Vol.2014-SE-183 No.4
2014/3/19

REBOK に基づく要求管理教育支援ツール REMEST

山下智史[†1]　三塩花菜[†1]　掛下哲郎[†1]

我々は，要求工学知識体系（REBOK）に基づいて要求管理の基礎教育を支援するツール REMEST を開発している．REMEST は，astah* professional を使用して作成するマインドマップなどの成果物を自動的に検査し，学習者の進捗状況に応じた適切なアドバイスを示すことで，学習プロセスを支援する．本研究では，ソフトウェア要求に重点を置き，REBOK のプロセスのうち要求獲得に取り組む．要求獲得の過程で学習者が作成する成果物の形式や REMEST の中核となる成果物のチェックポイントに基づいて作成したサブルールのリスト（チェック項目リスト）を定義する．REMEST を開発する際には，成果物の検査時に求められる高水準操作を定義・開発した．現在は，成果物がサブルールを満たすか検査する機能の開発を進めており，成果物のテンプレートを検査する高水準操作が完成した．

A Requirement Management Education Support Tool REMEST based on REBOK

SATOSHI YAMASHITA[†1]　HANANA MISHIO[†1]　TETSURO KAKESHITA[†1]

We are developing a software tool REMEST to assist education of the basics of requirements management based on Requirements Engineering Body Of Knowledge (REBOK). REMEST automatically examines artifacts such as mind map created using astah* professional, and supports the learning process by showing proper advises in accordance with the progress of the learner. We focus on the software request, and work on the Requirement Acquisition process of REBOK. In requirements elicitation, we define form of the artifacts which a learner creates, and the list of subrules created from the checkpoints of the artifact. We develop high level operations in order to check consistency of the artifacts and combine them into REMEST. Currently, we are developing functions to check whether artifacts satisfy subrules, and the part of the function to inspect artifact template is completed.

1. はじめに

ソフトウェア開発において要求定義や要求管理は重要性が高く，プロジェクト成功の鍵を握っている[1,2]．そのため，IT 業界だけでなく大学においても，要求工学の教育が期待されている[3]．また，要求管理に対する取り組みが不十分，要求工学に対する啓蒙・教育が必要等の課題が挙げられている[1]．そこで，我々は，要求管理の基礎教育を支援するツール REMEST を企画・開発を行っている．REMEST は，チェンジビジョン社が開発している astah* professional[4]（以下，astah*と略記）のプラグインとして開発している．astah*はマインドマップや各種 UML 図等の成果物を作成・編集する機能を提供しており，要求の整理や管理を行える．REMEST は astah*を使用して作成された成果物を自動的に検査し，サブルールの合否に基づいて学習者の進捗状況を把握する．そして，学習者の進捗状況に応じた適切なアドバイスを示すことによって，学習プロセスを支援する．そのため，REMEST は，学習者が作成した成果物を監視し，学習者の進捗状況に応じて要求管理や成果物を作成する過程を効果的かつ段階的に指導できる．

我々は，REMEST がサポートする学習を，要求工学知識体系（REBOK）[5]に基づいて行えるようにする．REBOK とは，要求工学を理解し，活用するための手引きとなる要求工学の知識を実践の視点から整理し，体系化したものである．REBOK では，要求をビジネス要求，システム要求，ソフトウェア要求等に分類している．本研究では，ソフトウェア要求に重点を置き，REBOK のアプローチを通して生成される成果物のうち，ソフトウェア要求仕様書を作成するまでの過程を学習できるようにすることを目指す．また，REBOK における要求を定義するプロセスは，以下の 4 つに分類されている．本研究では，まず要求獲得に取り組む．要求管理の工程全般にわたって適用される要求の検証・妥当性確認・評価は，REMEST の検査機能にて，これをカバーする．

① 要求獲得
② 要求分析
③ 要求仕様化
④ 要求の検証・妥当性確認・評価

2. REBOK に基づいた要求獲得学習

REBOK における要求獲得は，顧客を含むステークホルダを明らかにし，会議やインタビューなどを通して要求を引き出す過程と技術に関する知識である．その過程は，以下に示す 8 プロセスからなる．本節では，これらのプロセスの学習に対する REMEST の支援について説明する．

① ステークホルダの識別
② 現状システムの理解
③ 現状システムのモデル化

†1 佐賀大学
Saga University

④　課題の抽出と原因分析
⑤　課題解決に向けたゴールの抽出
⑥　ゴールを達成する手段の抽出
⑦　実現すべきシステムのモデル化
⑧　要求の記述と詳細化

2.1 REMEST における学習の基本方針

REMEST を用いた演習では，最初に学習者に対してテンプレートファイルを配布することとし，テンプレートに定義した情報が揃っていることをサブルール 1 とする．演習では，学習者に astah*を使用してテンプレートを編集することで成果物を作成してもらう．成果物の形式はマインドマップとアクティビティ図（業務フロー図）である．学習者には，進捗状況に応じて REMEST が提示する指示に従い，サブルールの順番に沿って成果物を作成してもらう．マインドマップによる成果物を作成する場合，学習者は，必要に応じてトピックを追加し，トピックに事項を記入するという流れで成果物を作成する．

2.2 ステークホルダの識別

①ステークホルダの識別は，その後のプロセスにおいて情報を得る対象者，関係するシステム，法律，規約，習慣などをステークホルダとして識別し定義する工程である．REMEST においては，マインドマップにより図 1 に示す成果物を作成させることで，学習者にステークホルダを識別してもらう．作成される成果物は，表 1 に示すサブルールに基づいて検査する．

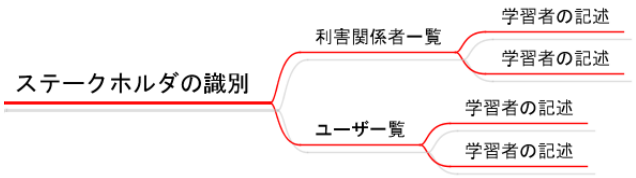

図 1　ステークホルダの識別に対する成果物
Figure 1　Mind map to identify stakeholders.

表 1　ステークホルダの識別のサブルール
Table 1　Sub Rules to identify stakeholders.

	サブルール
2	「利害関係者一覧」の配下に複数個のトピックがある．
3	「利害関係者一覧」の配下に追加されたトピックの記述は英数字のみ，または「トピック」,「トピック N」（N は番号）のみの文字列ではない．また，複数文字記入されている．
4	「ユーザー一覧」の配下に複数個のトピックがある．
5	「ユーザー一覧」の配下に追加されたトピックの記述は英数字のみ，または「トピック」,「トピック N」（N は番号）のみの文字列ではない．また，複数文字記入されている．

2.3 現状システムの理解・モデル化

②現状システムの理解は，現状システムを理解した成果物として，現状システムを説明するシナリオを作成する工程である．③現状システムのモデル化のプロセスでは，現状システムの理解を通じて得られたシナリオに基づき，現状システムのモデルを構築する．このプロセスのアウトプットは，システム責任者やステークホルダ等が現在の状況を共通に理解することが可能な System-as-is モデルである．

このプロセスに適用できる技術として，ビジネスモデリングがある．本研究では，ソフトウェア要求に重点を置くことから，まずはビジネスプロセスのみ行えばよいと考えた．そのため，REMEST においては，アクティビティ図（業務フロー図）を作成させることで，学習者にシステムの理解及びモデル化をしてもらう．作成された業務フロー図は，表 2 のサブルールに基づいて検査する．⑦将来システムのモデル化のプロセスもこのプロセスと同様に，業務フロー図を作成してもらい，表 2 に示すサブルールに基づいて検査する．⑦にあたるサブルールの番号は，36～51 である．

表 2　現状システムの理解・モデル化のサブルール
Table 2　Sub Rules for understanding and modeling the current system.

	サブルール
6	パーティション（登場人物）が（縦もしくは横方向に）複数ある．
7	開始ノードが 1 つだけある．
8	パーティションのラベルの記述内容が意味を持たない文字列でない．
9	アクションが 1 個以上作成されている．
10	アクション及びオブジェクトのラベルの記述内容が意味を持たない文字列でない．
11	不要な要素（業務フロー図の作成に不必要な要素）を作成していない．
12	図中に作成されているノードが全てフローにて繋がれている（独立しているノードがない）．
13	アクション・オブジェクト・開始ノードに対して繋がれているフローの数に間違いがない．
14	デシジョンノード・マージノード（分岐処理）がある場合，それに対して，正しくフローが繋がれている．
15	分岐処理があり，それがデシジョンの場合，デシジョンノード・マージノードから出るフローに対してガード条件が書かれており，その記述が意味を持たない文字列でない．
16	フォークノード（並行処理）がある場合，フォークノードに正しくフローが繋がれている．
17	フォークノード（並行処理）がある場合，フォークノードから出る全てのフローの先は，最終的に同じジョイン

	ノードに繋がる.
18	ジョインノードにフローが正しく繋がれている.
19	アクティビティ終了が 1 つ以上ある.
20	全てのパーティションに 1 つ以上のアクティビティがある.
21	開始ノードからアクティビティ終了までのフローが成立している.

2.4 課題の抽出と原因分析

④課題の抽出と原因分析は，現在の状況を説明するシナリオと現状システムを表すモデルに基づいて，解決すべき課題を抽出し，課題の原因分析を行う工程である．このプロセスの成果物として，課題を列挙すると共に，課題の依存関係や課題の影響を受けているステークホルダとの間の依存関係を示せばよいと考えられる．そこで，REMEST においては，各課題に対し，問題，影響を受ける人，問題の結果，解決策の利点の 4 項目[6]を学習者に記述させる．そのため，図 2 に示す形式でマインドマップを作成させることで，学習者に課題を列挙してもらう．また，課題とステークホルダ間の依存関係や課題間の依存関係は，トピック間リンクを用いて示す（図 2, 3）．作成された成果物は表 3 に示すサブルールに基づいて検査する．

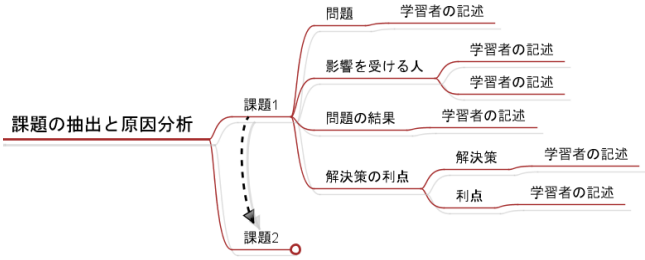

図 2　課題の抽出と原因分析に対する成果物
Figure 2　Mind map for problem extraction and cause analysis.

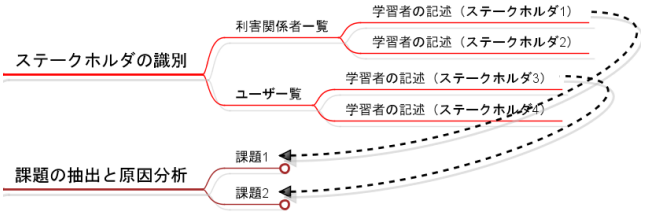

図 3　ステークホルダと課題の依存関係の示し方
Figure 3　Dependency between stakeholders and issues.

表 3　課題の抽出と原因分析のサブルール
Table 3　Sub Rules for problem extraction and cause analysis.

	サブルール
22	「課題の抽出と原因分析」の配下の新しく追加したトピックには「課題 N」（N は番号）と記述され，番号に重複がない.
23	全ての「課題 N」（N は番号）の配下にテンプレートで定義されている問題記述書の詳細項目 4 つが記述されたトピックがある．（「解決策の利点」の配下には「解決策」と「利点」のトピックがある）.
24	全ての課題に対して，テンプレートで定義された詳細項目 4 つに関するトピックの配下に新しいトピックが追加されている.
25	全ての課題に対して，テンプレートで定義された詳細項目 4 つに関するトピックの配下に追加されたトピックの記述は，英数字のみ，または「トピック」,「トピック N」（N は番号）のみの文字列ではない．また，複数文字記入されている.
26	前ステップ（ステークホルダの識別）にて挙げられたステークホルダが，「課題の抽出と原因分析」の配下のトピック（課題 N）と 1 つ以上関連付けが行われている.
27	列挙された課題に対して，課題間で 1 つ以上関連付けが行われている.

2.5 課題解決に向けたゴールの抽出

⑤課題解決に向けたゴールの抽出は，課題解決に向けて達成，または維持すべきゴールを抽出する工程である．このプロセスのアウトプットは，課題を解決することによって達成すべきゴール及び維持し続けなければならない状況の定義と，課題と課題を解決することによって達成されるゴールとの間の依存関係である．そのため，このプロセスの成果物として，達成すべきゴールと維持すべき状況（ゴール）を列挙し，前プロセスにて列挙した課題とゴールとの依存関係を示せばよい．REMEST においては，マインドマップにより図 4 の下側に示す形式にて作成させることで，達成すべきゴールと維持すべき状況を列挙してもらう．また，前プロセスにて列挙した課題とゴールとの間の依存関係はトピック間リンクにより繋ぐことで示す．作成された成果物は，表 4 に示すサブルールに基づいて検査する．

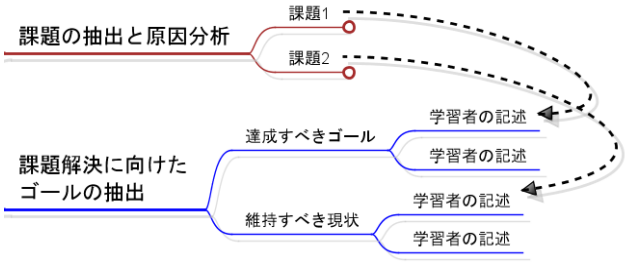

図 4　課題解決に向けたゴールの抽出に対する成果物
Figure 4 Mind map for goal extraction toward problem solution.

表 4　課題解決に向けたゴールの抽出のサブルール
Table 4 Sub Rules for goal extraction toward problem solution.

	サブルール
28	「達成すべきゴール」の配下に複数個のトピックがある.

29	「達成すべきゴール」の配下に追加されたトピックの記述は英数字のみ，または「トピック」，「トピック N」（N は番号）のみの文字列ではない．また，複数文字記入されている．
30	「維持すべき現状」の配下に複数個のトピックがある．
31	「維持すべき現状」の配下に追加されたトピックの記述は英数字のみ，または「トピック」，「トピック N」（N は番号）のみの文字列ではない．また，複数文字記入されている．
32	前ステップ（課題の抽出と原因分析）にて挙げられた全ての課題が，「達成すべきゴール」または「維持すべき現状」の配下のトピックと抜けなく関連付けが行われている．

2.6 ゴールを達成する手段の抽出

⑥ゴールを達成する手段の抽出は，現状のシステムの課題を解決し，ゴールが達成された状況，またはゴールが維持されている状況を実現する手段を抽出する工程である．このプロセスのアウトプットは，達成すべきゴールまたは維持すべき状況と，それらを実現する手段との関係を表した文書（例：ゴールモデル）や将来の状況を説明するシナリオである．そのため，このプロセスの成果物として，達成すべきゴールまたは維持すべき状況（ゴール）を実現する手段を列挙し，前プロセスにて列挙された各ゴールと手段との関係を示せばよい．そこで，REMEST では，マインドマップにより図 5 の下側に示す形式にて作成することで，ゴールを達成する手段を列挙させる．また，前プロセスにて列挙したゴールと手段との間の依存関係はトピック間リンクを用いて示す．作成された成果物は，表 5 に示すサブルールに基づいて検査する．

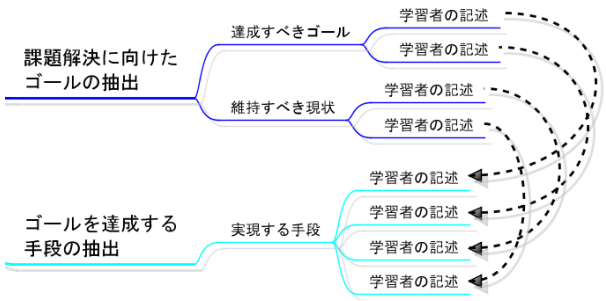

図 5　ゴールを達成する手段の抽出に対する成果物
Figure 5　Mind map to extract means to achieve goal.

表 5　ゴールを達成する手段の抽出のサブルール
Table 5　Sub Rules to extract means to achieve goal.

	サブルール
33	「実現する手段」の配下に複数個のトピックがある．
34	「実現する手段」の配下に追加されたトピックの記述は英数字のみ，または「トピック」，「トピック N」（N は番号）のみの文字列ではない．また，複数文字記入されている．
35	前ステップ（課題解決に向けたゴールの抽出）にて挙げられた全てのゴールが，「実現する手段」のトピックと抜けなく関連付けが行われている．

2.7 要求の記述と詳細化

⑧要求の記述と詳細化は，要求を現状システムと，将来システムのモデルと，将来システムのモデルの差分として定義し，記述する工程である．このプロセスのアウトプットは，要求の候補の集合を記述した文書である．REMEST においては，利害関係者・ユーザのニーズとそのニーズを満たす基本要件[6]を記述する．そのため，マインドマップにより図 6 に示す形式にて作成することで，要求を記述させる．また，ニーズと基本要件間の依存関係は，トピック間リンクを用いて示す．作成された成果物は，表 6 に示すサブルールに基づいて検査する．

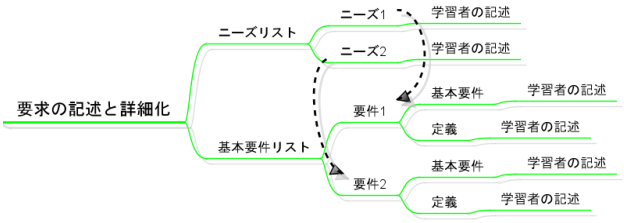

図 6　要求の記述と詳細化に対する成果物
Figure 6 Mind map for requirement description and refinement.

表 6　要求の記述と詳細化のサブルール
Table 6 Sub Rules for requirement description and refinement.

	サブルール
52	「ニーズリスト」の配下の新しく追加したトピックには「ニーズ N」（N は番号）と記述され，番号に重複がない．
53	全ての「ニーズ N」（N は番号）の配下に複数個のトピックがある．
54	「ニーズリスト」の配下に追加されたトピックの記述は英数字のみ，または「トピック」，「トピック N」（N は番号）のみの文字列ではない．また，複数文字記入されている．
55	「基本要件リスト」の配下の新しく追加したトピックには「要件 N」（N は番号）と記述され，番号に重複がない．
56	全ての「要件 N」（N は番号）の配下にテンプレートで定義されている基本要件の詳細項目 2 つが記述されたトピックがある．（「要件 N」の配下には「基本要件」と「定義」のトピックがある）
57	全ての要件に対して，テンプレートで定義された詳細項目 2 つに関するトピックの配下に追加されたトピックの記述は，英数字のみ，または「トピック」，「トピック N」（N は番号）のみの文字列ではない．また，複数文字記

	入されている.
58	全てのニーズから抜けなく基本要件に対して関連付けが行われている.

3. REMEST の機能

REMEST の中核は，要求管理の成果物を作成する際の確認事項をルール化したチェック項目リストである．チェック項目リストは要求管理ステップに対応するメインルールから構成され，各メインルールは複数のサブルールを含む．REMEST が提供する機能[7]は下記の 3 つである．

(1) チェック項目リストを管理する機能

学習者が成果物を作成する過程において，astah*プロジェクトから取得した成果物のデータを基に，2 節に記述したサブルールを番号順に検査し，合否を判定する．条件を満たしているサブルールは充足とし，そうでなければ不十分とする．ただし，不十分となったサブルールより先のサブルールの合否結果は全て不十分にする．チェック項目リストの更新に伴う各ルールの合否判定は自動化している．

(2) 学習者の進捗状況を把握する機能

チェック項目リストから学習者の進捗状況を特定し，必要な情報を取得する．進捗状況とは，学習者が取り組んでいる部分がチェック項目リスト中のどのメインルールにあたるか，さらにそのメインルールのどのサブルールにあたるかを表すものである．また，学習者が誤った部分に取り組んでいた場合，それを指摘するために学習者が現在編集した部分を特定し，現在取り組むべき部分より先の部分に取り組んでいないかを判断する．

(3) 成果物を作成する際に必要な情報を表示する機能

REMEST では，要求管理についての説明や成果物を作成する際に必要となる指示･アドバイス等を学習者に表示する．表示内容は以下の通りである．

- 学習者が現在取り組んでいる作業項目を表示する
- 学習者が現在取り組んでいる工程の説明を表示する
- 学習者が次に行う操作に対する説明を表示する
- 間違いに対するアドバイスを表示・挿入する
- 誤った部分への取り組みに対する警告を表示する
- 学習者の達成状況及び進捗状況を表示する

4. REMEST の開発

前節のことを踏まえ，REMEST の開発を進めている．ツールの実装は，Java と astah* API[8]により行っている．

4.1 REMEST の構成

REMEST の全体構成は図 8 に示す通りである．REMEST の設計には，MVC モデルと Observer パターンを適用している．図 8 のクラス図の内，緑の枠で示したクラスは View 部，青の枠で示したクラスは Model 部にあたる．これにより，系統的なオブジェクト指向設計を行い，修正や拡張等を行い易くしている． Model 部は，チェック項目リストや

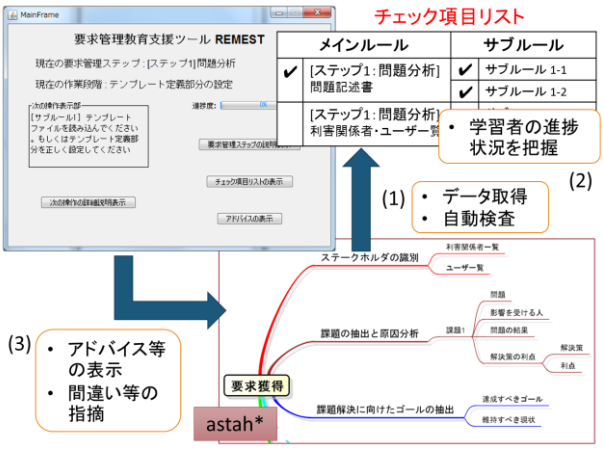

図 7　REMEST の仕組み

Figure 7　Mechanism of REMEST.

メインルール，サブルール，成果物の図のデータを表現しており，それぞれに対応するクラスは CheckItemList，MainRule，SubRule，DiagramData である．個別のメインルールやサブルール，成果物のデータは，これらのクラスのサブクラスで定義している．

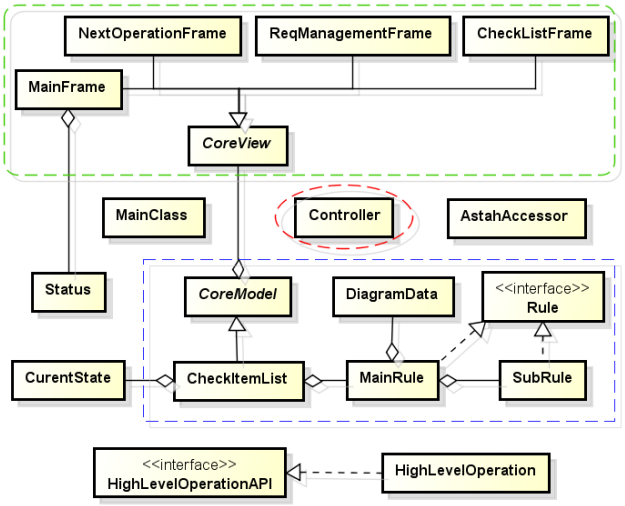

図 8　REMEST のクラス図

Figure 8　Class diagram of REMEST.

REMEST では，Model 部の状態更新処理を自動化しており，全て AstahAccessor から開始する．状態更新処理は，図 7 の(1)，(2)にあたり，その結果を View 部の表示に反映する．これは，astah* API の ProjectEventListener[9] を利用して実装しており，REMEST の起動時と astah*プロジェクトを開いた時，成果物の編集時に自動で成果物の検査を行う．これにより，REMEST は演習の開始時及び再開時から学習者の進捗状況を把握でき，演習中も常に成果物を自動で監視し，学習者の進捗状況に合った情報を提示できる．

REMEST では，マインドマップ の階層構造や特定の項目に対応するトピックの有無の検査等の高水準操作[10]を

行う．しかし，実装に使用している astah* API が提供する機能は，基本操作のみである．そのため，両者の間で大きな意味的ギャップが生じている．そこで我々は，REMEST の高水準操作を実現する API（図 8 下部）を定義し，高水準操作の開発を行った. これを astah*と REMEST の中間層とすることで，この問題を解消している． REMEST の高水準操作は 13 個あり，3 種類に分類している．それらについて，種類ごとに 4.2 節以降で説明する．

4.2 サブルール合否判定

サブルール合否判定を行うために 4 つの高水準操作を定義した. これらの高水準操作は, マインドマップを検査し，成果物が正しく作成されていることを確認する．

astah* API はルートトピック，子トピック，親トピックの取得といった基本操作を提供している．しかし，トピックデータ単体では，トピックの位置を特定できない．検査の際に，必要なトピックを取得するためには，ルートトピックから子トピックの取得を繰り返し行い，目的のトピックの探索を行わなければならない．探索時のキーになる情報は，トピックに記述された内容（以下，トピックの名前と表記）のみである．そこで，ルートトピックから指定されたトピックに至るまでに辿っていく全てのトピックの名前を探索時のキーとする．つまり，図 12 に示すように，必要なトピックを指す指定項目名とルートトピックから目的のトピックまでに至るトピックの名前を全て格納した探索時のキーリストを用意する．そして，これらを基に目的のトピックの探索を行う．

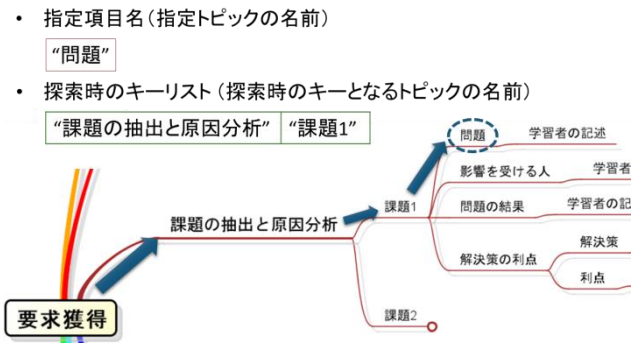

図 9　トピックの探索方法

Figure 9　Search method of the topic.

以下に，この判定に関する 4 つの高水準操作を示す．これらの API の引数は，共通してルートトピック，指定項目名，指定されたトピックを探索するために辿るトピック一覧格納リストである．

(1) 指定された項目に対応するトピックの有無を確認する

この操作では，指定項目に対応するトピックがあるか探索し，その有無を判定すると共に，戻り値として返す．

(2) 指定された項目に対応するトピックを取得する

この操作では，指定項目に対応するトピックを探索し，戻り値として返す. 目的のトピックがない場合は null を返す．

(3) 指定された項目に対応するトピックの子トピック群を取得する

この操作では，指定項目に対応するトピックの子トピック群を取得すると共にリストに格納し，戻り値として返す．目的のトピックが無い場合は，空のリストを返す．

(4) 指定された項目の子トピックの数を取得する

この操作では，上記(3)を使用して，指定項目に対応するトピックの子トピックを取得し，その個数を返す．

4.3 エラー箇所及びエラーの種類の判定

エラー箇所及びエラーの種類の判定を行うために以下に示す 3 つの高水準操作を定義した．これらの高水準操作は，成果物に対してエラーの原因となっているトピックやそのトピックに対するエラーの種類を特定する．ここでのエラーとは, 主にマインドマップの構造的な間違いである．

これらの高水準操作においても，4.2 節と同様の方法で目的のトピックを探索する．高水準操作の共通の引数は，前節と同様にルートトピック，指定項目名，指定されたトピックを探索するために辿るトピック一覧格納リストである. 戻り値は全てエラーの原因となっているトピックの ID とエラーの種類に対応するエラーID を結合した文字列である. トピック ID は astah* API が提供する機能により文字列として取得できる．エラーID は，エラー対象トピックがどのサブルールに関わるものであり，以下に示す API にてチェックする 10 項目のどれに該当したものかを判別できるものとしている．この判定に関する高水準操作は，3 節(3)の機能のうち，間違いに対するアドバイスを表示・挿入する機能に使用する．

以下に，我々が定義した 3 つの高水準操作を示す．

(5) 学習者の記述が不適切なトピックを特定する

この操作では，4.2 節(3)の高水準操作を使用し，指定項目に対応するトピックの子トピック（学習者が記述を行うためのトピック群）を取得する．取得したトピックに対し，学習者の記述が英数字記号のみからなる不適切なものでないか確認する．

(6) 各種番号に対応するトピックに間違いがあるものを特定する

対象となる子トピック群を取得して，以下に示す 4 項目のチェックを行う．各種番号に対応するトピックとは，図 2 の「課題 1」,「課題 2」や図 6 の「ニーズ 1」,「ニーズ 2」,「要件 1」,「要件 2」にあたるものである．この API の引数は，共通のものに加え，番号の種類がある．これは，課題，ニーズ，要件といったものにあたる．この処理では，下記の項目について検査するために，番号にあたるトピックの名前を番号の種類の部分と数字の部分に分割して検査を行う．

- 番号に重複がないか確認する
- 番号が連続しているか確認する

- 番号が振られていないトピックがないか確認する
- 番号に対応するトピックが種類ごとに 2 つ以上あるかを確認する

(7) 指定された項目の子トピックに対して，各種エラーチェックを行う

指定項目に対応するトピックの子トピック群を取得して，以下に示す 5 項目のチェックを行う．

A) 項目名が間違っていないか（項目の有無を）確認する
B) 余分なトピックがないか確認する
C) 各項目が重複していないか確認する
D) 各項目が不足していないか確認する
E) 各項目の順番が正しいか確認する

この API の引数は，共通のものに加え，指定の子トピックにあるべき項目一覧がある．これは，図 2 を基に例を挙げると，「課題 2」のトピックの配下に「課題 1」の配下にある 4 項目と同じものがあるかを確認する場合等に使用するものである．この例の追加の引数は，「課題 1」の配下にある 4 項目を格納したリストである．A)～D)のチェック処理では，あるべき項目のうち，無い項目と有る項目を特定する．また，あるべき項目に該当しないエラーの原因となるトピックを特定し，それらがどのチェックのエラーにあたるかを判定する．

4.4 学習者の誤った部分への取り組みの判定

学習者が誤った部分に取り組んでいることを判定するために 6 種類の高水準操作を定義した．これらの操作は，学習者が編集した箇所を特定し，本来取り組むべき範囲よりも先の部分に取り組んでいないか確認するためのものである．

編集個所を特定する際には，astah* API が提供する機能により，利用者が行った編集に対して，どの要素にどの操作が行われたかを示す更新イベントを保持するプロジェクト編集ユニットを取得できるため，これを利用する．astah*では，利用者が編集した要素だけでなく，その要素と関係のある要素全てに対し更新イベントが発生する．そのため，複数のプロジェクト編集ユニットが配列に格納された状態で返される. 更新イベントにおける操作の種類は，追加, 削除, 修正の 3 つに分けられるが, 修正の操作には，トピックの親子関係が変更される場合と変更されない場合がある．親子関係が変更される編集は「トピックの移動」とする．各要素に対する更新イベントの発生順序は，操作の種類ごとに，規則性があるため，学習者がどの要素に対し，何の操作を行ったかを特定できる．実際の判定は，学習者が編集したトピックの親トピックを基に，学習者が現在取り組んでいる部分がどのメインルールで取り組むべき範囲かを特定する．そして，学習者が編集した箇所が現在取り組んでいるメインルールより先のメインルールで取り組むべき範囲であれば，誤った取り組みと判定する．

この判定に関する 6 つの高水準操作を以下に示す．学習者の誤った部分への取り組みを判定する手順は，以下に示す操作の順番に沿って行う．

(8) 学習者が行った操作が undo 機能によるものか判定する

astah*では，元に戻す（Windows ショートカットキー：Ctrl + z）を行った場合も更新イベントが発生する．そのため，undo 機能使用時も，プロジェクト編集ユニットが取得されてしまうため，学習者の誤った部分への取り組みを特定する処理を行わないようにする必要がある．この操作の引数は，プロジェクト編集ユニット配列と擬似データのリンクリストである．

元に戻す機能を使用した場合，更新イベントは一つ前に行った編集で更新された要素の順番と逆の順番で同じ要素に対して発生する．そのため，取得したプロジェクト編集ユニットから，次に元に戻す機能を使用した場合に取得されるプロジェクト編集ユニットを表す擬似的なデータを事前に作成できる．そこで，元に戻す機能が使用されたか否かは，取得されたプロジェクト編集ユニットと擬似データとの比較により判定する．元に戻す機能を連続で使用した場合は, 2 回目の更新イベントは, 2 つ前に行った編集で更新された要素の順番と逆の順番で同じ要素に対して発生する．3 回目以降も同様である．そのため，作成した疑似データをスタックとキューの両方の操作ができるリンクリストに格納する．これにより，元に戻す機能を連続使用されても対応できるようにしている．

(9) 学習者が行った操作の番号を取得する

この処理では，学習者が行った操作が追加・削除・修正・トピックの移動のいずれかをプロジェクト編集ユニットのデータを基に特定する．この操作の引数は，プロジェクト編集ユニット配列と一操作前のマインドマップの状態を記録したトピックリストである．戻り値は，追加（0）・修正（1）・削除（2）・トピックの移動（3）の各操作に対応する操作番号である．

修正の操作について，トピックの親子関係が変更される場合か否かは，一操作前と現在のマインドマップの状態を比較することによって判別する．マインドマップの状態が同じであれば，学習者が行った操作はトピックの移動である．

(10) 学習者が編集したトピックの親トピックのトピック ID を取得する

学習者が削除操作を行った場合，編集したトピックが無くなっているため, それを特定できないといった問題がある．そこで，学習者が編集したトピックの親トピックで代用する．この操作の引数は，プロジェクト編集ユニット配列と(2)の操作の戻り値にあたる操作番号である．

(11) トピック ID を基に対応するトピックを取得する

対象となるマインドマップの全てのトピックを取得し，ID を基に対応するトピックを探索し，それを返す．この操作

は，(12)および(13)の操作にて使用する．この操作の引数はトピック ID である．

(12) 対象トピックの位置番号を特定する

この高水準操作は再帰関数であり，処理中に(4)の操作を使用する．位置番号の割り当てを図 10 に示す．この操作の引数はトピック，対象トピック ID，探索中トピックの位置番号である．最初は，ルートトピックを引数とし，子トピックを取得後，そのトピックを引数として，操作の呼び出しを再帰的に繰り返す．これにより，深さ優先探索にてマインドマップを辿り，対象トピック ID に対応するトピックを探索すると共に対象トピックの位置番号と特定する．

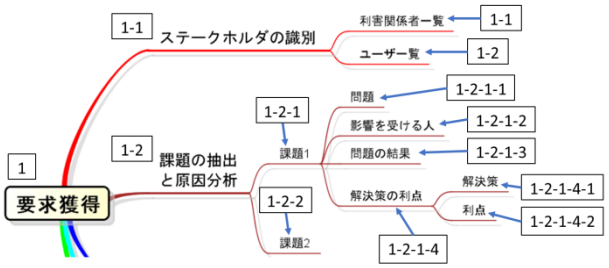

図 10　位置番号の割り当て

Figure 10　Allocation of the position number.

(13) 学習者が現在取り組んでいる部分がどのメインルールで取り組むべき範囲かを特定する

この操作の引数は，学習者が編集したトピックの親トピックの ID とそのトピックの位置番号である．サブルール 1（テンプレートの設定）で取り組む範囲は，ルートトピックから特定の階層までであり，その階層は，ツリーの位置によって異なる．そのため，この操作では，学習者が編集したトピックがある位置を特定し，取り組んでいる部分がメインルール 1 の範囲かを判定する．その際に(5)の操作で取得したトピックの位置番号を基に判定する．学習者が取り組んでいる部分がメインルール 1 の範囲でなければ，メインルール 2 以降のどの範囲か判断する．

4.5 項目間の関連付け及び業務フロー図の検査

項目間の関連付けの検査は，対象となるトピックにトピック間リンクが正しく繋がれているかを検査する．astah* API では，対象のトピックに繋がれているトピック間リンクやトピック間リンクのリンク元とリンク先のトピックを取得できるため，これを利用して実装する．

業務フロー図の検査では，作成手順に沿って正しく作成されているか検査する．astah* API では，アクティビティ図にて作成できる各ノードやパーティション，ノードに対して入るフローと出るフローを取得できるため，これを利用して実装する．

5. まとめと今後の課題

我々は，REBOK に基づいて要求管理の基礎教育を支援するツール REMEST の企画・開発を行っている．今回，REBOK のプロセスのうち要求獲得に取り組み，学習者が作成する成果物の形式やチェック項目リストの定義，必要な機能の設計・開発を行った．これにより，近年注目が高まっている REBOK に基づいて，要求管理の基礎を学習者の進捗状況や理解度レベルに合わせた教育支援を行える目途を立てた．今後の研究課題として，主に以下の項目が挙げられる．

(1) REMEST の評価実験

評価実験では，演習問題を作成し，REMEST を使用した演習を行う．評価実験を通じて得られたコメントなどを基にツールの修正及び改良を行う．また，ログを記録する機能を開発し，収集したログを基に学習者の学習過程を分析する．評価実験の実施に向けて，表示する説明文の作成や REMEST のユーザインターフェースの改良，下記の未実装機能やログ機能の実装を行う．

(2) 未実装の検査機能の開発

トピック間の関連付けを検査する機能とアクティビティ図（業務フロー図）を検査する機能を実装する．

(3) 未実装の支援機能の開発

3 節(3)にて紹介した機能の誤った部分への取り組みに対する警告を表示する機能と間違いに対するアドバイスを表示・挿入する機能の実現方法は検討済みであるが，今後，設計・実装を行う必要がある．

参考文献

1) 鎌田真由美: 特集要求工学 1 要求工学の現状と課題，情報処理，Vol.49，No.4，pp.347-356 (2008).
2) @IT 情報マネジメント：みんなが悩む要求管理（1），http://www.atmarkit.co.jp/farc/rensai/re_mgt01/re_mgt01.html
3) 要求工学:Requirements Engineering（月刊ビジネスコミュニケーション），第 10 回 要求工学の課題，http://www.bcm.co.jp/site/2005/2005-08/05-yokyu-08/05-yokyu-08.html
4) astah* - 最も身近なソフトウェア開発設計支援ツール，http://astah.change-vision.com/ja/
5) 一般社団法人 情報サービス産業協会 REBOK 企画 WG 編，要求工学知識体系 第 1 版，近代科学社(2011).
6) ディーン・レフィングウェル，ドン・ウィドリグ: ソフトウェア要求管理 新世代の統一アプローチ，株式会社ピアソン・エデュケーション (2002).
7) 山下智史，掛下哲郎: astah* professional を活用した要求管理教育支援ツール REMEST，情報処理学会ソフトウェア工学研究会，2013-SE-179(6)，pp.1-8(2013).
8) astah* API 利用ガイド，http://members.change-vision.com/javadoc/astah-api/6_6_3/api/ja/doc/index.html
9) astah* API JavaDoc，http://members.change-vision.com/javadoc/astah-api/6_6_3/api/ja/doc/javadoc/index.html
10) 山下智史，掛下哲郎: 要求管理教育支援ツール REMEST における高水準操作の設計，電気関係学会九州支部連合大会，06-2A-02(2013).