Vol.2012-SE-177 No.8
2012/7/20

クラウドソーシングを前提としたソフトウェアの分解方法の提案と評価

碓氷 裕紀[1]　森崎 修司[1]

概要：クラウドソーシングを活用したソフトウェア開発の際に重要となるソフトウェア分割の方法を提案する．本稿で前提とするクラウドソーシングでは分散環境において個々の開発者が非同期型のコミュニケーションをとるため，コミュニケーション量をなるべく小さくすることにより，効率化が期待できる．本稿で提案する分割方法では，コミュニケーション量を減らすことを目的とし，開発者間で相互に更新，参照されるデータの定義を事前に決めておく方法，及び，相互に更新，参照されるデータの参照，更新 API を定義しておく方法，である．提案方法を演習問題として実際に動作させたことのあるオンラインショッピングサイトのソフトウェアに適用し，適切な分割ができるかを試行した．また，分割方法と試行をオンラインショッピングサイトの構築に携わっているソフトウェア開発の熟練者から意見をもらった．

キーワード：クラウドソーシング，ソフトウェア分割

1. はじめに

商用ソフトウェアの開発はプログラマが同じ時間に同じ場所で開発を進めることが多い. このような開発方法とは別にプログラマ自身が仕事を選び，個別に作業ができるような開発方法を生み出すことで雇用の幅も広がり，プログラマ自身も規定の労働時間に縛られることなく仕事をすることができる．またプログラマが規定の場所や時間に縛られず個人ごとに分散的に開発を進められれば開発効率の向上も見込める．我々はこのような開発を実現する方法として，クラウドソーシングを活用したソフトウェア開発の概要を文献 [1] に示した．本稿では文献 [1] におけるソフトウェアの分割方法を述べる．

クラウドソーシングとは様々な課題解決のために多数の人々から手助けを得るものである．過去 10 年間，クラウドソーシングを利用した多数のサービスがインターネット上に登場してきた．実例としてはウィキペディア，Linux の開発，Yahoo!知恵袋などがあり，現在も多くのクラウドソーシングサービスの開発が進められている．[2] 本研究ではクラウドソーシングをソフトウェア開発へ適用する．通常のソフトウェア開発では開発するソフトウェアが大規模になる場合ソフトウェア分割をし，複数人で開発を担当する．クラウドソーシングを活用したソフトウェア開発においても開発するソフトウェアの規模を考慮し，ソフトウェアの分割が必要になる場合がある．本稿ではクラウドソーシングをソフトウェア開発に適用するためのソフトウェアの分割方法を述べる．

本研究の関連事項として 2006 年に AOL 社がクラウドソーシングを利用してソフトウェアの拡張を行った事例がある．彼らは自社が持つメールソフトの機能拡張を Top Coder から選抜されたチームに依頼した．Top coder とはオンライン上でプログラミングコンテストを行う web サービスである．AOL 社は Top coder から選抜されたチームに設計からテストまでを委託した．

この事例はチーム単位で開発を進められるようなソフトウェア分割を行なっているが，本研究ではプログラマが個人ごと開発を進められるソフトウェア分割を提案する．従来のソフトウェア分割は機能単位ごとにサブシステムに分割し，さらにモジュール単位に分割するというものが一般的であった．本研究では開発を任されたプログラマ同士が密接なやり取りをせず個人ごとに分散して開発でき，またソフトウェアを詳細に設計せずに分割できる方法を提案する．このような方法として本稿ではユースケースシナリオを用いたソフトウェアの分割方法を提案する．

2. 前提

クラウドソーシングの定義，クラウドソーシングを活用してソフトウェア開発を行う手順，また開発対象のソフト

[1] 静岡大学
Faculty of Informatics,Shizuoka University

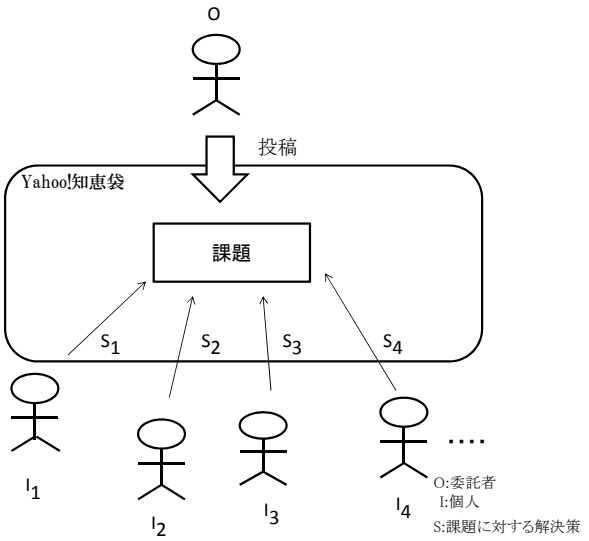

図 1 Yahoo!知恵袋によるクラウドソーシング例
Fig. 1 An example of crowdsourcing according to Yahoo! Answers

ウェアをどの程度まで定義しておくかを述べる．

本稿ではクラウドソーシングを活用してソフトウェア開発を委託する側を「委託者」，また開発を受託する側を「受託者」と呼ぶ．本稿での「委託者」とは個人ではなく，ユーザ企業などの組織を想定している．また「受託者」とはプログラミング経験のある個人を想定している．また本研究ではクラウドソーシングを活用した開発対象のソフトウェアとして WEB 画面とデータベースを含んだソフトウェアを想定しており，ゲームや組み込み系のソフトウェアは含まないこととする．

2.1 クラウドソーシングとは

クラウドソーシングとは様々な課題解決のために多数の人々から手助けを得るものである．過去 10 年間，クラウドソーシングを利用した多数のサービスがインターネット上に登場してきた．実例としてはウィキペディア，Linux の開発，Yahoo!知恵袋などがあり，現在も多くのクラウドソーシングサービスの開発が進められている．[2] クラウドソーシングでは委託者は問題解決や作業の委託のためにネットワーク上に存在する多くの個人に助力を求める．クラウドソーシングの概要を Yahoo!知恵袋の例を用いて図 1 に示す．Yahoo!知恵袋では委託者から投稿された課題に対してネットワーク上の多くの個人が解決策を提案する．このようにクラウドソーシングには従来のように特定の個人を指定して委託を行うのではなく，ネットワーク上の不特定多数の個人から労力や解決策を募集するという特徴がある．また委託された個人同士でのやり取りなども行われない．

2.2 クラウドソーシングを活用したソフトウェア開発の手順

2.1 節で述べたとおり従来のクラウドソーシングは受託者全員が委託者から提示された課題に対して成果物を提出するというものが一般的であった．本研究では受託者を募集し，その中から実際にソフトウェアの開発を委託する開発者を決定するという方法を用いたソフトウェア開発を提案する．図 2 にクラウドソーシングをソフトウェア開発に適用する際の手順を示す．図 2 中 (1)〜(3) の流れに従い手順を説明する．

(1) 委託者は作成予定のソフトウェアを受託者が個人ごとに作成できるよう複数の部分に分割する．また委託者は分割した部分について仕様と必要に応じてテストケースを作成する．委託者はネットワークを通じて作成予定のソフトウェア概要とソフトウェアの分割部分を提示する．

(2) 受託者は作成するソフトウェア概要を確認し，分割されたソフトウェアのどの部分を担当するのか選択する．受託者はどの部分を担当するか選択した後，委託者に自身のプログラミング経験や過去の開発事例，また可能であれば追加できる仕様やテストケースを考え委託者に伝える．委託者は受託者から伝えられた情報を基にどの部分をどの受託者に委託するのか最終決定を行う．

(3) 受託者は委託されたそれぞれの部分について作成を行う．受託者は作成完了後，自身が作成した部分を委託者に渡す．委託者は受託者により作成された部分を集め，正確に動作するか結合テストを行い確認をする．

2.3 作成予定のソフトウェアの定義

2.2 節で示した手順の中で「作成予定のソフトウェア」を，委託者がどの程度まで定義するかを述べる．本研究では，委託者は対象ソフトウェアとして以下を定義する．

- **仕様，テストケース，ユースケース等**
  委託者はソフトウェア開発の委託を確定する前に，分割したソフトウェアの各部分ごとの仕様と必要に応じたテストケース，受け入れテストケース，ユースケースを用意する．また手戻りを少なくソフトウェアの分割をするためにも要件からソフトウェアとして実現可能な機能を決めておく必要がある．委託者は要件からソフトウェアに必要な機能を定義する．表 1 に機能の例を示す．機能には名称と内容を含め，ソフトウェアとして実現可能なことを機能名ごとにまとめる．
- **その他の定義**
  委託者はクラウドソーシングを活用してソフトウェア開発を行うために，ソフトウェアの定義だけではなく，

表 1　スケジュール管理機能における機能定義例
Table 1　Example of functional defnition according to scheduling management

機能	機能の内容
登録，削除，修正，参照機能	・管理者は従業員の予定を登録，削除，修正，参照できる． ・管理者は登録した作業員の予定を公開，非公開どちらかを選択できる．
検索機能	・全従業員は「従業員名」から対象従業員のスケジュールを検索し表示できる． ・全従業員は日付から，該当日の従業員スケジュールを検索し表示できる
ファイル取得機能	・全従業員は従業員のスケジュールを個人ごとに Excel，CSV 形式の ファイルで出力できる．

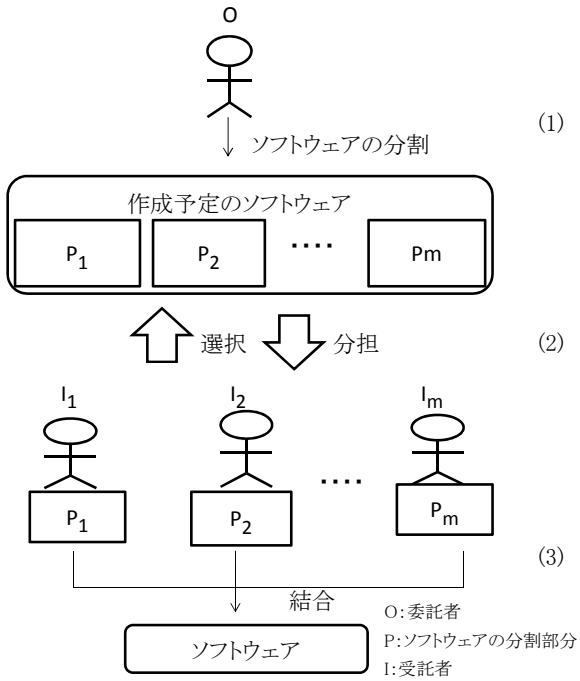

図 2　クラウドソーシングを活用してソフトウェア開発を行う手順
Fig. 2　Procedure of crowdsourcing software development

「受託者の作業内容」や「開発報酬」，また「開発に必要なスキル」についても必要に応じて定義できる．開発に必要なスキルは表 2 のような項目をスキルとして定義する．

3. ソフトウェアの分割方法

2.2 節で示した手順においてソフトウェアの分割方法を提案する．委託者に負担が掛からず，また受託者が個人で作業できるよう細かい分割方法を提案する．本稿ではユースケースシナリオを用いた分割方法を提案する．本稿で提案する方法ではプログラムの動作に関するユースケースシナリオを組み合わせてソフトウェアを作成する．そのためユースケースシナリオのプログラムの動作を分割の基準とし，アクターの動作は考慮するが分割の基準には含めない事とする．本稿の分割方法として用いるユースケースシナリオの例を表 3 に示す．

分割方法としてユースケースシナリオのみを考えデータベースを定義をしなければ，各ユースケースシナリオがデータベースのどのデータへアクセスするのか不明確になる場合がある．さらに異なるユースケースシナリオ同士で同じデータにアクセスするという問題も生じる．（図 3）本稿で用いるデータへのアクセスとはデータベースへのデータの登録，削除，更新，参照を行う操作である．

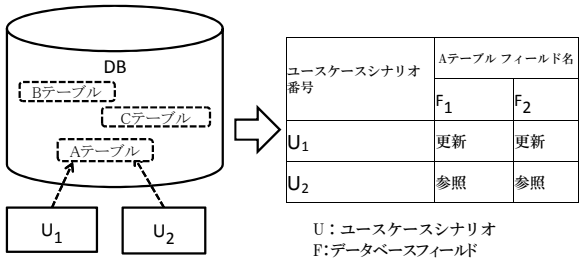

図 3　異なるユースケースシナリオ同士で同じデータにアクセスする場合
Fig. 3　A case to access the same data among different use case scenarios

図 3 の場合，U_1 で更新したデータを U_2 で参照しており，データベースを定義しない場合，U_2 はどのテーブルのどのデータを参照するのか不明確になる．本研究ではこれらの問題を解決するものとして，委託者の作業量を考慮し，「データ定義による分割方法」と「メソッド定義による分割方法」の 2 つを提案する．

3.1 データ定義による分割方法

データ定義による分割方法では委託者がユースケースシナリオによる分割を行い，データベースのテーブル，フィールド，データ型までを定義し，それぞれのユースケースシナリオがどのデータにアクセスするのか定義する方法である．この方法を用いることでどのユースケースシナリオがどのデータにアクセスするのか明確にすることができ，異なるユースケースシナリオ同士で同じデータにアクセスする問題も解決できる．分割の手順を以下に示す．

(1) 委託者は 2.3 節で定義したソフトウェアの情報からユースケースシナリオを抽出する．抽出したユースケースシナリオを分割の基準とする．
(2) 委託者はソフトウェアに必要なデータベースのテーブ

表 2 開発に必要なスキル例
Table 2 Example of the necessary conditions for software development

開発に必要なスキル	・java での web アプリケーション開発の経験がある． ・実務で医療システムの開発経験がある． ・ruby での実務経験が 2 年以上

表 3 図書館システムにおけるユースケースシナリオ例
Table 3 Example of use case scenarios according to library system

ユースケースシナリオ番号	ユースケースシナリオ
U_1	システムは蔵書一覧を表示する
U_2	システムは本の詳細情報を表示する
U_3	システムはデータベースの蔵書情報を貸出中にする
U_4	システムは貸出日と返却期限を表示する

ル，フィールド，データ型を定義する．また各ユースケースシナリオがデータベースのどのデータにアクセスするのかを定義し，ユースケースシナリオごとに受託者を募集する
(3) 受託者は実装したいユースケースシナリオを選択し，自身の開発経験や可能であれば追加の仕様とテストケースを委託者に提出する
(4) 委託者は提出された情報を基に開発を委託する受託者を決定する．
(5) 委託者は個々のユースケースシナリオがデータベースのどのデータにアクセスするのか確認して，ユースケースシナリオごとにプログラムを実装していく．

3.2 メソッド定義による分割方法

メソッド定義による分割方法では委託者がユースケースシナリオによる分割を行い，メソッドを定義する．この方法ではデータベースへアクセスする処理などをメソッドとして定義する．委託者はメソッドの中身の作成とデータベースの定義は行わない．

委託者は受託者として「メソッドを担当する受託者 (以下メソッド担当者)」と「メソッドを使用しユースケースシナリオごとにプログラムを実装する受託者（以下ユースケースシナリオ担当者)」の 2 種類の人材を募集する．メソッド担当者はメソッドの中身の作成とデータベースのテーブル，フィールド，データ型の定義を行う．ユースケースシナリオ担当者はメソッド担当者が完成させたメソッドを使い，ユースケースシナリオごとにプログラムを実装する．データベースへアクセスする処理をメソッドとして定義することで，データベースの構造が不明確でも，データへアクセスすることができる．(図 4) 分割の手順を以下に示す．
(1) 委託者はデータ定義による分割方法と同じくユースケースシナリオを抽出し，抽出されたユースケースシナリオを分割の基準とする．
(2) 委託者はデータアクセスに関する処理をメソッドとして定義する．また「メソッド担当者」と「ユースケースシナリオ担当者」の両者を募集する．
(3) 受託者は実装したいメソッドまたはユースケースシナリオを選択し，自身の開発経験や可能であれば追加の仕様とテストケースを委託者に提出する．
(4) 委託者は提出された情報を基に開発を委託する受託者を決定する．
(5) メソッド担当者は委託者が定義したメソッドが機能するようメソッドの中身を実装する．またデータベースのテーブル，フィールド，データ型の定義を行う．ユースケースシナリオ担当者は完成されたメソッドを使ってユースケースシナリオごとにプログラムを実装していく．メソッド定義による分割方法の委託概要を図 5 に示す．

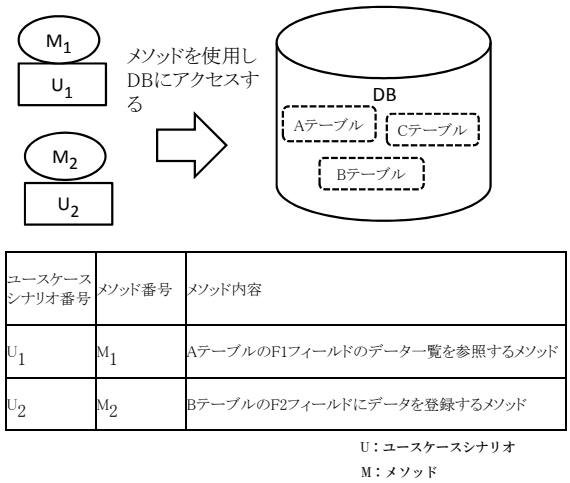

図 4 メソッド定義による分割方法の概要
Fig. 4 Overview of method difinition divide

3.3 分割に関連したテストケースの作成

ソフトウェアに複雑な計算ロジックがある場合，データ定義による分割方法のユースケースシナリオとメソッド定義による分割方法のユースケースシナリオ，メソッドに関

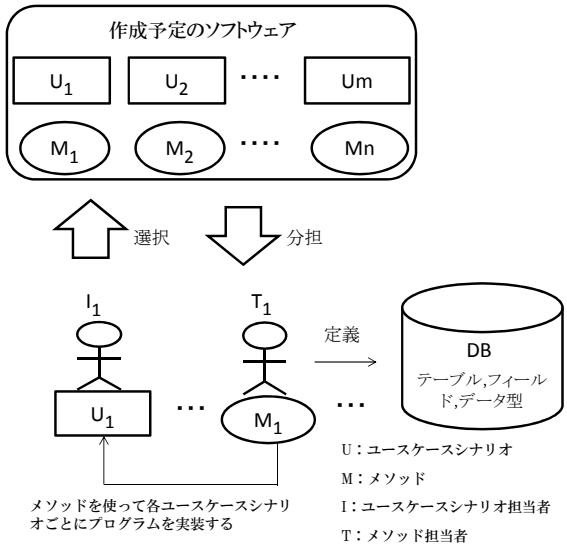

図 5 メソッド定義による分割方法の委託概要
Fig. 5 Procedure of method difinition divide

して，委託者がテストケースを作成する．テストケースの作成は委託者が負担を軽減するためにツール用いてもよい．テストの効率化には様々なツールが存在するが，委託者はユースケースシナリオによりユーザ側から見たプログラムの動作を抽出できるため，ユーザ側からみた外部設計に関してテストケースを作成できるツールを選択する．たとえば，Cucumber は自然言語でユーザ側から見たソフトウェアの動作シナリオを記述できる．自然言語で記述できるため非エンジニアでも読むことが可能である．このツールのように外部設計のテストを行えるツールをテストケースの作成ツールとして選択すれば委託者は効率的にテストケースを作成することができる．

4. 分割の試行

3 章で提案した 2 つの方法について試行を行い，熟練者から意見を頂いた．試行したソフトウェアの概要と 2 つの方法を試行した手順と結果を述べる．

4.1 試行に用いたソフトウェアの概要

本稿で提案した 2 つの分割方法をオンラインショッピングサイトシステムの仕様書を用いて試行を行った．このシステムは大学の演習課題として開発されたシステムであり，実際のシステムとして実現できる仕様書に基づき開発された．

システムの利用者は「ユーザ」と「管理者」に分かれる．ユーザはシステムを利用して「買い物」をすることができる．管理者はシステムを利用して「商品の管理」,「発送」,「顧客情報の管理」などを行うことができる．

対象としたオンラインショッピングサイトシステムの概要を表 4 に示す．またオンラインショッピングサイトシステムのユースケースを表 5 にオンラインショッピングサイトシステムの機能一覧を表 6 に示す．このシステムの仕様書を用いてデータ定義による分割方法，メソッド定義による分割方法を試行した手順と結果を述べる．

表 6 対象としたオンラインショッピングサイトシステムの機能一覧
Table 6 List of functions in online shopping site system

機能	機能の内容
ショッピング機能	ユーザが商品を購入する機能
商品発送機能	管理者が商品を発送する機能
商品情報管理機能	商品情報の登録，削除，変更，確認を行う機能
顧客情報管理機能	顧客情報の登録，削除，変更，確認を行う機能

4.2 データ定義による分割方法

データ定義による分割方法を試行した手順と結果を述べる．

(1) 仕様書に記載されたシステム全体のユースケース，ソフトウェアの機能情報等から 40 個のユースケースシナリオを抽出した．抽出したシナリオの一部を表 7 に示す．

(2) データベース定義については顧客情報，商品情報，伝票情報，明細情報，カート情報それぞれを格納するテーブル，フィールド，データ型を定義した．

4.3 メソッド定義による分割方法

メソッド定義による分割方法を試行した手順と結果を述べる．

(1) 仕様書に記載されたシステム全体のユースケース，ソフトウェアの機能情報等から 40 個のユースケースシナリオを抽出した．

(2) 抽出したユースケースシナリオにおいてデータベースへデータの登録，削除，更新を行う処理をメソッドとして定義した．今回のシステムではデータベースへの登録，削除，更新を行う処理として 18 個のメソッドを抽出した．メソッド一覧を表 8 に示す．

4.4 熟練者からの意見

試行した 2 つの分割方法とテストケースの作成，クラウドソーシングを活用してソフトウェア開発を行う際の考慮点について熟練者から意見を頂いた．

4.4.1 試行した分割方法

試行した 2 つの分割方法について熟練者からの意見を示す．

- **データ定義による分割方法**
  データ定義による分割方法はウォーターフォール開発に似た側面をもち，委託者がデータベースのテーブル，フィールド，データ型まで定義する．そのためデータ

表 4　対象としたオンラインショッピングサイトシステムの概要
Table 4　Overview of online shopping site system

内容	サーバ上で動作する Web アプリケーション ユーザはシステムから商品の購入を行える 管理者はシステムから顧客，商品の管理を行える
開発言語	Java
ステップ数	約 3000 行 (空白行，コメント行を除く)
開発プロセス	ウォーターフォール開発
データベーステーブル数	5 テーブル
画面数	約 30 画面

表 5　対象としたオンラインショッピングサイトシステムのユースケース一覧
Table 5　List of use cases in online shopping site system

ユースケース名	アクタ	目的
商品の購入を行う	ユーザ	ユーザがインターネットでショッピングをする
商品の発送を行う	管理者	管理者が商品の発送を行う
商品情報を管理する	管理者	管理者が商品情報を管理する
顧客情報を管理する	管理者	管理者が顧客情報を管理する

定義による分割方法は委託者と受託者の間でデータベースの構造などアーキテクチャの合意を取ることが難しくなる．クラウドソーシングをソフトウェア開発に適用する場合，委託者と受託者は離れて開発を行うため，委託者がデータベース設計などを修正した場合に，受託者との合意に時間がかかってしまう可能性があるという意見を頂いた．

- **メソッド定義による分割方法**
  メソッド定義による分割方法はアジャイル開発に似た側面を持ち，比較的規模が小さい web アプリケーションの開発などに適しているという意見を頂いた．バグを厳しく追求しないソフトウェアならばメソッド定義による分割方法でも開発は可能であるが，テストケースをどこまで細かく作成できるかが重要であり，メソッドを定義するだけでなくメソッドに何を入れたら何が返ってくるかという具体例も定義したほうが良いという意見を頂いた．
- **２つの方法の使い分け**
  作成しようとしているソフトウェアの規模や，バグをどれだけ厳しく追求する必要があるかによってデータ定義による分割方法とメソッド定義による分割方法の使い分けが考えられる．データ定義による分割方法は委託者がデータベースのテーブル，フィールド，データ型まで定義する．そのため正確な動作を求め，規模が大きいソフトウェアを作成するならばデータ定義による分割方法が適している．またメソッド定義による分割方法では，作成するソフトウェアの規模が大きくなればなるほど実際に適用することが難しくなるのでないかという意見を頂いた

4.5　その他の意見

テストケースの作成やクラウドソーシングを活用してソフトウェア開発をする際の考慮点，その他に得られた意見を示す．

4.5.1　テストケースの作成

本稿ではテストケースの作成について Cucumber を例示したが，Cucumber は要件やユースケースに対してテストケースを作成する場合に向いているツールであり，結果が自然言語で出力されるため委託者も結果を理解しやすいという意見を頂いた．

4.5.2　クラウドソーシングを活用してソフトウェアを開発する場合の限界点

クラウドソーシングを活用してソフトウェア開発をする場合，作成するソフトウェアの計算ロジックがどこまで複雑になるかという点で開発が可能になるか不可能になるか分かれる可能性があり，その点についても考慮したほうが良いという意見を頂いた．

4.5.3　動作環境など

クラウドソーシングを活用してソフトウェア開発をする場合には，ソフトウェアの分割方法だけでなく，動作環境など，インフラまでを考える必要がある．仮に動作環境をクラウドとするならば，後はソフトウェアの機能を満たすだけでよくなるという意見を頂いた．また開発方法だけでなく，開発費用の決め方や合意のとり方について考えるべきだという意見を頂いた．

5.　まとめ

本稿ではクラウドソーシングを活用したソフトウェア開発の概要を示し，ソフトウェアの分割方法を提案した．本稿で前提とするクラウドソーシングでは分散環境において

表 7　オンラインショッピングサイトシステムにおけるユースケースシナリオの一部
Table 7　Part of use case scenarios in online shopping site system

シナリオ番号	ユースケースシナリオ
U_1	システムは商品一覧を表示する
U_2	システムは商品詳細を表示する
U_3	システムはカートに商品を追加する
U_4	システムはカートの情報を表示する
U_5	システムは DB に合計金額と購入商品登録し，表示する
・・・	・・・

表 8　オンラインショッピングサイトシステムにおけるメソッドの一覧
Table 8　List of method in online shopping site system

メソッド番号	メソッド名	メソッド内容
M_1	get-itemlist	商品情報一覧を取り出す
M_2	get-iteminfo	商品の詳細情報を取り出す
M_3	ins-itemlist	新規商品を登録する
M_4	update-item	1 つの商品情報を更新する
M_5	delete-item	1 つの商品を削除する
M_6	get-sql-customerlist	顧客情報一覧を取り出す
M_7	get-customerinfo	顧客の詳細情報を取り出す
M_8	ins-customer	新規顧客を登録する
M_9	update-customer	一人の顧客情報を更新する
M_{10}	delete-customer	一人の顧客情報を削除する
M_{11}	get-cart-info	カートの中身を取り出す
M_{12}	ins-cart	カートに商品を登録する
M_{13}	delete-cart	カートの中の一つの商品を削除する
M_{14}	get-buytable	伝票情報一覧を取り出す
M_{15}	ins-buytable	新規伝票を登録する
M_{16}	update-send-data	商品の発送情報を更新する
M_{17}	get-detailtable	明細情報一覧を取り出す
M_{18}	ins-detail	新規明細を登録する

個々の開発者が非同期的に作業を進めていくため，定義，決定，相談のためのコミュニケーションが増えると，開発時間が長くなる場合がある．本稿で提案するソフトウェアの分割方法では，開発者間のコミュニケーションを減らすために，データの相互更新や相互参照を事前に定義することにより，これを実現する．具体的には，事前にデータベースのテーブル定義を実施しておく方法，相互更新や相互参照に必要な API を事前に定義しておく方法の 2 つであり，このいずれかとシステムのユースケースを委託者が提示し，受託者を募集する．

2 つの分割方法の試行には大学の演習課題として仕様書，設計ドキュメント，ソースコードが存在し，実際に動作するオンラインショッピングサイトシステムを対象として 2 つの分割方法を用いて，分割の試行を行い，実際に分割できることを確認した．また，分割方法と試行結果に対して，オンラインショッピングサイトの開発に携わるソフトウェア開発の熟練者から意見をもらった．今後は委託者と受託者間における作業管理や課題管理，変更要求管理を検討していく予定である．また熟練者からの意見を参考に動作環境や，テストケースについても検討し，提案方法を実証的に評価する予定である．

謝辞　貴重な意見をくださった楽天株式会社 川口恭伸氏に感謝の意を表する．本研究は文部科学省科学研究補助費（基盤研究 B:課題番号 23300009）による助成を受けた．

参考文献

[1] Usui Y. and Morisaki S.: *An Approach for Crowdsourcing Software Development*, Proceedings of IWSM-MENSURA 2011, pp. 32-33(2011)
[2] Doan A.,Ramakrishnan R and Halevy A Y.: *Crowd-Sourcing Systems on the World-Wide Web*, Communications of the ACM vol. 54, No. 4, pp. 86-96(2011)
[3] Howe J.: *The rise of crowdsourcing*, Wired Magazine, Vol. 14, No. 6(2004)
[4] Hatanaka M.: *Analyses of Contractors' Nash Equilibria on the Problem Solving Mechanism by Crowdsourcing Master Thesis*, Department of Social Informatics Graduate School of Informatics Kyoto University(2010)