ソフトウェア工学 125−11
(2000. 1. 14)

ソフトウェア開発対象業務の動的側面に注目した要求獲得支援

森田俊夫† 山田宏之‡

† 愛媛大学大学院 理工学研究科 情報工学専攻
‡ 愛媛大学 工学部 情報工学科

要求分析段階では，開発者と依頼者とのコミュニケーションが重要であり，開発対象のシステムについて互いに同一の理解を持つ必要がある．依頼者はオブジェクトの配置，関係のようなシステムの静的な側面よりも業務フローのような動的な側面に関心を持ちやすい．そのため，業務の動的側面をモデル化した方が依頼者の理解が得やすく，要求も引き出しやすいと考える．そこで，本稿では開発対象業務の動的側面に注目し，ユースケース図，ユースケース図の拡張，及び，アクティビティ図による業務のモデル化に基づく要求獲得支援方法について提案し，ある例題に適用した結果を示す．

A Requirement Acquisition Support Focused from a Dynamics Viewpoint of Businesses

Toshio Morita† and Hiroyuki Yamada‡

† Graduate School of Computer Science, Ehime University
‡ Dept. of Computer Science, Ehime University

It is important that developers and users communicate with each other in the requirements analysis stage, and then they need to have the same understanding of the system. Users can be interested in the system in a dynamics viewpoint, such as a business flow, rather than a statics viewpoint, such as objects and relations among objects. Thus we think that dynamics modelings are better for users to understand the system and for developers to acquire requirements. So this paper focuses on dynamics viewpoints of businesses, suggests a requirement acquisition support based on usecase diagrams, extensions of them, and activity diagrams, and shows the result of application of them to some example.

1 はじめに

ソフトウェア開発における困難な問題として開発対象業務を分析し，どのような機能をシステムに付加するか，さらに，その機能をどのように実現するかを決定することがある[1]．要求分析・獲得に関する研究はこの問題の解決を目指している．要求分析・獲得の不十分さや要求分析段階での開発者と

依頼者との間におけるシステム理解の不一致は，後の開発プロセスに影響を及ぼし，コスト増加，スケジュールの遅れなどの問題を引き起こす．

この問題を避けるために，依頼者と開発者とは十分にコミュニケーションを行い，システムに対する共通理解を深めていく必要がある．開発者は業務を分析し，理解できない部分，あるいは依頼者からの要求がシステム開発において不十分である部分を依頼者に提示し，明確化する必要がある．そのためには依頼者にとって理解しやすい形でシステムを表現する必要がある．依頼者がシステム開発において，最も重視することは「入力に対して，正しい出力結果」が得られることである．そこで，システムを表現する場合，オブジェクトの配置，関係などの静的な側面をモデル化して依頼者に提示するよりも，業務の開始から終了までのプロセス(あるいは，入力から出力までのシーケンス)の動的な側面をモデル化して提示した方が依頼者もシステムに対してより興味を示しやすい．

しかし，システムのモデル化を扱ったオブジェクト指向開発方法論は種々提案されているが，これらは要求分析・獲得については明確には扱っていない．そのため，方法論で用いられているモデルをどのように要求分析・獲得に適用するかは開発者に任されており，現在のところ分析・獲得に向いた方法はほとんど存在していない．

本稿では，要求工学ワーキンググループ[4]で取り上げられている共通例題「国際会議のプログラム委員長の業務」[5]で挙げられている12の業務の動的側面に注目し，業務シーケンスのモデル化を通して，依頼者と開発者とのコミュニケーションに基づく要求獲得支援方法を提案する．モデル化の際にはユースケース図及びアクティビティ図を用いているが，その表記法は統一モデリング言語：UML1.3[3]に基づいている．

2 要求獲得支援プロセス

依頼者とのコミュニケーションに基づいて要求獲得を行う場合，まず，開発対象システムの大まかな部分について議論を行い，徐々に詳細な部分へと議論を進めていくという方法が妥当である．本稿では以下のような手順で業務のモデル化を行い，要求獲得支援を行う．

1. ユースケース図の作成
2. 要求の分類
3. 拡張ユースケース図の作成
4. アクティビティ図の作成

このような手順の中で，システム化を行う上で必要な事項が依頼者から要求として明確に提供されていないことがある．このような部分は開発者と依頼者とのコミュニケーションを通して明確にすることになる．本研究では，できるかぎり図を活用して開発者からの開発システムの提案を行い，依頼者から情報を引き出す形で行うようにする．なぜなら，依頼者がコンピュータに関する知識を十分に持っていない場合，開発者からの専門的な質問が理解できない場合があるため，あるいは何もない状態から依頼者の見解を求めるよりも，何かを提示し，それに対する依頼者の見解を求める方が依頼者も意見を出しやすいのではないかと判断したからである．

2.1 ユースケース図

ユースケース図[2]はその記述方法が単純であることから依頼者の理解が容易であるため，業務分析の初期段階において，業務の概要を理解するためのモデルとして有益である．本稿では，継続していると考えられるプロセス(例えば，商品販売の場合，客が商品をレジに持って来てからお金を払って商品を受け取るまでの一連のプロセス)を一業務単位としてその単位毎にユースケース図を作成する．これは 1)大規模なシステムになるにしたがって，ユースケース数も多くなり，それらを依頼者に一度に提示すると図が膨大になり，依頼者の混乱を招くため，2)後のプロセスであるユースケース図の拡張，アクティビティ図による詳細化への移行を考慮したためである．

また，ユースケースの抽出方法は基本的に依頼者から提出される業務ドキュメント内の動詞を基準に抽出していくこととする．

2.2 要求の分類

ユースケース図を拡張するためにまず，要求の分類を考える．依頼者から出される要求には機能に関するもの，コストに関するものなどさまざまな要求が考えられるが，それらは1つの要求という集合で提示される．その集合内の1つ1つをあるカテゴリ毎に分類することにより，システムをカテゴリ毎の側面から捕らえることができる．

要求の分類方法の1つとして，機能的要求と非機能的要求とに分類することが議論されている[1]．本稿では，さらに図1のように機能的要求を機能要求，入出力要求，例外処理要求の3つに分類する．入出力要求と例外処理要求を機能的要求から分離した理由は次のようなことからである．

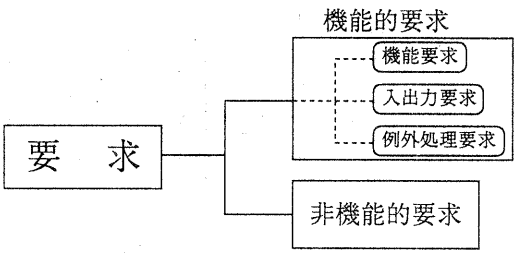

図 1: 要求の分類

入出力要求　入出力要求は「依頼者の本質」であり，依頼者が最も感心を寄せている部分である．つまり，依頼者は入力に対して正しい出力が得られるということに1番の重点を置き，その実現方法に関してはあまり感心を示さないことがある．また，入出力要求はモデルを用いて依頼者に提示するよりも，実際に簡単なユーザインタフェースの見本を作り，それを見せる方が要求を獲得しやすいと考えられる．

例外処理要求　例外処理は業務に関するもの，入力誤りなど人的な誤り，システム故障など多くの要因に対して考える必要がある．また，これらの例外を要求分析段階で全て把握することは非常に困難である．それと同時に，たとえ要求分析段階で分かっていたとしても，それを依頼者に提示する必要がないもの(例えば，コンピュータ知識の乏しい依頼者にシステム故障の場合の例外処理を提示しても無意味である)も存在する．従って，例外処理要求は発見された時点，あるいは提示するのに適当な時点で追加する必要がある．これらを機能的要求から分離しておくことで，新たな例外処理が追加されても本来の機能には追加されないので依頼者の混乱を防げる．

図1におけるそれぞれのカテゴリは次のようなことを含む．

- 機能要求
  - (制約や条件を含む)システム機能に関する要求．業務シーケンスも合わせて記述する．
- 例外処理要求
  - 現時点で認識できているすべての例外処理(ただし，操作誤り，あるいはシステム故障等に関する例外処理はこの時点で提示する必要はないので，これに含まない)
- 入出力要求
  - 入力，及び，出力が何であるかのみを記述．ユーザインタフェースのレイアウトなどの詳細は含まない．
- 非機能的要求
  - コスト，スケジュールなどシステム機能には直接的に関係しない要求．

この分類を前節で定義した業務単位毎に行う．本稿ではこれらの4つのカテゴリの内，特に機能要求，例外処理要求に注目し，この2つに対する要求獲得支援を考える．

2.3 拡張ユースケース図

ユースケース図はシステム内のプロセスのみを記述し，動作シーケンスはコラボレーション図，アクティビティ図などの動的モデル，あるいはシナリオに任されている．複数のモデル図を用いることは個々のモデル特有の観点からシステムを分析でき，そのモデルの観点から新たな要求の抜けが発見でき

るという利点がある．しかしながら，依頼者には複数のモデルを理解するための労力を必要とする．また，変換による依頼者の混乱を最小限に抑えるようにする必要もある．本稿では最終的にはアクティビティ図への変換を行っているが，ユースケース図から直接，アクティビティ図に変換を行った場合，システムの詳細度が一度に増加し，依頼者が混乱を起こす可能性があると考えた．そこで，まず，ここで述べている拡張ユースケース図を用いてシステムを詳細化し，その後，アクティビティ図に変換することを試みている．

前節における要求の分類をもとにユースケース図に次のような項目に対する表記法を追加することで拡張を行う．

- 業務シーケンス．
- ユースケースで必要とされるデータ，あるいは産物 (artifact)．
- 他のユースケース図との依存関係．

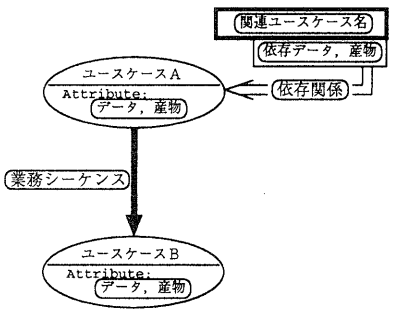

図 2: 拡張ユースケース図の表記法

図2は拡張ユースケース図の表記法を示している．業務シーケンスは実線の矢印で表現する．時間は矢印の向きに進む．矢印が途中で分岐している場合は処理が並列に進むことを意味している．データ，及び，産物はUMLにおけるユースケースの属性表記に準ずる．ユースケース間の依存関係は二重線矢印で表現し，関係名の文字列を矢印に付随する．また，その二重線矢印の根本は依存関係にあるユースケース図(ユースケース名と関連するデータのみを略記したもの)と結ばれている．

2.4 アクティビティ図

要求のさらなる詳細化を行う手段としてアクティビティ図を用いる．ここでは，各ユースケース毎に1つのアクティビティ図を作成し，拡張ユースケース図の業務シーケンス(太矢印)によってそれらをつなぎ合わせる．図3はアクティビティ図の例である．角の丸まった長方形はアクティビティを示し，矢印でアクティビティの遷移を示す．破線矢印はデータ遷移を示す(アクティビティも同時に遷移する)．また，この図は3つの swimlane で構成され，左の区画から順にそれぞれアクタ(システムユーザ)，システム，他のアクタ(あるいは他のシステム)とする．

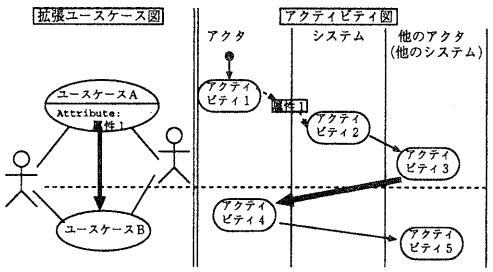

図 3: アクティビティ図の例

業務のモデル化を行う場合，業務全体を把握するためにシステム化する必要がない部分，あるいはシステム化が現在の技術では不可能な部分に対してもモデル化を行っている．siwmlane をこのように割り当てることはシステム化すべき部分と，システム化する必要がない部分，つまり，システム化の範囲を明確にするのに有効である．

3 例題への適用

これまで，要求の詳細化のプロセスについて述べてきたが，実際にこの方法をある例題に適用した結果を示す．ここで用いた例題は要求工学ワーキンググループで取り上げられている「国際会議のプログラム委員長の業務」である[5]．この例題の目的として，「国際会議のプログラム委員長の業務を支援する．投稿論文の管理だけでなく，依頼者への連絡の手紙などの作成も支援する．」が挙げられており，

12の業務によって構成されている．

本稿ではその12の業務の内，(a)スケジュール決定に関する業務と(c)プログラム委員の選出に関する業務についての分析結果を示す．それぞれの業務は次のようなドキュメントで与えられている．

(a) スケジュールを決める．決めるべきスケジュールは以下のとおり．
論文投稿締切日，プログラム委員会開催日，著者への論文の採否通知日，Camera Ready締切日，印刷業者への発送日．

(c) プログラム委員を選び，依頼を行なう．委員の名簿を作成する．
プログラム委員の名簿データは，委員名，住所，所属，電話番号，FAX番号，E-mail address，得意とする分野(担当論文の割り当てを決める際に使用する)からなる．

3.1 ユースケースの抽出

ユースケースを抽出する際にまず，文章中の動詞を探す．業務(a)「スケジュールを決定する．」の場合，下線部の「決定する」が動詞である．「決定する」のはスケジュールであるから，それらを合わせてユースケースとして「スケジュール決定」が抽出される．また，「決定する」のはプログラム委員長であるから，ユースケース「スケジュール決定」のアクタは「プログラム委員長」になる．

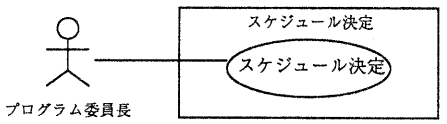

図 4: 業務(a)のユースケース図

同様に，業務(c)「プログラム委員長を選び，依頼を行う．委員の名簿を作成する．」の場合，「選び」，「行う」，「作成する」が動詞であり，それぞれユースケースとして「委員候補の選出」，「依頼」，「委員名簿の作成」が抽出される．これらのことを行うアクタは「プログラム委員長」である．ここで，「国際会議のプログラム委員長の業務」の目的として，「投稿者への連絡の手紙などの作成も支援する」ということがあった．この場合，投稿者に対するものではないが，依頼を行う際に依頼状を作成することを提案し，ユースケース「依頼状の作成」を追加する．

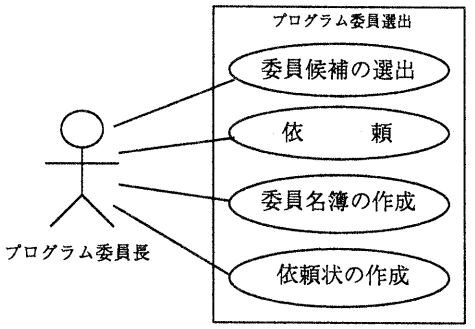

図 5: 業務(c)のユースケース図

3.2 例題に対する要求の分類

例題に対して要求の分類を行う．ただし，この例題では依頼者から非機能的要求が全く出されていないため，非機能的要求の記述は省略する．

(a)スケジュール決定については次のように分類できる．

- 機能要求——スケジュールを決定する．
- 例外処理要求
  - プログラム委員から委員会開催日の変更を要請される．
  - 委員長が日程を決定後に変更したいと考える．
- 入出力要求
  - 入力——国際会議開催日
  - 出力——論文投稿締切日，プログラム委員会開催日，論文の採否通知日，Camera Ready締切日，印刷業者への発送日

同様に(c)委員候補の選出については次のように分類できる．

- 機能要求——プログラム委員を選び，依頼を行う．委員の名簿を作成する．
  1. 委員候補を選出する．
  2. 依頼状を作成する．

3. 依頼を行う.
4. 委員名簿を作成する［制約：論文投稿締切日までに］.

- 例外処理要求
  - 委員候補に依頼を断わられる.
  - 委員が名簿データの一部(例えば，FAX番号)を持っていない.
- 入出力要求(委員名簿の作成時)
  - 入力——委員名，住所，所属，電話番号，FAX番号，E-mail address，得意とする分野
  - 出力——委員名簿

3.3 ユースケース図の拡張

前節の分類結果をもとに(a)スケジュール決定，(c)委員の選出のユースケース図を拡張した結果をそれぞれ図6，図7に示す.

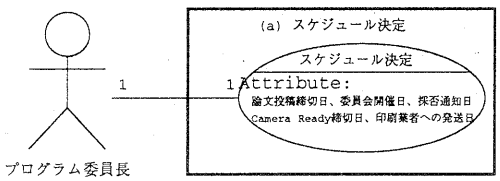

図 6: 業務(a)の拡張ユースケース図

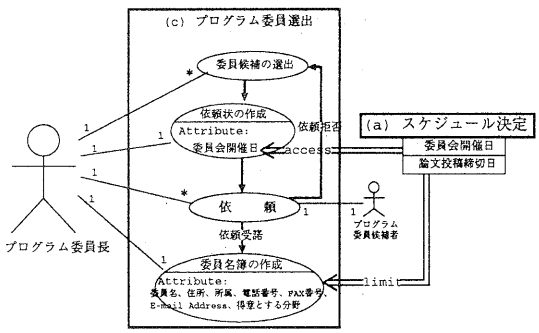

図 7: 業務(c)の拡張ユースケース図

図6はユースケース「スケジュール決定」がそのユースケースで決定すべき項目，論文投稿締切日，委員会開催日などを持ち，また，ユースケースとアクタとを結んだ直線上の数字はユースケースとアクタとが1対1対応であることを示している.

また，図7では「依頼文書作成」ユースケースが「委員会開催日」の属性を持ち，その属性が業務(a)で決定された「委員会開催日」を「access(参照)」するを示している.「依頼」ユースケースは依頼を拒否された場合,「委員候補の選出」ユースケースに戻り，新たな候補者を選出するという例外処理が示されている.「委員名簿の作成」ユースケースは属性として「委員名，住所，…」を持つことを示し，また，業務(a)の論文投稿締切日によって「limit(制限)」されることを示している.

3.4 アクティビティ図による詳細化

拡張ユースケース図では表現できないために，まだ，詳細化が不十分な部分が存在する．例えば，スケジュールの決定方法がそれに当たる.「国際会議のプログラム委員長の業務」において，スケジュールを決定する業務が存在することは記述されているが，どのようにしてスケジュールを決定するかという方法までは記述されていない．拡張ユースケース図ではこの部分が表現できず，その決定方法を依頼者に提案することができない.

このような問題に対して，アクティビティ図を用いたユースケースの詳細化を行う．スケジュールの決定方法は依頼者から要求が出されていないので開発者側から提案する形を取る．スケジュール決定の方法として次のような手順を取ることを考える.

1. 国際会議の開催日を入力する.
2. 過去の国際会議のスケジュールを参照する.
3. スケジュールを決定する.
4. スケジュールに問題がなければ終了．問題があった場合，変更項目を入力する.

これをアクティビティ図を用いて表現すると図8のようになる．業務(a)は他のアクタ(あるいはシステム)は存在しないので，swimlaneは2つであり，左側がアクタ(プログラム委員長)，右側がシステムである.

業務(c)においてはプログラム委員候補をどのようにして選ぶかという問題がある．この場合の対処

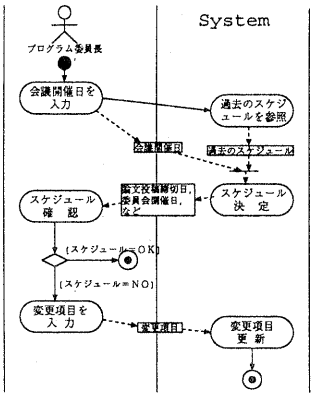

図 8: 業務(a)のアクティビティ図

法として「ある研究会の会員リストから選ぶ」ということを提案し，それをアクティビティ図に示す．また，依頼の方法は依頼状を郵送することで行うようにする．業務(c)のアクティビティ図は図9のようになる．

この業務ではアクタとして「プログラム委員長」以外に「委員候補」が存在する．従って，swimlaneは3区画で左から順にプログラム委員長，システム，委員候補である．

4 要求獲得支援

本稿では，ユースケース図，拡張ユースケース図，アクティビティ図の3つのモデルを順に依頼者へ提示し，要求獲得を行うことを考えている．

4.1 ユースケース図による要求獲得支援

開発者は業務ドキュメントから抽出されたユースケース図を依頼者に提示し，その図をもとに業務の確認を行う．ユースケース図は開発者が業務ドキュメントの内容を見落としていたり，依頼者が要求ドキュメントに書き忘れているような項目があった場合に対しての獲得支援が行える．

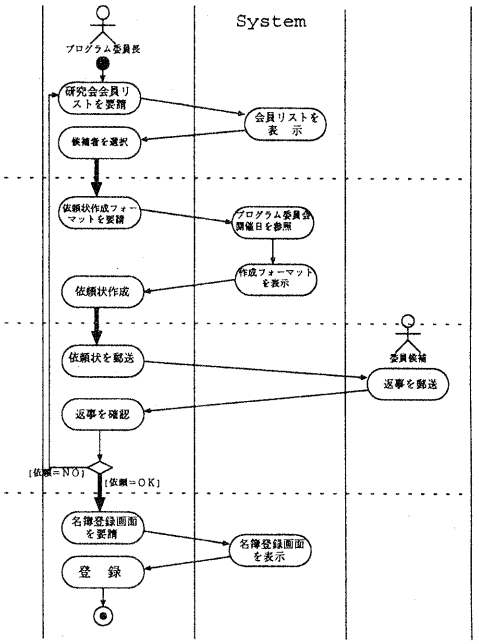

図 9: 業務(c)のアクティビティ図

4.2 拡張ユースケース図による要求獲得支援

拡張ユースケース図はユースケース図に業務シーケンスを追加している．開発者が考えた業務シーケンスを依頼者に提示し，そのシーケンスが正しいのか，誤っているのかの判定を依頼者に求めることで，業務シーケンスにおける依頼者と開発者との間の共通理解を得ることが可能である．

例えば，図7における「依頼」ユースケースに注目すると，例題においては依頼を断わられた場合の例外処理については触れられていない．断わられた場合に対する例外処理を知っておきたい開発者は，「もう一度，委員候補を選出する」という案を拡張ユースケース図に示し，それに対する依頼者の意見を聞くことで，この例外処理に対する依頼者の要求を獲得でき，共通理解を得ることが可能である．

また，図7では依頼を断わられると単純に候補を再選出するという拡張ユースケース図になっているが，委員の人数がある程度集まっていれば再選出を行わなくても良いのではないか，という新たな疑問

を浮かび上がらせることも可能であると考える．

4.3 アクティビティ図による要求獲得支援

アクティビティ図の目的は拡張ユースケース図で表現できない部分に対する要求獲得支援，システム化の境界に対する要求獲得支援である．

例えば，スケジュールの決定方法は拡張ユースケース図では表現できなかったので，アクティビティ図を用いてモデル化を行った．その結果，スケジュールの決定方法に対する開発者なりの考えが依頼者に提示でき，依頼者はその方法に対して依頼者の考えを開発者に伝えることができる．従って，スケジュールの決定方法に対して依頼者と開発者とが共通理解を得ることができる．

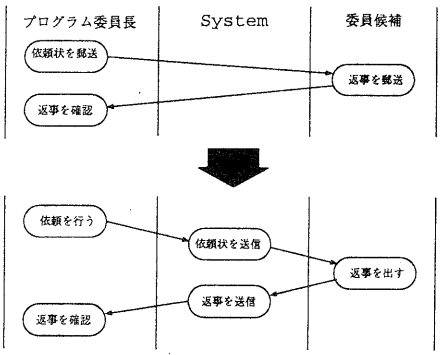

図 10: システム化領域の変更例

また，依頼を行う際は「依頼状を郵送する」ということにしているが，郵送では時間とコストがかかるので E-mail で処理するように依頼者が要求するかもしれない．そのような場合は図10のようにアクティビティ図を変更することで表現でき，システム化すべき業務を明確にすることが可能になる．

5 おわりに

システム開発を行う際，システム開発に必要な，あるいは重要な部分であるにも関わらず，依頼者から要求として出されていないという状況はよくある．この部分に対し，開発者は依頼者が理解できる形でこの部分を指摘し，それに対する依頼者の意見を引き出さなければならない．

本稿では，その方法としてシステムの動的側面に注目し，ユースケース図，拡張ユースケース図，アクティビティ図の3つのモデルによる要求のモデル化を通した開発者提案型による要求獲得支援方法について提案した．その結果，この方法を用いることより，徐々に依頼者と開発者とのシステムに対する共通理解を深めることが可能であり，例外処理，システム化範囲に対する要求獲得支援が可能であることを確認した．今後の課題として，動的なモデルから静的モデルへの変換方法，非機能的要求を考慮した要求分析，獲得方法などの検討が挙げられる．

参考文献

[1] Daniel M.Berry and Brian Lawrence: *IEEE Software, Requirements Engineering, pp. 26-33*, IEEE (1998)

[2] Graig Larman: *Applying UML and Patterns*, Prentice Hall(1998)

[3] *UML documentation, version 1.3*, Rational Software Corporation.
URL: http://www.rational.com/index.jtmpl

[4] 要求工学ワーキンググループ,
URL: http://www.selab.cs.ritsumei.ac.jp/ohnishi/RE/rewg.html

[5] 共通例題「国際会議のプログラム委員長の業務」,
URL: http://www.selab.cs.ritsumei.ac.jp/ohnishi/RE/problem.html