア開発工程のテストに対応づけられた上流工程の設計書等のドキュメントではなく，クラスやメソッドの説明やメソッドの引数や戻り値の説明［3］といったコードについてのドキュメントを対象とする。加えて，本研究では，ソフトウェアに関するドキュメントの管理を目的としないので，対象ドキュメント以外のドキュメントは存在しない状況を想定する。このドキュメントはソフトウェア保守において最も重要な成果物である［6］。現在，コードについてのドキュメントとして，各プログラミング言語に対応するドキュメンテーションツールが存在する。Java 言語には javadoc［7］というツールが存在する。

保守者にとってドキュメントは重要な手掛かりとなる［6］。しかし，ドキュメントが存在しない，クオリティが低い，役に立たない［8］という実状がある。この理由として，開発者がドキュメントに何を残すべきか不明確ということが考えられる。

1．2　目的

1．1節で述べた背景から，問題を次に挙げる。

保守に関する問題

ドキュメントが残されていないことで，保守の時間的コストが増大，知識が消失する。

研究室に関する問題

開発者のコーディングスタイルが独特で，不要な記述が多く［5］，コード理解に時間的コストがかかる。開発者がドキュメントとして何を残すべきかわからない。

研究室に関する問題に対しては，開発者が残すべきドキュメント記載項目を提案する。そして，ドキュメントが残ることによって，開発者の独特なコーディングスタイルの理解を支援することができ，コード理解が容易になると考える。このことが，保守に関する問題の解決にも繋がると考える。

ドキュメント記載項目は，一般的なドキュメントで扱われる項目や関連研究，事前調査から抽出する。抽出されたドキュメント記載項目を用いて提案ドキュメントを作成し，実験を行う。実験から提案ドキュメントが保守工程の時間的コストへ与える影響とドキュメント記載項目が十分であるかについて評価を行う。

本研究では，保守工程の時間的コストを削減することを目的とした研究室におけるコードに関する知識消失防止のためのドキュメント記載項目を提案する。

2．ドキュメント記載項目の抽出

本節では，ドキュメント記載項目の抽出を行う。2．1節でソフトウェア保守に関する規格に表記されているドキュメント記載項目やドキュメントツールで用いられる項目について，2．2節でドキュメント記載項目を抽出することができる関連研究について，2．3節で事前調査について述べる。

2．1　一般的なドキュメント記載項目

ソフトウェア保守に関する規格に表記されているドキュメント記載項目やドキュメントツールで用いられる項目について述べる。

2．1．1　ソフトウェアライフサイクルプロセス―保守

ISO/IEC 14764を基に作成された日本工業規格であるソフトウェアライフサイクルプロセス―保守［9］において，ソフトウェアの引き継ぎを行う際の提言が記述されている。ソースコードを引き継ぐ際は，「未解決又は延期された問題報告及び新規要求事項」という情報を引き継ぐべきと提言している。

この規格から未解決，新規要求事項という項目が必要だと考える。

2．1．2　javadoc

ドキュメントを作成するツールとして，各プログラミング言語に対応するドキュメントツールが存在する。Java言語にはjavadoc［7］が存在する。

javadocはHTML形式のドキュメントを生成することができる。加えて，javadocコメントとして説明を記述することができ，javadocに用意されているタグを用いて説明文に追加することができる。タグとその説明を表1に示す。

表1　javadocのタグと説明

タグ	説明
@author	著者（作成者）を示す。
{@code}	不等号を記述する際に用いる。
{@docRoot}	生成されたドキュメントのルートディレクトリを基点とする相対パスを示す。
@deprecated	非推奨であることを示す。
@exception	例外を示す。
{@inheritDoc}	ドキュメントを継承する際に用いる。
{@link}	参照リンクを追加する際に用いる。
{@linkplain}	{@link}と同義。
{@literal}	{@code}と同義。
@param	引数に説明を追加する際に用いる。
@return	戻り値を示す。
@see	関連項目とのリンクを追加する際に用いる。
@serial	デフォルトの直列化可能フィールドのdocコメントで用いる。
@serialData	データの型と順序を直列化形式でドキュメント化する。
@serialField	SerializableクラスのObjectStreamFieldコンポーネントをドキュメント化する。
@since	導入された際のバージョン（改版数）を示す。
@throws	@exceptionと同義。
{@value}	定数値を示す。
@version	バージョン（改版数）を示す。

javadocからタグで表している項目が必要だと考える。

２．２　関連研究

ドキュメント記載項目を抽出することに関わる研究について述べる。

２．２．１　ソフトウェア開発初学者によるコーディングの分析に関する研究

若林らの研究［5］では，Javaプログラミング初学者のプログラムに見られる問題点を事例から分析した。分析の結果，ソフトウェア開発初学者はインデントや変数の命名規則に統一性がないことが多く，可読性を下げる要因とされている。

この研究から，命名規則という項目が必要だと考える。

２．２．２　ソフトウェア保守の際に利用する成果物に関する研究

一般的にソフトウェア開発に関する成果物はソフトウェア保守を手助けする重要な手掛かりだとされている。しかし，ソフトウェア保守においてすべての成果物が必要となるのかという疑問により，必要となる成果物の調査と分析をSergio Cozzetti B. de Souzaらの研究［6］では行った。

調査として以下の2種類が行われた。

・保守者が必要と考える成果物についての調査：ソースコードが最も重要な成果物，次いでコメント，論理データモデルが重要ということを示した。
・保守者が実際に利用している成果物の調査：ソースコードが最も重要な成果物，次いでコメント，単体テスト，ユースケース記述，非機能要求のプロトタイプが重要ということを示した。

Sergio Cozzetti B. de Souzaらは２つの調査の結果を基にソフトウェアの保守を行う際にソースコードとコメントは最も重要な成果物と述べた。そして，残りの重要な成果物としてデータモデルと非機能要求のプロトタイプを挙げた。一方で，全体把握を行う際に利用し，何度も確認しない設計モデルやシステムの全体図は重要でないと分析した。

この研究からデータモデルが重要ということが分かり，このことからデータの動きを理解するために関数の戻り値，引数という項目が必要だと考える。

２．２．３　ソフトウェア保守のコード理解で行うアプローチに関する研究

Claudia Szaboの研究［10］では，大学２年生のソフトウェア開発初学者に約１万１千行のソフトウェアの保守を行ってもらい，その際に学生がどのようなアプローチでコード理解を行ったかを分析した。その結果，学生のコード理解のアプローチはトップダウンとボトムアップに分類することができたと述べている。

トップダウンのアプローチのとっかかりとして，パッケージの構造を把握，パッケージとクラスの名前から役割の推測，クラスやオブジェクトの関係を記したUMLを作成するものが見られた。ボトムアップのアプローチのとっかかりとして，クラスの目的を書き留める，データ構造によって何が起きているのかを読み取るものが見られた。

この研究から，各パッケージの位置付け，パッケージとクラスの役割と目的，開発者が構造を考える上で基となった図，データ構造（関数の戻り値，引数，例外）という項目が必要だと考える。

２．２．４　本研究の位置づけ

若林らの研究では，ソフトウェア開発初学者のコーディングを分析した。分析結果から，ソフトウェア開発初学者のプログラムからはコーディングスタイルが独特であることや，不要な記述があるといった問題が見つかった。

本研究では，ソフトウェア開発初学者のコーディングについての問題をドキュメントで対応することが必要であると考える。

Sergio Cozzetti B. de Souzaらの研究では，ソフトウェア保守の際に実際に利用する成果物の調査と分析を行った。結果から，ソフトウェア保守を行う際にソースコードとコメントは最も重要な成果物と述べた。一方で，全体把握を行う際に利用し，何度も確認しない設計モデルやシステムの全体図は重要でないと分析した。

本研究では，保守者が実際に利用している成果物を反映したドキュメント記載項目を提案し，コメントとして利用することが必要であると考える。

Claudia Szaboの研究では，ソフトウェア開発初学者がソフトウェア保守を行う際に，どのようにコード理解を行うのかを分析した。分析結果から，コード理解のアプローチはトップダウンとボトムアップに分類することができた。

本研究では，どちらのアプローチにも対応できるドキュメント記載項目を提案し，円滑にソフトウェア保守を行えるようにする必要があると考える。

2．3　事前調査

保守者が必要となる情報について知ることを目的とし，本研究室で利用されている「ゼミ資料管理システム」の保守を筆者が行った。「ゼミ資料管理システム」とは本研究室の卒業生が学部3年次に開発を行ったシステムである。このシステムにはドキュメントが残っておらず，コードのみを理解して保守を行った。保守内容はゼミ資料を登録する際の日付入力をドロップダウンリストからカレンダーに変更するというものである。保守を行う際につまずいたことは2点ある。1点目はファイルの名称だけでは何を行っているファイルなのかを特定することは困難だったことである。2点目は変数が多く使われており，その変数が何を表しているのかが分からなかったことである。その結果，手を加える，または削除してもよい変数かどうかを読み取ることに時間が掛かった。

この事前調査から命名規則，変数の内容と目的を記録に残すことが必要と分かり，開発者自身が考慮した命名規則や変数の意味，データの受け渡し方という項目が必要だと考える。

3．ドキュメント記載項目の検討と説明

2節からドキュメントに記載すべき項目として以下の7項目を抽出した（対応を表2に示す。○がそれぞれの文献，調査から抽出されたことを表す）。項目の説明を次に述べる。

① 命名規則：ファイル名，変数名，関数名，クラス名の命名規則
② 各パッケージの位置づけ：システム内のパッケージの役割
③ クラスの役割，目的：クラスの説明
④ 変数の内容，目的：変数の説明
⑤ 関数の引数，戻り値，例外：関数のデータの受け渡し方の説明
⑥ 開発者が構造を考える上で基となった図：開発する際に記述したメモ等
⑦ 非推奨，未解決，新規要求事項：延期された問題がある部分，未解決の部分，今後導入したいことの説明

抽出した項目をドキュメント記載項目とするかの検討を行う。開発初学者を対象とした文献［5］と文献［10］の研究の項目は必要とする。また，どのような項目が適しているのか，より多くの項目を検討するために，2つ以上必要と判断した項目については必要とする。これらの条件から，すべての項目を必要と判断する。

表2　ドキュメント記載項目の対応表

項目	文献［9］	文献［7］	文献［5］	文献［6］	文献［10］	事前調査
①			○			○
②					○	
③					○	
④		{@value}				○
⑤		@exception			○	
		@param		○	○	○
		@return		○	○	○
⑥					○	
⑦	○	@deprecated				

4．評価実験

本節では，評価実験について述べる。

4．1　実験目的

本実験の目的は，次の2つの観点から評価を行うことである。

・保守工程の時間的コストの評価
・保守工程を行う上でドキュメント記載項目が十分であるかの評価

4．2　実験対象者

本大学の「情報システム設計」を履修済みの学生を対象とする。「情報システム設計」とは，次をねらいと目標とするJava言語でのソフトウェア開発が含まれている授業である。「ソフトウェアシステムの開発の流れ（ソフトウェアプロセス，ソフトウェア開発過程）と，開発過程の各工程で行う作業について学ぶ。とりわけオブジェクト指向に基づくモデリング（UML）とJava言語と関係データベースを用いたWebアプリケーション開発に重点を置く。本科目の受講により，ソフトウェア工学に関する基本的な用語が説明できること，UMLで記述されたモデル図の読み書きならびにJava技術を用いた基本的なWebアプリケーションが開発できることを目指す［11］」

4．3　実験方法

実験の手順，保守ルール，保守完了条件について述べる。

４．３．１　手順

実験の手順を次に示す。

1）学生をグループXとグループYに人数と経験（システムの開発数）が均等になるように分ける。

2）グループXに属する学生には提案ドキュメントとソースコードを与える。グループYに属する学生にはソースコードのみを与える。

3）グループX，グループYそれぞれに属する学生に同じ課題を与え，個人でソフトウェア保守を行ってもらう。

４．３．２　保守ルール

実験を行う際の保守ルールを次に示す。

・他の機能が使えなくならないようにすること。
・データベースの構成は変更しない（新規テーブルの作成，カラムの追加等は行えない）。
・実験に参加している他の人に聞いてはいけない。
・インターネットでの検索可（ただし，インターネットを介して誰かに聞いてはならない）。
・制限時間は２時間。

４．３．３　保守完了条件

本実験における保守完了条件を次に示す。

・課題を満たしている。
・保守ルールが守られている。

４．４　評価方法

実験の評価方法について，４．１節で述べた実験の目的に沿って述べる。

４．４．１　保守工程の時間的コストの評価

ドキュメントを利用した場合と利用しなかった場合，それぞれの課題の終了時間で評価を行う。

４．４．２　保守工程を行う上でのドキュメント記載項目が十分であるかの評価

実験後にアンケートに回答してもらい，アンケート結果から評価を行う。アンケート内容はグループX，グループYそれぞれ次に示す。

グループX

・本ドキュメントの必要性
（不要１，２，３，４必要）（選択）
・利用しなかったドキュメント項目（複数選択）
・その他に必要と感じたドキュメント項目（記述）

グループY

・ドキュメントの必要性
（不要１，２，３，４必要）（選択）
・ソースコード以外で欲しかった情報（記述）

４．５　実験課題

実験の課題と保守対象のシステムについて述べる。

４．５．１　課題内容

授業の成績管理システム（機能は４．５．３で説明する）に関する次の２つの課題についてソフトウェアの保守を行ってもらう。

課題（1）授業の評定をA，B，C，D，FからS，A，B，C，Fに変更。

課題（2）評定の変更に伴う，GPAの算出法の変更。その際，2016年11月15日より前に評定付与が行われているものは変更前の評定，2016年11月15日以降のものは変更後の評定で算出する。

４．５．２　課題の妥当性

４．５．１節で述べた課題内容の妥当性について述べる。

課題（1）では，該当する変更箇所を発見することができれば満たせる課題である。

課題（2）では，日付による場合分けを行うことが必要となる。成績を取得し，GPAを算出している箇所を発見し，日付の場合分けができれば満たせる課題である。

これらから，ドキュメント記載項目③④⑤を利用することで指定箇所を発見することができる。したがって，この課題は妥当と考える。

４．５．３　課題システムの機能

課題のシステムである「成績管理システム」について述べる。システム名は「SyllabusRecorder」である。約２千行のシステムで，筆頭著者が開発したシステムである。

成績管理システムの機能とその説明を次に示す。

・ログイン機能：あらかじめ登録してあるIDとパスワードが，入力したIDとパスワードと一致したときのみ本システムを利用することができる。
・ログアウト機能：ログイン状態を解除する。
・授業開設機能：授業情報（授業名，単位数）を登録し，授業を開設することができる。
・授業削除機能：開設されている授業を削除することができる。
・授業編集機能：開設されている授業情報（授業名，単位数）を編集することができる。

・履修登録機能：開設されている授業の履修登録をすることができる。
・履修取り消し機能：履修登録を行った授業の履修を取り消すことができる。
・成績登録機能：開設されている授業を履修している学生の評定（A，B，C，D，F）を登録することができる。
・成績参照機能：登録された成績を参照することができる。
・GPA参照機能：登録されている成績のGPA算出結果を参照することができる。GPAの算出方法は，評定数値×単位数の和を総単位数で割る。

5．実験結果

本節は，評価実験の結果について述べる。

5．1　実験協力者

8名の学生に協力いただいた。実験協力者のうち，5名がチーム開発を1回，個人での開発を1回行っている。残りの3名が個人での開発を2回行っている。実験協力者全員がソフトウェア保守の経験がない学生であった。

実験協力者のグループ分けを次に示す。
グループX：2名（チーム開発を1回，個人での開発を1回），2名（個人での開発を2回）
グループY：3名（チーム開発を1回，個人での開発を1回），1名（個人での開発を2回）

5．2　実験結果

4．5．1節の課題（1），（2）の終了時間を表3に示す（×は制限時間内に課題が達成できなかったことを表す）。

実験後のグループXのアンケート結果を表4に，グループYのアンケート結果を表5に表す。

表3　課題の終了時間

グループX		グループY	
課題（1）	課題（2）	課題（1）	課題（2）
15分	55分	8分	×
18分	×	10分	×
24分	×	15分	98分
32分	×	36分	×

表4　アンケート結果（グループX）

質問内容	回答（点線以下は理由）
(1) 本ドキュメントの必要性 不要（1・2・3・4）必要	4：0名
	3：3名 ・どの引数や変数が何を表しているか日本語でまとまって見られたのでわかりやすかった。 ・どのファイルを変更すれば良いか考えることができたから。 ・ドキュメントのおかげで，それぞれのクラスがどういう役割をしているかがわかり，直す場所が見つけやすかった。
	2：1名 ・ファイル名がServlet-____ではなくて，そのままの_____Servlet.javaがよかった。 ・変数名は独特の名前だったので参考になった。 ・コードを見てコード通りに進めば目的のファイルにたどりつけた。
	1：0名
(2) 本ドキュメントで利用しなかった項目	①4名　②3名　③2名　④2名 ⑤2名　⑥2名　⑦3名
(3) その他で必要と感じた項目，情報	・beans内のそれぞれの変数名の意味 ・jspのそれぞれの内容 ・セッションのやり方 ・SQL文の文法

表5　アンケート結果（グループY）

質問内容	回答（点線以下は理由）
(1) ドキュメントの必要性 不要（1・2・3・4）必要	4：2名
	・初め，どこに何が定義されているかわからなかったので，探す時間と手間がかかった。 ・わからないコードもあったので，コードの説明や変数の説明等の説明を付け加えて欲しかった。
	3：1名
	・ハウツーのスキルが圧倒的に少ない。
	2：0名
	1：1名
	・ブラウザのデベロッパーツールを見ればどのファイルが読み込まれているかわかるから。
(2) 欲しかった情報	・日付比較のメソッド。 ・SQL文で何を取り出しているかわからなかった。 ・どういった意図でコードを書いたのか。 ・変数についての説明。

6．考察

4．1節で述べた実験の目的に加え，実験の内容についての考察を述べる。

6．1　時間的コストの評価

課題（1）は全ての実験協力者が達成することができたが，課題（2）を達成することができた実験協力者が8名中2名という数値のため，課題（1）のみについて評価を行う。

課題（1）の終了時間の平均はグループXが22.25分，グループYが17.25分であった。この結果から，時間的コストに関するドキュメントの有効性は現れなかった。グループXのアンケート結果の「ファイル名がServlet-____ではなくて，そのままの_____Servlet.javaがよかった」というドキュメントの記載の方法に関する意見から，ドキュメントが見づらいということが要因の1つとして挙げられる。

この要因から見やすいドキュメントを提供することで保守の時間的コストが下がるのではないかと考える。

6．2　ドキュメント記載項目の評価

グループYのアンケート結果からドキュメントの必要性を，必要（3以上）とした回答が4名中3名から得られた。また，グループXのアンケート結果から本ドキュメントの必要性を，必要（3以上）とした回答が4名中3名から得られた。この結果から，ドキュメントは必要であることが明らかになった。このことは，保守対象であるコードに関連する部分のドキュメントを参照していたことからも裏付けられる。

ドキュメント記載項目に関しては，グループXのアンケートから，ドキュメント記載項目①②⑦を利用しなかったという回答が4名中3名以上から得られた。この理由は，保守について理解していないことが考えられる。これはグループYのアンケートの欲しい情報として，開発時に記されることはない情報（今回の実験では，保守課題となっている「日付比較のメソッド」）を回答していることから伺える。

グループXのアンケートからドキュメント記載項目③④⑤⑥を4人中2人が利用したことが分かる。加えて，グループYのアンケートで，ドキュメント記載項目④が欲しかった情報という回答を得られた。ドキュメント項目④は文献［7］と事前調査から抽出された項目であり，他の文献にはない。他人が作成したコードを理解することに慣れていない保守経験の少ない学生ならではの意見と考える。

ドキュメント記載項目の他に欲しかった情報として，「SQL文が何を表しているか」という意見がグループX，グループYともに得られた。複雑なSQL文に触れる機会が多くない学生ならではの意見と考えられる。

ドキュメント記載項目①②⑦については，保守の意義を説明した後に実験を行うこと等再検討を行う必要があると考える。ドキュメント記載項目③④⑤⑥は必要である。加えて，「SQL文の説明」が新たなドキュメント記載項目として挙げられた。

６．３　実験内容の評価

今回の実験では，課題（2）を達成することができた実験協力者が８名中２名という低い結果になってしまった。この原因として，制限時間を設けなければならなかったことが挙げられる。インターネットでの検索を可としたので，情報収集を行うことができる。このことから，時間が有限でなければ達成できた課題だと考える。

課題（2）を達成することができた実験協力者を比較すると，グループYの実験協力者より，グループXの実験協力者の方が，40分以上短い時間で完了させている。課題（2）の終了時間のデータを多く集められたら，６．１節の評価も変わった可能性がある。

７．おわりに

本節では，本研究のまとめと今後の課題について述べる。

７．１　まとめ

本研究では，研究室におけるコードの知識消失防止のためのドキュメント記載項目を提案した。また，提案ドキュメントによる保守工程の時間的コストへの影響とドキュメント記載項目が十分であるかを明らかにするための実験を行った。時間的コストへの影響を確認することはできなかったが，４名中３名から提案ドキュメントが必要という回答があり，提案ドキュメントの必要性を確認することができた。ドキュメント記載項目に関しては，ドキュメント記載項目③④⑤⑥は必要，ドキュメント記載項目①②⑦は再検討を行う必要があるという結果となった。加えて，「SQL文の説明」という新たな項目を発見した。

７．２　今後の課題

本研究では，保守者に対するドキュメント記載項目を検討した。今後は，更なるドキュメント記載項目の改善と，見やすいドキュメントの提案を行う。加えて，ドキュメントを残す者に対する支援を行い，保守工程の時間的コストを削減できるようにしていきたい。

謝辞

評価実験にご協力いただいた東京学芸大学の学生の皆さんに心から感謝申し上げます。

参考文献

［1］五味俊明，上村学，吉野利明，松尾昭彦，伊藤栄信：保守ドキュメント自動生成システム「SysSurf」−ドキュメント化手法−，情報科学技術フォーラム一般講演論文集，Vol. 3（1），pp. 199−200，2004.

［2］大森隆行，丸山勝久，林晋平，沢田篤史：ソフトウェアの進化研究の分類と動向，コンピュータソフトウェア，Vol. 29，No. 3，pp. 3−28，2012.

［3］佐々木唯，肥後芳樹，楠本真二：プログラム文並び替えに基づくソースコードの可読性向上の試み，ソフトウェアエンジニアリングシンポジウム2013，pp. 1−6，2013.

［4］安達久人，小林吉純，太田理：概念構造を利用したドキュメント保守支援方式，電子情報通信学会技術研究報告知能ソフトウェア工学，Vol. 94，No. 130，pp. 31−36，1994.

［5］若林智徳，松浦佐江子：Javaプログラミング初学者のクラス構成学習のための依存関係解析，電子情報通信学会技術研究報告知能ソフトウェア工学，Vol. 101，No. 386，pp. 55−60，2011.

［6］Sergio Cozzetti B. de Souza, Nicolas Anquetil, Kathia M. de Oliveira: A Study of the Documentation Essential to Software Maintenance, Proceedings of the 23rd International Conference on Design of Communication: Documenting & Designing for Pervasive Information, ACM, pp. 68−75, 2015.

［7］javadoc - Java APIドキュメントジェネレータ：https://docs.oracle.com/javase/jp/1.5.0/tooldocs/windows/javadoc.html（参照日：2017/01/21）.

［8］Qian Hu: Software Documentation Writing on the Project of Software Upgrade and Maintenance, Proceedings of the International Conference on Computer Science and Information Processing (CSIP), pp. 1400−1403, 2012.

［9］JIS X 0161:2008 ソフトウェア技術 − ソフトウェアライフサイクルプロセス − 保守：http://kikakurui.com/x0/X0161-2008-01.html（参照日：2017/01/21）.

［10］Claudia Szabo: Novice Code Understanding Strategies During a Software Maintenance Assignment, Proceedings of the 37th International Conference on Software Engineering, IEEE Press, Vol. 2, pp. 276−284, 2015.

［11］東京学芸大学 授業シラバス：http://portal.u-gakugei.ac.jp/syllabus/（参照日：2017/01/21）.