社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
SS2005-13 (2005-06)

協調フィルタリングに基づくソフトウェア開発技術の推薦

秋永 知宏† 大杉 直樹† 柿元 健† 角田 雅照† 門田 暁人† 松本 健一†

†奈良先端科学技術大学院大学情報科学研究科 〒630-0192 奈良県生駒市高山町 8916-5

E-mail: †{tomohi-a, naoki-o, takesi-k, masate-t, akito-m, matumoto}@is.naist.jp

あらまし 近年，数多くのソフトウェア開発技術が提案されている．ソフトウェア開発技術者が，これらの技術すべてを習得することは困難であるため，あらかじめ習得すべき技術の選択を行う必要がある．そこで，協調フィルタリングに基づき，各技術者にとって有用だと思われるソフトウェア開発技術を推薦する方法を提案する．提案方法では，まず各技術者に対し，各開発技術に対する興味の度合いを調べる．そして協調フィルタリングに基づいて興味の傾向が類似した技術者を発見する．類似した技術者が高い評価をしており，かつユーザが知らない技術を，有用な技術として推薦を行う．提案方法を 4 つの基準（絶対誤差，適合率，再現率，F1 値）により評価したところ，提案方法の精度が，単に平均値を推薦するよりも，絶対誤差で 0.16，F 値で 0.09 良い結果が得られた．

キーワード 推薦アルゴリズム，アイテムベース，ユーザベース，類似度，技術者，開発技術，絶対誤差，F1 値

Software Technology Recommendation Based on Collaborative Filtering

Tomohiro AKINAGA† Naoki OHSUGI† Takeshi KAKIMOTO†

Masateru TSUNODA† Akito MONDEN† and Ken'ichi MATSUMOTO†

† Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, 630-0192 Japan

E-mail: †{ tomohi-a, naoki-o, takesi-k, masate-t, akito-m, matumoto }@is.naist.jp

Abstract In recent years, much software development technology is proposed. It is difficult for the software engineer to be master of all these technologies. So it is necessary to select the technology that should acquire it beforehand. Then, we propose a system recommending the software exploitation technology that seems to be useful for engineer by using Collaborative Filtering. In the proposal method, first of all, the interest to each development technology is first investigated to each engineer. And, the engineer to whom the tendency to the interest is similar is discovered based on cooperated filtering. "A similar engineer is doing a high evaluation" and "technology that user doesn't know" so recommends it as useful skills. Four standards (an absolute error, precision, recall, F1 value) estimated the proposal method. As a result, it was better than the accuracy of the proposal technique recommended a simple mean value (absolute error is 0.1, and F1 value is 0.09).

Keyword Recommendation Algorithm, Item-based Collaborative Filtering, User-based Collaborative Filtering, Engineer, Development Technology, Absolute Error, F1 Value

1. まえがき

ソフトウェア開発に関する技術（以降では，開発技術，あるいは単に技術と略記する）は増加を続けている．米 Thomson Corporation の報告によると，2002 年度現在，ソフトウェアに関する登録特許は世界全体で 470 万件に及んでいる[12]．さらに，プロセス改善モデルとして著名なソフトウェア成熟度モデル（SW-CMM）[9]など，特許として登録されていないものまで含めると，非常に多くの技術が存在している．

ソフトウェアを開発する技術者（以降では，開発技術者，あるいは単に技術者と略記する）は，これら開発技術の中から，自らが関わる開発業務に適したものを取捨選択して適用しなければならない．これら技術の多くは生産性や品質の改善を目的として考案されたものであるが，技術を適用する状況に応じてその効果は大きく異なる．例えば，SW-CMM に代表される計画駆動型手法と eXtreme Programming（XP）[1]に代表されるアジャイル開発手法は，共に生産性や品質改善を目的とし，規範となる開発方法や手順を規定している．しかし，前者は大規模な組織，後者を小規模な組織に適しており，適用する技術を誤ると改善効果が得られないだけでなく，生産性や品質が低下する場合さえある[2]．

さらに開発技術者は，これら技術の中から，より容易に習得できるものを取捨選択して適用しなければならない．開発技術者は，開発業務に非常に多くの時間を費やしており，技術の習得に利用できる時間は限られている．経済産業省による 2004 年度の調査によると，

	t_1	t_2	…	t_j	…	t_b	…	t_n
u_1	$r_{1,1}$	$r_{1,2}$	…	$r_{1,j}$	…	$r_{1,b}$	…	$r_{1,n}$
u_2	$r_{2,1}$	$r_{2,2}$	…	$r_{2,j}$	…	$r_{2,b}$	…	$r_{2,n}$
…	…	…		…		…		…
u_i	$r_{i,1}$	$r_{i,2}$	…	$r_{i,j}$	…	$r_{i,b}$	…	$r_{i,n}$
…	…	…		…		…		…
u_a	$r_{a,1}$	$r_{a,2}$	…	$r_{a,j}$	…		…	$r_{a,n}$
…	…	…		…		…		…
u_m	$r_{m,1}$	$r_{m,2}$	…	$r_{m,j}$	…	$r_{m,b}$	…	$r_{m,n}$

図 1. CF に用いる m 行 n 列の表データ

組み込みソフトウェアを開発する技術者は，月平均160 時間以上を業務に費やしているのに対し，教育に利用できる時間は年平均で 200 時間以下である[6]. 限られた時間の中で,効率よく技術を習得するためには，現在までに習得した技術と関連が深く，既に習得した知識を最大限に利用できる技術を選択することが望ましい.

しかし，非常に多くの技術の中から，技術者が自らの開発業務に適し，かつ，容易に習得できる開発技術を取捨選択することは容易ではない. 技術が業務に適するか，あるいは，これまでに習得した技術との関連が深いかを判断するためには，技術者自身が各技術について学習し，その内容に関してある程度の知識を得なければならない. 当該技術が技術者の業務に適さない，あるいは，技術者にとって習得が困難である場合には，それまでの学習に費やした時間が無駄になる可能性がある. 限られた学習時間の中で，効率よく技術を習得するためには，技術の取捨選択を支援する方法が必要である.

この問題を解決するため，本論文では，個々の技術者のソフトウェア開発業務に適し，かつ，技術者にとって習得が容易だと思われる技術を協調フィルタリング（CF）[4], [8], [11]に基づいて推薦する方法を提案する. CF は，多量に存在するアイテム(書籍，楽曲，映画作品など)の中からユーザの好みに合うと予測されるアイテム情報を選出して推薦するシステムの基盤技術として用いられている. CF に基づく推薦システムでは，「あるアイテムに対して同様の評価をするユーザ同士は，他のアイテムに関しても同様の評価をするであろう」と考え，自分と類似した評価をしているユーザが高い評価をしたアイテムの推薦を行う.

提案方法では，「ソフトウェア技術に関する興味の傾向が似ている技術者同士は，他の技術に関しても同様の興味を示すであろう」と考え，推薦を行う. 推薦を行うために，技術者のソフトウェア技術に対する興味の度合いを調べ，データとしてまとめたものを用いる. まず，推薦対象の技術者を決め，ソフトウェア技術に対する興味の傾向が類似している技術者（類似技術者）を探し出す. そして，類似技術者が大きな興味を持っており，かつ，推薦対象の技術者にとって未知の技術を推薦する.

以降，2章では CF を用いた推薦手順ついて説明し，3章では提案方法における推薦の手順を説明する. 4章では提案方法の有効性を評価する実験について報告し，6章で実験結果対する考察を述べる. 6章では関連研究を紹介し，7章でまとめと今後の課題について述べる.

2. 協調フィルタリング

CF は，多量に存在するアイテム(記事，Web，書籍，楽曲，映画作品など)の中からユーザの好みに合うと予測されるアイテム情報を選出して推薦するシステムの基盤技術として用いられている. CF を用いた推薦システムとして，Resnick[10]らの GroupLens が挙げられる. GroupLens はユーザの興味に合う Usenet ニュース記事を推薦するシステムである. 一般に，CF を用いた推薦は次の手順で行われる.

手順1. 各ユーザは使用したアイテム(書籍・楽曲・映画など)に対する評価を行う.

手順2. 推薦対象のユーザ（対象ユーザ）と評価の傾向が似ているユーザ（類似ユーザ）を探す.

手順3. 対象ユーザが未使用のアイテムに対する評価を，類似ユーザの評価に基づいて予測する.

手順4. 対象ユーザが高く評価すると予測されたアイテムを対象ユーザに推薦する.

Sarwar ら[11]は，アイテム数が非常に多く，ユーザによる評価が全体の 1%以下しか評価が行われていないデータに対して適用可能なユーザの好みを予測できるアルゴリズムを提案した. GroupLens で利用されている手法がユーザベース手法[3]と呼ばれるのに対して，この手法はアイテムベース手法[11]と呼ばれる.

アイテムベース手法では，類似ユーザではなく，評価が似ているアイテム(類似アイテム)を選び出し，類似アイテムの評価を用いて推薦する．類似ユーザと比較して類似アイテムの変動は小さく，類似アイテムのキャッシュが可能であり計算時間の短縮が行えるため，アイテム数が非常に多い場合に有効な手法である．このアルゴリズムは Amazon.com が提供する本推薦システムでも用いられている．

提案方法では，ユーザはソフトウェア開発技術者，アイテムはソフトウェア開発技術として，推薦を行う．「ソフトウェア開発技術に関する興味の傾向が似ている技術者同士は，他の開発技術に関しても同様の興味を示すであろう」と考え，推薦を行う．また，提案方法では，ユーザベース手法とアイテムベース手法の両方を用いる．

3. 提案方法における推薦手順

提案方法では，図 1に示すような m 行 n 列の表データを用いる．図中，$u_i \in \{u_1, u_2, \ldots, u_m\}$は i 人目の技術者を表し，$t_i \in \{t_1, t_2, \ldots, t_n\}$は j 番目の開発技術を表す．また，$r_{i,j} \in \{r_{1,1}, r_{1,2}, \ldots, r_{m,n}\}$は技術者 u_i の開発技術 t_i に対する興味の度合いを表す．興味の度合いは興味がない(1)～大変興味がある(4)の 4 段階で表し，数値が大きいほど興味の度合いが大きいとする．技術者が開発技術を知らない場合，$r_{i,j}$には値を記入しない．

提案方法では，CF のユーザベース手法，並びに，アイテムベース手法，それぞれに基づく 2 種類の手順で推薦を行う．ユーザベース手法に基づいて推薦を行う場合，ソフトウェア開発技術に対する興味の傾向が似ている開発技術者を発見し，次の手順で推薦を行う．

1. 推薦対象の技術者 u_a と，その他の技術者 u_i の間の類似度を次式(1)で計算する．

$$sim(u_a,u_i)=\frac{\sum_{j\in T_a\cap T_i}\left(r_{a,j}-\overline{\overline{t_j}}\right)\times\left(r_{i,j}-\overline{\overline{t_j}}\right)}{\sqrt{\sum_{j\in T_a\cap T_i}\left(r_{a,j}-\overline{\overline{t_j}}\right)^2}\sqrt{\sum_{j\in R_a\cap R_i}\left(r_{i,j}-\overline{\overline{t_j}}\right)^2}} \quad (1)$$

ただし，T_a と T_i はそれぞれ技術者 u_a と u_i が評価した開発技術の集合を表す．また，$\overline{\overline{t_j}}$ は開発技術 t_j に対する評価の中間値を現す．

2. 計算した類似度に基づき，推薦対象の技術者 u_a の開発技術 t_b に対する評価の予測値 $\hat{r}_{a,b}$ を次式(2)で計算する．

$$\hat{r}_{a,b}=\frac{\sum_{i\in k-neighbors}\left(r_{i,b}-\overline{\overline{r}}_{k-neighbors,b}\right)\times sim(u_a,u_i)}{\sum_{i\in k-neighbors}sim(u_a,u_i)}+\overline{\overline{r}}_{k-neighbors,b} \quad (2)$$

ただし，*k-neighbors* は，技術 t_b に対して評価を行っており，かつ，技術者 u_a と類似度が高い k 人の技術者の集合を表す．類似技術者数 k は予測精度に影響を与えるため，評価実験では k を変動させ，最も精度が高くなる k を採用する必要がある．また，$\overline{\overline{r}}_{k-neighbors,b}$ は類似技術者の t_b に対する評価の中間値を表す．

3. 推薦対象技術者 u_a が評価していない開発技術全てについて予測値を計算し，予測値が高い開発技術から順に，当該技術者に対して推薦する．

一方，アイテムベース手法に基づいて推薦を行う場合，予測対象の技術と似た技術者に興味を持たれている技術を発見し，次の手順で推薦を行う．

1. 推薦対象の技術者 u_a が評価していない技術 t_b に対し，技術 t_b とその他の技術 t_j の間の類似度を次式(3)で計算する．

$$sim(t_b,t_j)=\frac{\sum_{i\in U_b\cap U_j}\left(r_{i,b}-\overline{u_i}\right)\times\left(r_{i,j}-\overline{u_i}\right)}{\sqrt{\sum_{i\in U_a\cap U_j}\left(r_{i,b}-\overline{u_i}\right)^2}\sqrt{\sum_{i\in U_b\cap U_j}\left(r_{i,j}-\overline{u_i}\right)^2}} \quad (3)$$

ただし，U_b と U_j はそれぞれ技術 t_b と t_j に対して評価を行った技術者の集合を表す．また，$\overline{u_i}$ は技術者 u_i の評価の平均値を表す．

2. 計算した類似度に基づき，推薦対象の技術者 u_a の開発技術 t_b に対する評価の予測値 $\hat{r}_{a,b}$ を次式(4)で計算する．

$$\hat{r}_{a,b}=\frac{\sum_{j\in k-nearestTechs}\left(r_{a,j}-\overline{t_b}\right)\times sim(t_b,t_j)}{\sum_{j\in k-nearestTechs}sim(t_b,t_j)}+\overline{t_b} \quad (4)$$

ただし，*k-nearestTechs* は，技術者 u_a が評価を行っており，かつ，技術 t_b と類似度が高い k 個の技術の集合を表す．類似技術数 k は予測精度に影響を与えるため，評価実験では k を変動させ，最も精度が高くなる k を採用する必要がある．また，$\overline{t_b}$

表 1. アンケートに記載したソフトウェア開発技術と各技術の評価率

ソフトウェア開発技術	評価率	ソフトウェア開発技術	評価率
エクストリームプログラミング（XP）	84.31%	Pascal	90.20%
スクラム	31.37%	Perl	96.08%
リーン・ソフトウェア開発（LSD）	37.25%	Ruby	84.31%
適応型ソフトウェア開発（ASD）	25.49%	Python	60.78%
クリスタル・ファミリ	15.69%	LISP	84.31%
フィーチャ駆動型開発（FDD）	23.53%	COBOL	92.16%
エクストリーム・モデリング（XM）	23.53%	ファンクション・ポイント法	88.24%
ラショナル統一プロセス（RUP）	70.59%	COCOMO	84.31%
ISO9000	80.39%	COCOMOII	72.55%
ソフトウェア能力成熟度モデル（SW-CMM）	86.27%	AgileCOCOMO	23.53%
能力成熟度モデル統合（CMMI）	88.24%	GoalQuestionMetric（GQM）アプローチ	70.59%
パーソナルソフトウェアプロセス（PSP）	68.63%	ExtensibleMarkupLanguage（XML）	100.00%
チームソフトウェアプロセス（TSP）	56.86%	SOAP	80.39%
統一モデリング言語（UML）	100.00%	Web サービス	98.04%
ピンボック（PMBOK）	68.63%	サービス指向アーキテクチャ	76.47%
シーボック（SWEBOK）	56.86%	構造化設計	94.12%
Java	98.04%	オブジェクト指向設計	100.00%
EnterpriseJavaBeans（EJB）	84.31%	全社的品質管理（TQC）	58.82%
JavaServerPages（JSP）	82.35%	総合的品質管理（TQM）	54.90%
ActiveServerPages（ASP）	66.67%	統計的品質管理（SQC）	50.98%
HypertextPreprocessor（PHP）	76.47%	J2SESDK	76.47%
CommonGatewayInterface（CGI）	86.27%	WindowsAPI	86.27%
C／C＋＋	100.00%	MicrosoftFoundationClasses（MFC）	60.78%
ドットネット（.NET）	94.12%	VisualComponentLibrary（VCL）	25.49%
VisualBasic（VB）	96.08%	Qt	23.53%

はu_a以外の技術者の技術t_bに対する評価の平均値を表す．

3. 推薦対象技術者 u_a が評価していない開発技術全てについて予測値を計算し，予測値が高い開発技術から順に，当該技術者に対して推薦する．

4. 評価実験

提案方法の予測精度を評価するため，実験を行った．以下に実験の詳細について述べる．

4.1. 実験データ

実験では，ソフトウェア開発者 31 名と情報科学を専攻する大学院生 20 名の計 51 名に対してアンケートを行った結果に基づいてデータセットを作成した．アンケートに記載したソフトウェア技術と，各開発技術に対する評価率を表 1に示す．ここで，評価率とは，全 51 名中評価を行った人数の割合を示す．アンケートでは，それぞれの技術者に，ソフトウェア開発技術 51 個に対して，知っているか知らないかを答えてもらい，知っているならばその開発技術に対して，興味の度合いを 4 段階で表現してもらった．各段階は，1 は興味が無い，2 は少し興味がある，3 は興味がある，4 は大変興味があるとした．知っているか知っていないかの判断基準は，その開発技術の概要を説明できるかどうかとした．

4.2. 評価基準

評価基準には，絶対誤差，適合率，再現率，F1 値の 4 つを用いた．これらは，CF による推薦システムの評価基準である[5]．適合率は，推薦結果に含まれる適切な結果の割合を示し，再現率は，全ての適切な結果のうち推薦結果に含まれた割合を示す．F1 値は，適合率と再現率を一つの値で表現する値である．これらは推薦精度を表す評価基準で，各評価基準の値が大きいほど，予測精度が高いことを示す．これらは，以下の式でそれぞれ表される．

絶対誤差:

$$\text{絶対誤差} = |\text{実測値} - \text{予測値}| \quad (3)$$

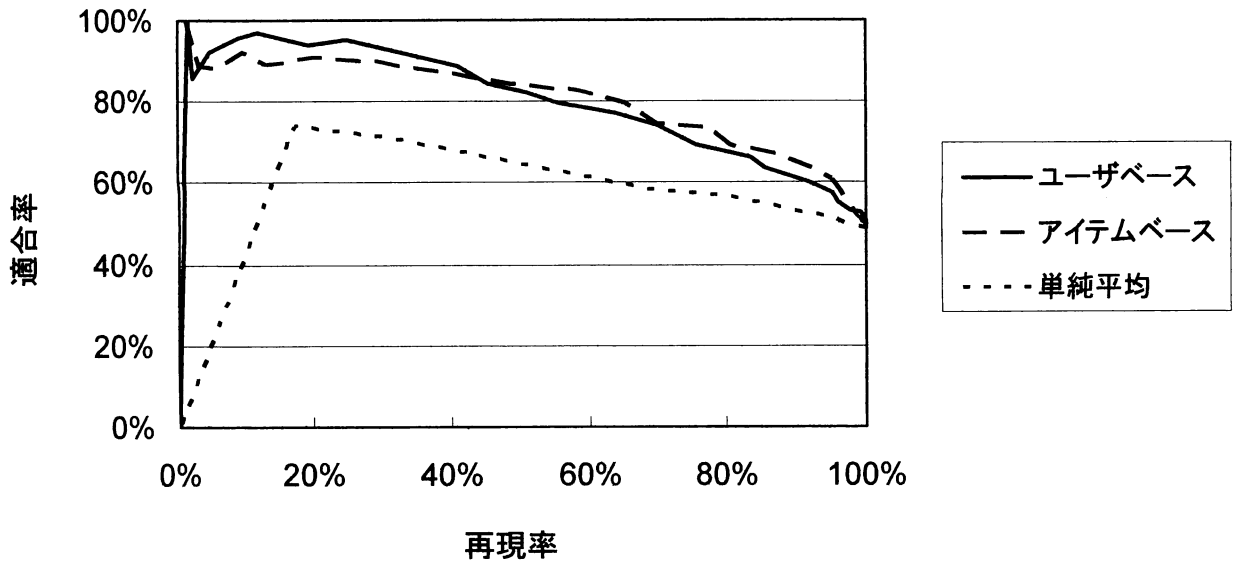

図 2各手法による予測の適合率と再現率

適合率:

$$適合率=\frac{予測の正答数}{使用と予測した数}\times 100 \quad (4)$$

再現率:

$$再現率=\frac{予測の正答数}{実際に使用している数}\times 100 \quad (5)$$

F1値:

$$F_1=\frac{2\times 適合率\times 再現率}{適合率+再現率} \quad (6)$$

4.3. 実験手順

実験は,ユーザベース手法である類似したソフトウェア開発者からの予測(EU)と，アイテムベース手法で

$$R(a,b)=\frac{\sum_{i=1}^{m} u_{i,b}}{全技術者数} \quad (7)$$

ある類似したソフトウェア開発技術からの予測(EI)について評価した．実験手順を以下に示す．

1. アンケートで得られた結果である１～４の評価値を，そのまま変数の値とし，知らないと答えた箇所は値を記入せずにデータセットを作成した．
2. 類似度計算アルゴリズム，予測値計算アルゴリズム，重み付けアルゴリズム，類似ケース数を設定し，CF を用いて予測値を算出し，最適なアルゴリズムの組み合わせを決定した．
3. 決定した手法を用いて，ジャックナイフ法に基づき予測精度を求めた．ジャックナイフ法とは，m 個のケースと n 個のアイテムを持つデータセットに対して，1 つのケースをテストデータとして選択し，残りの m-1 個のケースを用いて m 個のアイテム全てについて予測するアイテムの評価だけを不明と考え予測を行う．これを n 回繰り返して予測精度を調査する手法である．
4. EU，EI に対して手順3を適用した後，それぞれの予測精度を調べた．

また，提案方法の精度と比較するために，平均値を用いた予測(EA)を行った．この場合の推薦スコア $R(a,b)$ は下記の式で求める．

表 2 各手法の精度

	絶対誤差平均	F1値
ユーザベース	0.56	0.76
アイテムベース	0.58	0.74
単純平均	0.72	0.67

4.4. 実験結果

最適な類似ケース数を見つけるために，予備実験を行った．予備実験の結果，ユーザベース手法に基づく推薦，アイテムベース手法に基づく推薦，いずれも類似ケース数が 6 のときに最も F1 値が大きくなり，予測精度が高いことを示した．F1 値が最も大きい場合の各評価を表 2 に示す．また，ユーザベース，アイテムベース，単純平均それぞれの適合率と再現率の関係を図.1 に示す．絶対誤差，F1 値，共に提案方法による予測は単純平均による予測よりも精度が高いといえる．今回実験を行った中では，ユーザベースで予測を行う

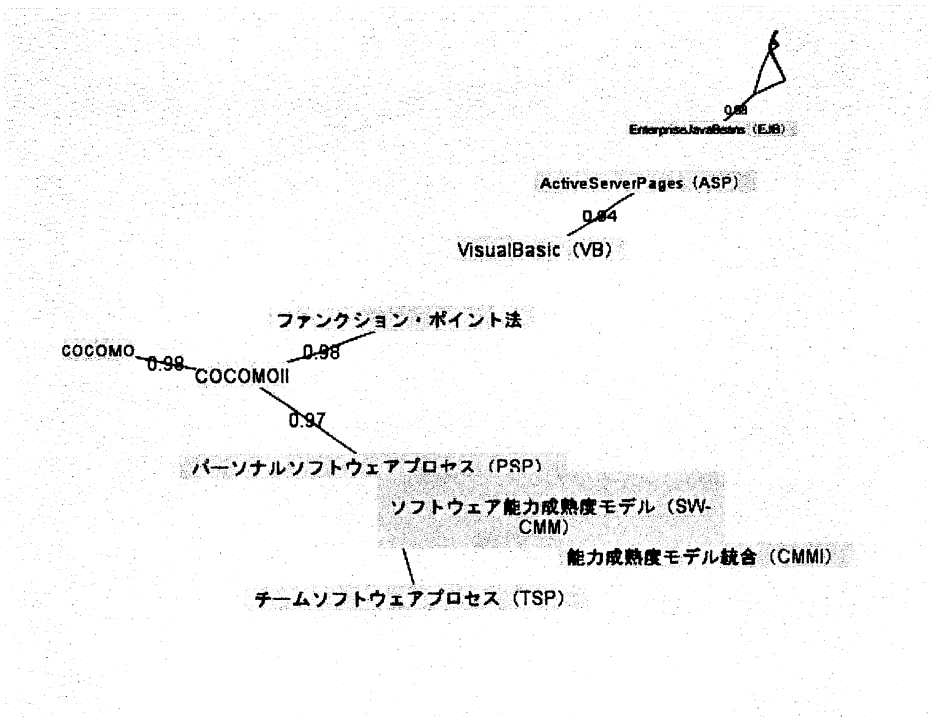

図 3 類似開発技術と類似度

ことが一番良い結果が得られた．今回の提案方法を用いて開発技術の類似度を計算し，可視化したものを図.2 に示す．図.2 では，主にソフトウェア開発のプロセス改善を行う手法が類似度の高い技術としてつながりがあるのが確認できる．

5. 考察

提案方法による推薦では，ユーザベース，アイテムベース共に予測値の精度が高い結果が得られたことから，　開発者に対して有用な開発技術の学習と発見に有効であると考えられる．

この実験は，数多くあるソフトウェア開発技術のうちの 1 部の開発技術に対して行ったものであり，今後はさらに多くの開発技術を含むデータセットに対して実験すると共にアンケートに対してもより多くの開発者に回答してもらい，大規模なデータセットに対しても提案方法が有効であることを検証する必要があると考えられる．

また，今回の実験では，アンケートの回答数が比較的多い結果が使用されたが，新しく出てきた開発技術を推薦する場合，多くの開発者が未評価であることが考えられる．今後は回答数と予測精度の関係を調査する必要があると考えられる．

6. 関連研究

Ohira ら[7]は，ソフトウェア開発者とプロジェクトの関係を可視化することを提案している．ユーザが携わっているプロジェクトに応じて，ユーザにとって有効な情報を持っていると考えられる開発者を見つけ出すことを助ける．ユーザは，教えてもらいたい技術などから，開発者がいるであろう所属や役職を推測して，実際に必要としている技術を修得している開発者を検索することは可能である．しかし，ユーザが推測できない所属や，ユーザが検索しようとしない（存在に気づいていない）開発者を探し出すことはできない．

提案方法は，プロジェクトの情報とそれに携わる開発者のデータを収集し，開発者の類似度の計算に，CF を用いている．その結果を元に simfinder を用いて可視化を行っている．しかし，この手法では，技術者が携わっているプロジェクトとの関連性だけに注目しているので，尋ねた技術者が自分の求めている技術を修得しているとは限らない．

7. むすび

本論文では，ソフトウェア開発技術の興味の大きさから CF を用いてソフトウェア開発技術を推薦することを提案した．評価実験の結果全ての評価で推薦精度が高いことが確認された．これより，提案方法を用いた推薦は有効であると考えられる．

今後の課題は，我々が提案したソフトウェア開発技術だけでなく，実際に現場で今でも使用されている開発技術についても取り入れて評価実験を行うことである．また，提案方法をシステムとして完成させ，開発者に実際にシステムを利用してもらい，実用性の評価を行う必要がある．

謝　辞

本研究の一部は，文部科学省「e-Society 基盤ソフトウェアの総合開発」の委託に基づいて行われた研究成果に基づく．

文　献

[1] K. Beck, Extreme Programming Explained: Embrace Change, Addison-Wesley, New York, 1999.

[2] B. Boehm, and R. Turner, "Balancing Agility and Discipline: A Guide for the Perplexed," Addison-Wesley, 2003.

[3] J. S. Breese, D. Heckerman, and C. Kadie : Empirical Analysis of Predictive Algorithms for Collaborative Filtering, Proc. of the 14th Conference on Uncertainty in Artificial Intelligence, pp.43-52, (1998).

[4] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using Collaborative Filtering to Weave an Information Tapestry," Comm. of the ACM, Vol.35, No.12, pp.61-70 (1992).

[5] J. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl, "Evaluating collaborative filtering recommender systems", ACM Transactions on Information Systems (TOIS), Vol.22 , No.1, pp.5-53, 2004.

[6] 経済産業省, 組込みソフトウェア開発力強化推進委員会(監修): 2004年版 組み込みソフトウェア産業実態調査報告書, 2004.

[7] M. Ohira, N. Ohsugi, T. Ohoka, and K. Matsumoto, "Accelerating cross-project knowledge collaboration using collaborative filtering and social networks" MSR 2005: International Workshop on Mining Software Repositories ,Saint Louis, May, 2005

[8] N. Ohsugi M. Tsunoda, A. Monden, and K. Matsumoto, "Applying Collaborative Filtering for Effort Estimation with Process Metrics," Proc. of the 5th Int'l Conf. on Product Focused Soft. Process Improvement, Springer, Berlin Heidelberg, pp.274-286 (2004).

[9] M. Paulk, B. Curtis, M. Chrissis, and C. Weber, "Capability Maturity Model for Software (Version 1.1)," CMU/SEI-93-TR-024, 1993.

[10] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews," Proc. ACM Conf. on Computer Supported Cooperative Work (CSCW'94), pp.175-186, Chapel Hill, North Carolina, U.S.A, Oct. 1994.

[11] B. M. Sarwar, G. Karypis, J. A. Konstan, and J.T. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," Proc. 10th International World Wide Web Conference (WWW10), pp. 285-295, Hong Kong, May, 2001.

[12] Thomson Corporation, "Thomson Derwent Patent Focus Report 2002-3," http://scientific.thomson.com/, 2003.