Vol.2013-SE-179 No.6
2013/3/11

astah* professional を活用した要求管理教育支援ツール REMEST

山下智史[†1]　　掛下哲郎[†1]

本研究では，要求管理の基礎教育を支援するためのソフトウェアツール REMEST を企画・開発を行う．REMEST は astah* professional のプラグインとして開発している．REMEST の中核部分は，MindMap などの成果物のチェックポイントから作成したサブルールのリスト（チェック項目リスト）である．REMEST は astah* professional を使用して作成した成果物を自動的に検査し，サブルールの状態に基づいて，学習者の進捗状況を把握する．このような学習者の行動の自動監視は，astah* API によって提供されているプロジェクトイベントリスナーを利用して開発している．REMEST は，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．

REMEST: A Requirement Management Education Support Tool utilizing astah* professional.

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

We develop a software tool REMEST to assist education of the basics of requirements management in this paper. REMEST is designed as a plug-in of astah* professional. Core of REMEST is the list of subrules created from the checkpoint of the artifacts such as mind map. REMEST automatically examines artifacts created using astah* professional, and grasps the progress of the learner based on the status of the subrules. Such automatic monitoring of the learner's behavior is developed by utilizing Project Event Listener provided by astah* API. REMEST supports the learning process by showing proper advises in accordance with the progress of the learner.

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[1,4]．これらの課題に対し，本研究では，要求管理の基礎について学習できるツール REMEST を提案する．我々はチェンジビジョン社が開発している astah* professional[5]（以下，astah*と略記）に着目した．astah*には MindMap や要求図などの図を作成する機能があり，要求の整理や管理を行える．また，プラグインを開発することで機能拡張ができる．そこで astah*を活用して REMEST を開発する．

要求管理は，システムに対する要求を引出し，整理し，文書化するための系統的なアプローチであり，以下の 6 つのステップで行うことができる[6]．ステップ 1～5 は要求を定義するスキルに対応し，ステップ 6 はシステム構築時の要求を管理するスキルに対応している．

① 問題分析　　② 顧客ニーズの理解
③ 開発構想書の作成　　④ 開発範囲の管理
⑤ 基本要件の詳細化　　⑥ 開発への移行と変更管理

astah*では，MindMap や UML により要求分析を行い，SysML の要求図と要求テーブルにより要求管理を行うことができる．そこで，要求管理ステップと astah*との関連について検討した．astah*による要求分析はステップ 1，2 をカバーし，要求管理はステップ 3 と 5 を部分的にカバーしていると考えられる．また，開発構想書など，ステップ 1～4 の成果物を MindMap により作成でき，ステップ 5 の成果物を要求図により作成できると考えた．そこで，本研究では，要求管理の内，ステップ 1～5 の教育上重要な部分に焦点を当てる．図 1～5 に astah*により作成した成果物の例を示す．

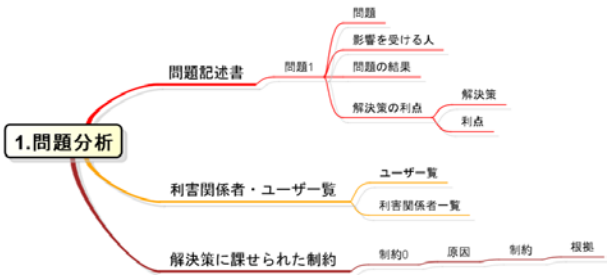

図 1　ステップ 1 におけるテンプレート作成例

Figure 1　Template Example at Step 1.

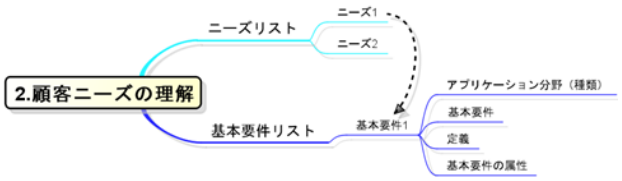

図 2　ステップ 2 におけるテンプレート作成例

Figure 2　Template Example at Step 2.

†1 佐賀大学
Saga University

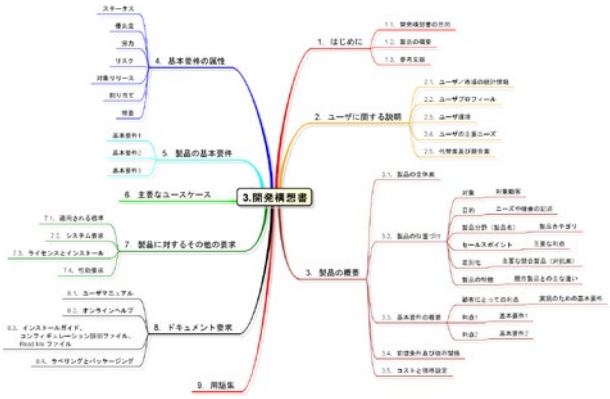

図 3　ステップ 3 におけるテンプレート作成例
Figure 3　Template Example at Step 3.

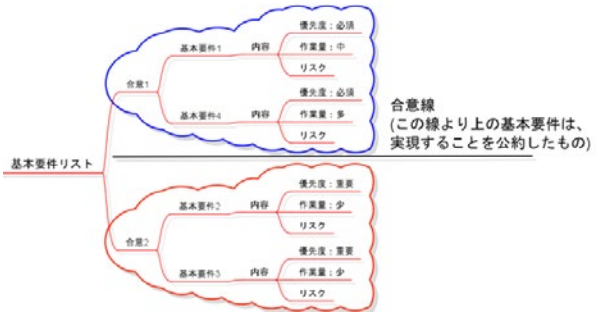

図 4　ステップ 4 におけるテンプレート作成例
Figure 4　Template Example at Step 4.

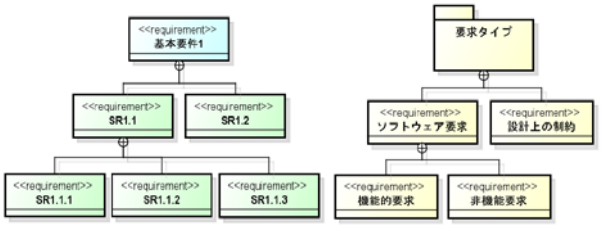

図 5　ステップ 5 におけるテンプレート作成例
Figure 5　Template Example at Step 5.

astah*では，各ステップの成果物間で関連付けを行える．同一の MindMap 内であれば，トピック間リンクで行うことができ，別々の MindMap 及び MindMap と要求図等の他の図同士では，ハイパーリンクを設定することで行える．また，関連付けを介して対応部分に移動できる．これにより，成果物間の関係も追跡できる．また，ニーズ，基本要件，ソフトウェア要求の 3 つの要求タイプは，要求図により階層的に管理できる．しかし，現状の astah*の標準機能では，MindMap と要求図の連帯は十分できない．また，成果物間の整合性や必要項目の有無のチェックができない等，要求管理の教育に利用するには不十分な点がある．

astah*は，プラグインを開発することで，astah*が提供している拡張ポイント（「独自メニューの追加」と「拡張ビューに独自のタブの追加」の 2 つ）に対して機能拡張を実現できる[7]．プラグインの機能は，Java と astah* API により実装できる[8]．astah* API は，astah*のプロジェクトファイル（以下，astah*プロジェクトと表記）へのアクセス等ができる ProjectAccessor[9]やプロジェクト内のモデル・図要素等へのアクセス機能がある．また，astah* API は，イベントリスナー（ProjectEventListener[9]）を提供している．これにより，astah*プロジェクトの開始，終了，編集時に発生するイベントを捕捉して任意の処理を行える．また，イベントで更新された要素等に関する情報を取得できる．

以上のことから本研究では，REMEST を astah*のプラグインとして開発し，astah*を利用した要求管理に対する教育支援を可能にする．REMEST は，学生や新入社員を対象とし，要求管理ステップ 1～5 の中で MindMap と要求図の関連，成果物間の整合性を自動点検することを最終目標としている．それにより，ニーズ，基本要件，ソフトウェア要求を記述する一連の流れを学習できる．この目標に向けて，当面は要求管理ステップ 1（問題分析）についての学習を目標に REMEST の開発を行っている．REMEST によって，学習者が作成しているステップ 1 の成果物を監視し，学習者の進捗状況に応じて要求管理や成果物の作成過程を効果的かつ段階的に指導する．

2. REMEST の要件定義と使用方法

REMEST の開発にあたって，要求管理の教育に必要な機能を企画検討し，それを基に要件定義を作成した．

2.1 REMEST の構想

要求管理の教育を行う際，学習者の進捗状況や理解度に合わせたアドバイスや指示を与えることが重要だと考えた．そこで REMEST では，要求管理ステップの成果物を作成する際のチェックポイントから作成したルールのリストであるチェック項目リストを利用して学習者の進捗状況の把握を行う．チェック項目リストには，成果物に対するチェック項目の最小単位であるサブルールと作業段階毎にサブルールをグループ化したメインルールを設けている．ルールをグループ分けすることで，進捗状況の管理等を容易にし，学習者が現在の作業段階をすぐに判断できる．また，メインルール単位で適切な情報を学習者に提示できる．チェック項目リストでは，ルールが満たされるとその項目に対してチェックを付ける．このルールがどこまで満たされているかにより学習者の進捗状況及び理解度レベルを把握する．

上記の考えを基に REMEST は 3 つの機能を提供する．1 つ目は表 1 に示すチェック項目リストを管理する機能である．2 つ目は学習者の進捗状況を把握する機能である．3 つ目は成果物を作成する際に必要な情報を表示する機能である．以下の節にて，これらの機能について説明する．

2.2 ステップ 1 に対するチェック項目リストの定義

ステップ 1 の各メインルール及びサブルールを表 1 に示す．REMEST を用いた演習を行う際，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．テンプレートは，図

表 1　要求管理ステップ 1 のチェック項目リスト
Table 1　Check Item List of Requirement Management Step 1.

メインルール		サブルール	
1	テンプレート定義部分の設定	1	テンプレートに定義した情報が全て揃っている.
2	問題記述書の作成	2	問題記述書の配下にトピックを追加している．記述は「問題 N」（N は番号）とし，番号に重複がない.
		3	全ての問題 N の配下に問題記述書における 4 つの詳細項目を記述したトピックがある.
		4	問題定義書における 4 つの詳細項目に対応するトピックが追加されている.
		5	問題定義書における 4 つの詳細項目が記述されている．（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.）
3	利害関係者・ユーザ一覧の作成	6	ユーザ一覧の配下に複数個のトピックが追加されている.
		7	ユーザ一覧が記述されている．（記述はサブルール 5 と同様）
		8	利害関係者一覧の配下に複数個のトピックが追加されている.
		9	利害関係者一覧が記述されている．（記述はサブルール 5 と同様）
4	解決策に課せられた制約の作成	10	解決策に課せられた制約の配下にトピックを追加している．記述は「制約 N」（N は番号）とし，番号に重複がない.
		11	全ての制約 N の配下に，原因について記述するためのトピックが 1 つだけ追加されている.
		12	全ての原因について記述するトピックの配下に，制約について記述するためのトピックが 1 つだけ追加されている.
		13	全ての制約について記述するトピックの配下に，根拠についての記述するためのトピックが 1 つだけ追加されている.
		14	原因について記述されている．（記述はサブルール 5 と同様）
		15	制約について記述されている．（記述はサブルール 5 と同様）
		16	根拠について記述されている．（記述はサブルール 5 と同様）

1 の MindMap である．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．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していく．この過程で表 1 の上から順に各サブルールを検査し，合否を判定する．その際には，astah*プロジェクトから成果物（MindMap）のデータを取得する．条件を満たしているサブルールは充足，そうでなければ不十分とする．メインルールは，配下のサブルールが全て充足ならば充足，そうでなければ不十分とする．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．

チェック項目リストは，普段ツールの裏側で保持するが，学習者が必要に応じていつでも表示できるようにする．それにより，各ルールの充足状況を確認できるようにする．表示するチェック項目リストは，各ルールに対応するチェック欄と条件を表示する．チェック欄では，充足となったルールにチェックを付け，充足から不十分となったルールからチェックを外す．そして，学習者が必要に応じてチェック項目リストを表示できるように表形式で保持する．

2.3 学習者の進捗状況の把握

学習者が作成している成果物から学習者の進捗状況を把握する必要がある．そこで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，それを指摘する必要がある．そのため，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．以下，これらの機能について示す．

(1)　チェック項目リストから進捗状況を把握する

学習者が現在取り組んでいる要求管理ステップで，どの作業段階（メインルール）まで完了しているかを判断し，学習者が「現在取り組んでいるメインルール」を保持する．また，学習者が現在取り組んでいるメインルールから現在取り組んでいる要求管理ステップを判断し,「現在の要求管理ステップの番号」を保持する．現在学習者が取り組んでいるメインルール中のサブルールの合否から，当該メインルールのどの段階（サブルール）まで完了しているかを判断し，学習者が「現在取り組んでいるサブルール」を保持する．これにより，チェック項目リスト中の各ルールの合否から学習者の進捗状況を把握する．

(2)　学習者の誤った部分への取り組みを把握する

学習者が現在取り組んでいるメインルールに対して取り組むべき部分ではなく，先のメインルールに該当する部分に取り組んでいないかを検査する．その部分に取り組んでいる場合は，「警告表示が必要」とする．誤った部分への取り組みの検出は，まず学習者が現在編集したトピックが，どのメインルールで取り組む部分かを特定する．そして，学習者が取り組んでいる部分と取り組むべき部分の比較により行う．学習者が現在編集した要素は，学習者の操作に対して発生する astah*のイベントにより，更新された要素等に関する情報を取得し，特定する．これにより，学習者が現在編集した要素を特定し，学習者が取り組むべき部分よりも先の部分に取り組んでいないか判断する．

2.4　成果物作成時に必要な情報の提示及びツール画面

教育するにあたって，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必要がある．そこで，学習者は必要に応じて，必要な情報を表示できるようにする．また，警告などを自動で表示する．表示内容はメインルール及びサブルール毎に用意し，2.3 節の機能で得た進捗状況に関する情報を基に表示内容を決定する．図 6 に REMEST のメイン画面を示す．以下に各機能について説明する．

(1)　学習者が現在取り組んでいる作業項目を表示する

メイン画面上に，現在取り組んでいる要求管理ステップ名を表示する（図 6 ①）．表示内容は，「現在の要求管理ステップの番号」に基づいて決定する．また，現在取り組んでいる作業段階（メインルール）を簡単に表示する（図 6 ②）．表示内容は，「現在取り組んでいるメインルール」に基づいて決定する．これにより，学習者が現在何に取り組んでいるかを大まかに把握できるようにする．

(2)　学習者が現在取り組んでいる要求管理ステップの説明を表示する

メイン画面とは別に，要求管理ステップの説明ページを新たに表示する．説明ページは各ステップの最初に自動表示し，学習者に説明を読むように促す．表示内容は「現在取り組んでいるメインルール」に基づいて決定する．また，要求管理ステップの説明ページを閉じてもメイン画面上の「要求管理ステップの説明表示」ボタンを押すことで，いつでも要求管理ステップの説明ページを表示できるようにする．これにより，学習者がこれから取り組む要求管理ステップについて学習できるようにする．

(3)　学習者が次に行う操作に対する説明を表示する

メイン画面上の「次の操作表示部」に次の操作に関する簡単な説明を常に表示する（図 6 ③）．表示内容は「現在取り組んでいるサブルール」に基づいて決定する．また，メイン画面上の「次の操作の詳細説明表示」ボタンが押されると，次の操作に関する詳細説明ページをメイン画面とは別に表示する．これにより，学習者にこれから行うべき操作に関する指示を提示する．

(4)　アドバイスを表示する

メイン画面上の「アドバイスの表示」ボタンが押されたら，学習者の成果物（MindMap）に対して，作業を行うべき部分を示す．それと共にその部分に関する指摘やアドバイスを文章等で挿入する．MindMap への挿入は，テキストボックスや直線を利用する．また，表示内容は，「現在取り組んでいるサブルール」に基づいて決定する．これにより，学習者に対して成果物を作成する際のアドバイスや不十分な部分への指摘を行う．

(5)　警告を表示する

「警告表示が必要」と判断された場合，学習者が誤った部分に取り組んでいる旨を警告として学習者に対し，自動で表示する．表示はダイアログにより行う．これにより，学習者が取り組むべき部分よりも先の部分に取り組んでいる場合，そのことを学習者に警告する．

(6)　学習者の達成状況及び進捗状況を表示する

メイン画面上に学習者の進捗度合をステータスバーにより表示する（図 6 ④）．進捗度合は全てのサブルールのうち，充足されている物の割合により決定する．また，メイン画面上の「チェック項目リストの表示」ボタンが押されたら，チェック項目リストの表示ページをメイン画面とは別に表示する．それにより，学習者が自分の達成状況を視認できるようにする．表示時は 2.2 節の機能により表形式で保持している物を使用する．これにより，学習者が今どの程度できているか確認できるようにする．

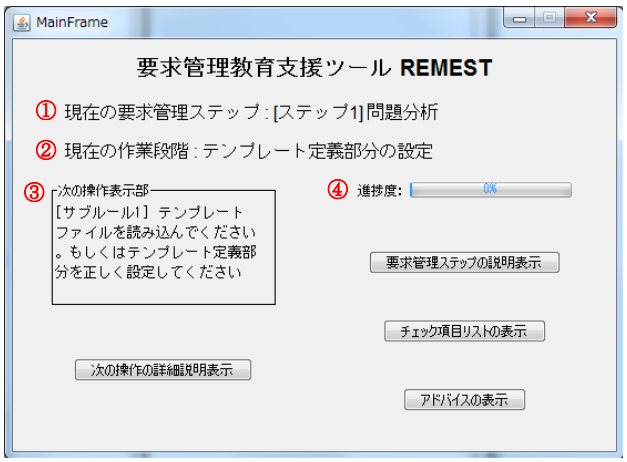

図 6　REMEST のメイン画面

Figure 6　Main Window of REMEST.

2.5　サブルール合否検査に対するツールの表示

現段階の REMEST は，要求管理ステップ 1（問題分析）の成果物を作成する際のサブルール 1～16, メインルール 1～4 の全ての合否検査を行う処理を実装している．また，各ルールの合否から学習者が現在取り組んでいるルールを特定し，それに適した説明等を表示する一連の処理を自動

的に行う機構まで完成させた．そのため，ステップ 1 の成果物を作成すると，REMEST が自動で成果物を検査し，進捗度に応じて画面の表示内容を更新される．REMEST を起動するには，astah*に追加した独自メニューをクリックする．それにより，図 6 のメイン画面が表示される．REMEST を用いた演習では，事前に配布するテンプレートファイルに対し，REMEST が提示する指示等を基に編集を行う形式となる．以下に REMEST を用いて要求管理ステップ 1 の成果物を作成する手順の内，例として，問題記述書までの作成手順にあたるサブルール 1〜5 について説明する．また，検査結果に対するメイン画面の表示内容を示す．各サブルールが充足されると，次のサブルールに進む．

■　サブルール 1

最初に図 1 のテンプレート定義部分が作成されたテンプレートファイルを開く．演習開始時，もしくはテンプレート定義のトピックが全て揃っていない場合や余分なトピックがある等，誤りがある時は，メイン画面の表示内容は図 6 の状態となる．サブルール 1 では，テンプレート定義部分を正しく設定する．

■　サブルール 2

サブルール 2 では，メイン画面の表示が，図 7 の状態となる．図 8 の「問題 2」のように，各問題について記述する際の基準となる問題（問題＋[番号]）を作成する．基準となる問題が 2 つ以上あり，番号に重複がないようにする．

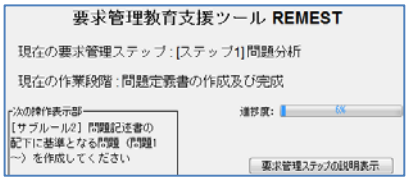

図 7　サブルール 2 に対する REMEST の表示

Figure 7　REMEST Message for Sub Rule 2.

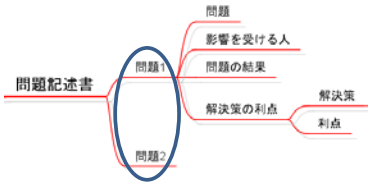

図 8　サブルール 2 に対応する Mind Map のノード

Figure 8　Mind Map Nodes Corresponding to Sub Rule 2.

■　サブルール 3

サブルール 3 では，メイン画面の表示が図 9 の状態となる．図 10 のように，全ての基準となる問題の配下にテンプレート定義部分である「問題 1」の部分と同様に問題記述書の詳細項目 4 つを作成する．

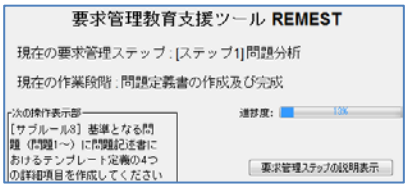

図 9　サブルール 3 に対する REMEST の表示

Figure 9　REMEST Message for Sub Rule 3.

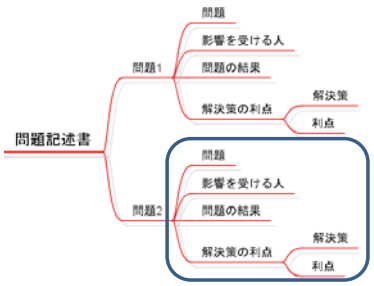

図 10　サブルール 3 に対応する Mind Map のノード

Figure 10　Mind Map Nodes Corresponding to Sub Rule 3.

■　サブルール 4

サブルール 4 では，メイン画面の表示が図 11 の状態となる．図 12 のように，問題記述書の詳細項目 4 つに関する記述を行うためのトピックを作成する．それらのトピックの配下にトピックは不必要であるため，作成しない．

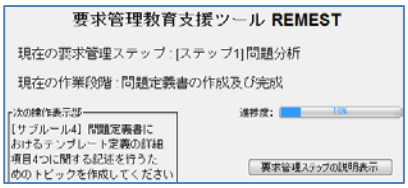

図 11　サブルール 4 に対する REMEST の表示

Figure 11　REMEST Message for Sub Rule 4.

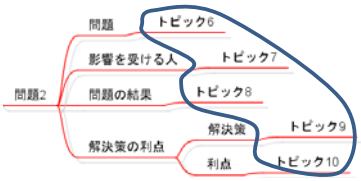

図 12　サブルール 4 に対応する Mind Map のノード

Figure 12　Mind Map Nodes Corresponding to Sub Rule 4.

■　サブルール 5

サブルール 5 では，メイン画面の表示が図 13 の状態となる．図 14 のように，サブルール 4 で作成したトピックに問題記述書の詳細項目 4 つについてそれぞれ記述する．この時，トピックの記述内容が全て図 12 のように初期状態（トピック＋[番号]）でない．また，英数字・記号のみからなる記述でなく，文字数は 3 文字以上にする．

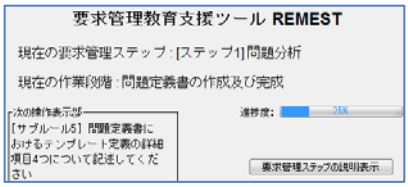

図 13　サブルール 5 に対する REMEST の表示
Figure 13　REMEST Message for Sub Rule 5.

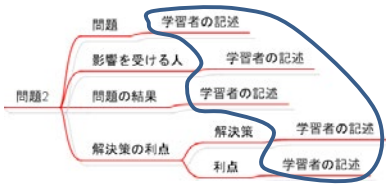

図 14　サブルール 5 に対応する Mind Map のノード
Figure 14　Mind Map Nodes Corresponding to Sub Rule 5.

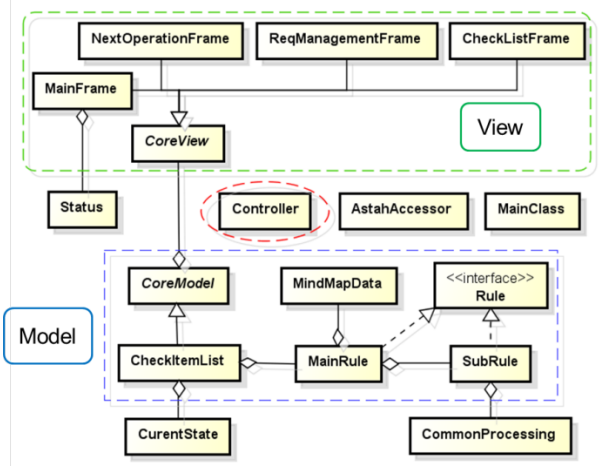

図 15　REMEST のクラス図(Model 部のサブクラス省略)
Figure 15　Class diagram of REMEST.

サブルール 6～16 においても，上記と同様に必要なトピックを作成し，各項目に対して記述を行っていく．成果物の作成を進めると，上記のように，学習者の進捗度に合わせて REMEST の表示が更新される．また，学習者の手戻りにも対応できる．

3. REMEST の開発

今回作成した要件定義を基に，クラス設計及びアルゴリズム設計を行った．また，ツールの実装は，Java と astah* API により行っている．

3.1 クラス設計

設計では，オブジェクト指向を用いるため，クラス図を作成した．作成したクラス図の全体構成を図 15 に示す．REMEST には，設計における工夫点が 2 つある．

1 つ目は，MVC モデルと Observer パターン[10]の適用である．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た．REMEST の設計では，Model 部をチェック項目リストやメインルール，サブルール，MindMap のデータとし，メインルール 1～4 やサブルール 1～16，ステップ 1 の成果物のデータに対応する物をサブクラスにしている．

2 つ目は Model 部の状態更新処理の自動化である．astah* API の ProjectEventListener を利用することで，REMEST は astah*プロジェクトを開いた時と成果物の編集時に自動で各サブルールの合否判定を行うようにした．上記の工夫により，学習者が作成中の成果物を自動で検査し，常に学習者の進捗状況を把握できる．

以下に，図 15 の各クラスについて説明する．各クラスの内，CheckItemList と MainRule の各サブクラス，SubRule の各サブクラス，MindMapData のサブクラス，CoreView のサブクラスには Singleton パターン[10]を適用している．

- CoreModel クラス

モデルの中心的なクラスであり，チェック項目リストのデータが更新された時，そのことを View 部に通知するのが主な役割である．このクラスは，Observer パターンの Subject 役にあたる．

- CheckItemList クラス

チェック項目リストに対応するクラスである．学習者の「現在取り組んでいるメインルール」及び「現在取り組んでいるサブルール」等の進捗状況に関する情報を保持する．

- Rule インターフェース

チェック項目リスト内のルールを定義するもので，ルールの合否を検査する validate メソッドを宣言している．

- MainRule クラス

チェック項目リスト中のメインルールにあたるクラスである．このクラスは，各メインルールを識別するための「メインルール番号」，メインルールの合否を記録する「ルールの合否」，メインルールが所属する要求管理ステップを表す「要求管理ステップ番号」，メインルールに対応する表示内容を保持する．また チェック項目リスト中の各メインルールに対応する MainRule1～4 の 4 個のサブクラスがある．

- SubRule クラス

チェック項目リスト中のサブルールに対応するクラスである．このクラスは，各サブルールを識別するための「サブルール番号」，ルールの合否を記録する「ルールの合否」，各サブルールが所属するメインルールを表す「所属メインルール番号」，各サブルールに対応する表示内容を保持する．このクラスには，SubRule1～16 の 16 個のサブクラスがあり，チェック項目リスト内の各サブルールに対応する．サブクラスでは，validate メソッドをオーバーライドし，サブルール毎にルールの合否検査処理を実装している．

- MindMapData クラス

astah*プロジェクトから取得した成果物に関する MindMap のデータをツール側で保持するクラスである．このクラスでは，要求管理ステップ毎の MindMap の識別に使用するルートトピックを保持する変数を定義している．また，要求

管理ステップ1のMindMapのデータを保持するためのサブクラスMindMapData1がある．

- CoreViewクラス

View部の中心的なクラスで，JFrameを実装している．また，Observerパターンの Observer役にあたる．

- MainFrameクラス

REMESTのメイン画面（図6）に対応するクラスである．

- NextOperationFrameクラス

REMESTの要求管理の説明表示画面（2節参照）に対応するクラスである．

- ReqManagementFrameクラス

REMESTの次の操作の詳細説明表示画面（2節参照）に対応するクラスである．

- CheckListFrameクラス

REMESTのチェック項目リストの表示画面（2節参照）に対応するクラスである．

- Controllerクラス

各画面のボタンのイベントハンドラの集合である．

- MainClassクラス

Model部及びView部のインスタンスの初期化を行う．

- AstahAccessorクラス

astah*本体に追加した独自メニューによるREMESTの起動やastah*上のプロジェクトの更新イベントの補足，astah*プロジェクトにアクセスしてデータを取得する役割を担う．このクラスは，astah*に追加する独自メニューの処理を実装するためのIPluginActionDelegate[9]とProjectEventListenerを実装している．

- CurrentStateクラス

メインルールとサブルールの合否判定の結果に基づいて，学習者の進捗状況を特定する．

- Statusクラス

ツール画面上の進捗バーに必要な値（ステータス値）の計算及び保持を行う．

- CommonProccessingクラス

各サブルールの合否判定処理の中から共通の処理をまとめたものである．

3.2 REMESTの状態更新処理

REMESTは状態更新処理を自動的に行う．その際には，まず，astah*プロジェクトから成果物のデータを取得し，ツール側で保持するMindMapのデータを更新する．次に，更新したMindMapデータを基にチェック項目リスト内の各ルールの合否を更新する．最後に，各ルールの合否から学習者の進捗状況を把握し，進捗状況に応じた説明文を表示する．

REMEST起動時の処理は，最初にModel部及びView部のインスタンスの生成及び初期化を行い，メイン画面を表示する．ツールの多重起動を防ぐために，これらの処理はツールが起動していない時のみ行う．初期化時は，全てのルールの合否を不十分とする．次に，astah*プロジェクトへのアクセスに必要なProjectAccessorを取得し，ProjectEventListenerを使用するために必要な処理を行う．最後に状態更新処理を行うという流れとなっている．状態更新処理は，ProjectEventListenerを利用して，ツールの起動時のみではなく，astah*プロジェクトが開かれた時と編集された時にも行う．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．以下に状態更新処理の流れの詳細を順に記す．状態更新処理は全てAstahAccessorから開始する．

1. astah*プロジェクトから要求管理ステップ1の成果物にあたるMindMapから必要なトピックデータを取得する．取得したデータをMindMapDataに渡すことで，ツール側で保持するMindMapのデータを更新する．
2. MainRule1～4に合否検査時に参照するMindMapデータとしてMindMapData1のインスタンスを渡す．
3. チェック項目リストの更新を行う．まず，各MainRuleの合否検査を行う．この時，合否結果が「不十分」となったMainRuleより先のMainRuleと配下のSubRuleの検査は行わず，合否結果を全て「不十分」にする．
4. MainRuleの合否検査では，MainRuleの配下にある各SubRuleの合否検査を行う．合否検査の際，MainRule同様，合否結果が「不十分」となったSubRuleより先のSubRuleの合否結果を全て「不十分」にする．これにより，学習者の手戻りにも対応でき，予期せぬエラーを防ぐことができる．
5. SubRuleの合否検査では，SubRuleが所属するMainRuleから検査時に参照するMindMapのデータ（MainMapData1）を取得する．そして，SubRule毎に検査時に必要なトピックデータを取得し，合否検査処理を行う．SubRuleによっては，CommonProcessingのメソッドを利用して合否検査を行う．
6. 学習者の進捗状況に関するデータを更新する．CurrentStateにて，現在学習者が取り組んでいるMainRule及びSubRuleを特定し，それぞれCheckItemListにて保持する．
7. REMESTのメイン画面に表示する進捗バーに必要なデータ（進捗度合）をStatusにて計算し，保持する．
8. 最後に，Observer役であるCoreModelのメソッドを呼び出して，Model部のデータが更新された事をView部に通知し，各画面の表示内容を更新する．この時，CheckListFrameでは，CheckItemListから全てのMainRuleを取得し，各MainRuleの配下のSubRuleを取得する．そして，各ルールの「ルールの合否」により画面上表示されるチェック項目リストの合否欄のチェックの付け外しを行う．その他の画面は，CheckItemListから現在学習者が取り組んでいる

MainRule 及び SubRule を取得し, それらが保持している表示内容のデータに更新する.

各サブルールの内，サブルール 5，7，9，14，15，16 の合否検査では，学習者が各項目に対して記述を行うトピックの記述内容を検査する．この検査では，正規表現を利用して記述内容が英数字・記号のみからなる文字列でないか等，可能な範囲で不適切な記述でない事を確認する．残りのサブルールの合否検査では，主に規定されたトピックの有無や余分なトピックがないか検査する．このように，成果物の基本的な部分を REMEST が自動で検査を行う.

4. 関連研究

本研究と同様に要求工学の教育に関する研究として，(1) プロトタイプ生成可能なモデル駆動要求分析手法の要求工学教育への適用[11]と(2) 開放型演習による要求工学教育の経験[3]が挙げられる.

[11]は，小形らが以前に提案したプロトタイプ自動生成可能なモデル駆動要求分析手法[12]を要求工学教育に適用したものであり，本研究と同様に astah*を利用している．しかし，本研究とは教育範囲が大きく異なる．本研究は要求管理ステップ 1～5 を含むのに対し, [11]はステップ 2 の要求を引き出す方法の一つであるプロトタイプをカバーしており，要求管理ステップのごく一部しか含んでいない．また, UML モデルが妥当か自己検証しなければならないが, 本研究はツールが自動的に可能な範囲で成果物の基本的な部分の妥当性を検証する.

[3]は要求分析の実践的な演習を行うことで，システム開発の現実に近い状況を学習するものである．それに対し，本研究は要求管理の基礎に対する教育を支援し，要求管理とは何か，どのように行うかを学習するものである．そのため，教育範囲が大きく異なる.

5. まとめと今後の研究課題

本研究では，astah* professional を活用した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，REMEST には要求管理ステップ 1（問題分析）の成果物を自動で検査することで，学習者の進捗状況を把握し，進捗状況に応じた適切な説明等を画面上に表示する機構まで実装を行った．これにより，REMEST は，要求管理ステップの成果物の作成過程を通して，要求管理の基礎教育を支援するための基盤となる機構を確立した．そのため，本研究では，重要性が高いが，まだ取り組みが不十分である要求管理について，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ることができた.

今後の研究課題として, 主に以下の 5 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，協力者に REMEST を使用して演習を行ってもらう．この評価実験で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，新たに要求管理の教育に有用な機能の企画検討を行っていく.

(2) 表示する説明文の作成

現段階の REMEST は, メイン画面に表示する説明文以外は未作成であるため，作成する必要がある.

(3) 要件定義の内，未実装である機能の開発

今回作成した要件定義の内，学習者の誤った部分への取り組みを判定する機能（2.3 節(2)と 2.4 節(5)）とアドバイスを表示する機能（2.4 節の(4)）の 2 つが未実装である．これらの機能の実現方法を確立し，実装する.

(4) ユーザインターフェースの改良

REMEST の各画面の内，メイン画面以外は試作段階であり, 修正する必要がある．また，利便性の観点から REMEST の画面を astah*の画面に統合できるか検討する必要がある.

(5) 要求管理ステップ 2～5 に対応する教育支援機能の開発

各要求管理ステップのチェック項目リストを作成し，チェック項目リストを管理する機能に加え，各ステップの成果物間の関連付けの妥当性を検査する機能を開発する.

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) 山本修一郎, 神戸雅一: 開放型演習による要求工学教育の経験, 電子情報通信学会技術報告, Vol109, No.307, pp.49-53 (2009).
4) 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
5) 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/
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) astah* Plug-in 開発チュートリアル, http://astah.change-vision.com/ja/plugin-tutorial/
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) 結城浩: 増補改訂版 Java 言語で学ぶデザインパターン入門, ソフトバンククリエイティブ株式会社 (2004).
11) 小形真平, 松浦佐江子: プロトタイプ生成可能なモデル駆動要求分析手法の要求工学教育への適用, 電子情報通信学会技術研究報告.KBSE, Vol.110 (468), pp.37-42 (2011).
12) 小形真平,松浦佐江子: UML 要求分析からの段階的な Web UI プロトタイプ自動生成手法, コンピュータソフトウェア, Vol. 27, No.2, pp.14-32 (2008).