三重大学教育学部研究紀要　第 69 巻　教育実践（2018）513－520 頁

スクラッチ・プロジェクトの形成的および総括的評価方法

大野　恵理*・須曽野仁志**

Formative and summative assessment for students' Scratch projects

Eri ONO and Hitoshi SUSONO

要　旨

アメリカ・マサチューセッツ工科大学で開発された無料の教育用プログラミング言語「スクラッチ」は、次期指導要領における 2020 年からのプログラミング教育必修化で注目が集まっている。本研究ではスクラッチでプログラミング教育を行う授業における児童・生徒の学習の評価について、アメリカの教育現場で用いられている合計 10 種類の形成的および総括的評価を紹介し、日本の教育現場における応用の可能性について論じる。

キーワード：形成的評価、総括的評価、プログラミング教育、スクラッチ、ルーブリック

はじめに

2017 年 3 月に次期学習指導要領が公示され、小学校では 2020 年に、中学校では 2021 年にプログラミング教育が完全実施されることとなった。様々なプログラミング言語が選択肢としてある中、本稿ではマサチューセッツ工科大学（MIT）で開発された無料の教育用プログラミング言語「スクラッチ」に焦点をあてる。NHK の E テレで 2017 年 4 月から放送されている「Why!?プログラミング」という番組では、スクラッチを通して楽しみながら「プログラミング的思考」を学ぶことができるようになっている（日本放送協会, 2017）。番組では、番組ウェブページで動画や資料を公開しており児童・生徒が学びやすい形になっている。また、指導者用に指導案、指導計画、およびワークシートが公開されていて、指導者も指導しやすくなっている。しかし、教師が児童・生徒の学習評価に使えるものはワークシートのみで、「大切な言葉などのメモをとる」「理解した内容を図や言葉で説明する」「振り返り（例、アルゴリズムについて理解しましたか）」「楽しかったこと、難しかったこと、もっと面白くする方法」について児童・生徒に記入させる形である。

スクラッチが開発されたアメリカでは、スクラッチでプログラミング的思考を教える教育者のためのオンライン・コミュニティ ScratchEd（スクラッチ・エド）があり、そこではスクラッチ・プロジェクトを評価する様々な方法が投稿されている。本稿ではスクラッチ・プロジェクトを評価する 10 種類の方法を「形成的評価」「総括的評価」「形成的および総括的評価」に分類した “Ten diverse ways to assess student learning with Scratch”（Leon & MacDonald, 2016）を紹介し、日本の教育現場での応用の可能性について論じる。

＊三重大学教育学部　東紀州サテライト東紀州教育学舎

＊＊三重大学教育学部教職大学院

1．形成的評価

形成的評価（formative assessment）とは、指導の途中で行われる評価で、意図した通りの教育効果がもたらされているかを確認する目的で行われ、もたらされていない場合には即時に授業計画の変更や回帰指導（補充学習など）の実施等の軌道修正を図る（西岡, 2015）。Leon ら (2016)によれば、スクラッチ・プロジェクトには 4 種類の形成的評価を用いることができる。

1.1 Gallery Critique（ギャラリー・クリティーク）

Gallery Critique（「ギャラリー・クリティーク」、以下、GC と略す）は、児童・生徒が主体的に関わる評価（student-engaged assessment）方法の一つで、ギャラリーを歩きながら展示されている絵画を評価する様子に似ていることから Gallery Walk（ギャラリー・ウォーク）とも呼ばれる。Berger ら（2015）によって紹介されている一般的な GC は 4 段階で構成されている。まず、四段階の前のイントロダクションでは教師が各段階の手順（プロトコル）の説明をし、「学習のねらい」（learning target）を説明する。（1）第一段階：すべての児童・生徒が作品の草稿（first draft）を教室の壁に掲示する（所要時間 5 分）。（2）第二段階：児童・生徒は掲示された作品を「学習のねらい」に沿って見て、作品に長所があればメモを取る（所要時間 5 分）。（3）第三段階：児童・生徒は自分の席に戻り、教師が主導となって、「どういった点に気付いたか？」を話し合う（所要時間 5 分）。（4）第四段階：教師が主導となって、前段階で話し合われた「気付いた点」に具体例を加えて深く掘り下げる（所要時間 15 分）。

スクラッチのプロジェクトにおいては、児童・生徒が自分のプロジェクトの設計図（例、図やスクリプトを用いたもの）を教室の壁に貼り出したり、試作段階のスクラッチ・プロジェクトをコンピュータに表示しておく方法が可能である。Leon ら（2016）によると、四段階の過程を通して、児童・生徒は他の児童・生徒の作品を評価するとともに、自分の作品を自己評価（self-evaluation）する学習効果がある。

GC を日本の教育現場で利用するには、学級の児童・生徒数を考慮する必要がある。文部科学統計要覧（平成 28 年版）によると、日本の小学校の 1 学級あたり平均児童数は 24 人で、中学校の平均生徒数は 28 人である。この児童・生徒数で GC を行うのであれば、所要時間 5 分で 20 人超の作品を見て、その長所を具体例も含めてメモを取ることは不可能である。そこで日本の教育現場に応じた GC の方法として提案されるのが、第二段階を 6 人以下の小グループで行うことである。児童・生徒は自分のグループ内のすべての作品を見ることができ、他の児童・生徒の作品を評価とともに、自分の作品を自己評価する十分な時間を確保することができる。また、第二段階で付箋等を利用して作品の長所を作品に直接貼ることで長所が一目瞭然となり、教師が第三段階、第四段階の話し合いを主導しやすくなる。

1.2 Design Scenarios

Design Scenarios（「デザイン・シナリオズ」、以下、DS と略す）は、児童・生徒が computational thinking（計算論的思考、以下、CT と略す）をどの程度習得できたか評価するための方法の一つとして、MIT のメディアラボの Brennan と Resnick（2012）によって考案された、教室（classroom settings）で限定的に使う評価法である。児童・生徒はペア（対）のスクラッチ・プロジェクトを見せられる。それぞれのプロジェクトは、同じ CT を含んでいるが全く違うプロジェクトである。児童・生徒は自分の好みのプロジェクトを選び、以下の 4 つの学習活動に取り組む。（1）選んだプロジェクトについて説明する（critique）、（2）選んだプロジェクトを良くする方法を説明する（extending）、（3）プログラムのバグ修正（debugging）、（4）新しい特徴を付け加えて改造する（remixing）。

DS の長所の一つは児童・生徒が違うタイプの「知っている（knowing）」を体験できることである。児童・生徒は上記の 4 種類の活動に取り組むことで、体系的な 4 種類の探究を体験することになる。また、DS は児童・生徒の CT の習得の段階を評価することができる有効な手段である。さらに、DS の 4

つの活動では、児童・生徒は情報を頭の中で処理する（process-in-memory）のではなく、実行すること（process-in-action）が求められ、CT を本当に身に着けることができているのか評価することができる。その一方で、DS の短所は、4 つの活動に取り組むことが求められるため、授業という限られた時間内に取り組むには時間がかかりすぎる。また、DS で提示されるスクラッチ・プロジェクトは他者が制作したものであるため、児童・生徒の個々の興味・関心に応じたものでない可能性が高い。

DS は、児童・生徒に 2 つのプロジェクトから 1 つ選ばせるため、評価する教師にとっては評価するプロジェクトは 2 パターンしかないので、短い時間で評価することができ、1 学級あたり平均児童・生徒数が多い日本の教育現場では理想的な評価法である。しかし、Brennan らがインターネット上で公開している DS 評価法で用いるプロジェクトは、英語版スクラッチを用いて制作されている。そのため、日本語版スクラッチを使う日本の児童・生徒には、これらのプロジェクトをそのまま使うことはできない。もし、これらのプロジェクトが日本語版スクラッチで制作されれば、日本での DS 評価法のニーズは高いと考えられる。

1.3 Student Design Journals

ジャーナル（journals）とは一般的に「日記」や「雑誌」と日本語に訳されるが、Leon ら（2016）が提案する student design journal とは「児童・生徒はプロジェクトの一環として達成したこと、問題点、新しく学んだこと等を記すこと」を指し、これはアメリカの教育現場では一般的に内省ジャーナル（reflective journal）と呼ばれるものと同じものと考えられる。この学習活動は、児童・生徒にとって「学ぶために書く（writing to learn）」という学習効果があり、また、教師にとっても、児童・生徒が学習内容を理解できたかチェックすることのできる有効な評価方法である（Berger, Rugen, & Woodfin, 2015）。前述の NHK の「Why!?プログラミング」という番組のウェブサイトに公開されているワークシートには、「大切な言葉などのメモをとる」「理解した内容を図や言葉で説明する」「振り返り」「楽しかったこと、難しかったこと、もっと面白くする方法」について用紙に記入する形であり、ワークシートがジャーナルの役目を果たしている。

1.4 Peer Feedback in Small Groups

Peer Feedback（「ピア・フィードバック」、以下、PF と略す）は仲間に意見や感想を述べることで、Leon ら（2016）は、PF のためにクラス全員で PF 専用のルーブリックを作り、そのあと小グループに分かれ、グループ内でルーブリックに沿って仲間の作品を評価することを提案している。Berger ら（2015）によると、PF の問題点はフィードバックの内容が漠然としたあいまいなものになりやすいことや、フィードバックとは関係のない話に逸脱してしまいやすいことである。これらの問題点を解決するために、Leon らは児童・生徒による PF 用のルーブリック作りを提案している。

日本でも PF のような活動があり、「隣の人の作品を見て、どんなところが素晴らしいか、意見を言いましょう。」のような活動はどの教科の授業でも行われているが、Berger（2015）らが指摘するように漠然としたフィードバックになっていることが多い。提案されている通りルーブリックを用いることで、より具体的で客観的なフィードバックが可能となり、小グループでの PF では複数の仲間からのフィードバックが可能となる。残念ながら、日本にはルーブリックを用いた評価はまだ馴染みがなく、児童・生徒がルーブリックを作ったり、それを用いて仲間を評価する文化は根付いていないと考えられる。日本で PF を学習活動として導入するには、まず教師がルーブリックを作り、サンプル・プロジェクトをルーブリックで評価するのを児童・生徒に見せて、PF の仕方の見本を見せることから始めなければならない。

２．総括的評価

総括的評価（Summative Assessment）とは、「単元末や学期末、年度末といった学習の締めくくりに学習の到達点を把握する目的で行われるもの」（西岡, 2015, p.6）と定義される。スクラッチ・プロジェクトには３種類の総括的評価を用いることができる（Leon & MacDonald, 2016)。

2.1 Scratch Project Rubric

ルーブリックとは、パフォーマンス課題の遂行状況の評定にあたって用いられる評価指標であって、ひとまとまりのパフォーマンスの質を複数の側面から採点するための指標として用いられる（遠藤, 2015)。ルーブリックの長所は、実演や作品の審査制を高めることができ、また学習活動の初期段階から子どもに示すことで、子どもたちの自己評価を促すことが目指される。Leon ら（2016）によると、教師が作ったルーブリックでも、教師と児童・生徒の合作のルーブリックでも、スクラッチ・プロジェクトの評価基準となり、また、児童・生徒の自己評価を促すとされている。そこで参考として挙げられているのが Randall（2009）が作成した “Rubric for assessing Scratch projects” である。このルーブリックを日本語に訳したものが表 1「スクラッチを用いた教科横断的学習を評価するルーブリック」であり、４つの観点において 4 段階で評価する。このルーブリックは、教科横断的な学習にプログラミングを加えた学習を総括評価する際に使うルーブリックで、まだ日本では教科横断的学習でスクラッチ・プロジェクトを指導している教育現場は非常に限られている。しかし、教科担任制でない小学校では、プログラミングの専門性を有しないで担任が各教科の学習の中でプログラミング教育を行うことになり、教科の学習に含まれる以上、プログラミング学習や学習の過程を何らかの評価をしていく必要があるため、このような教科横断的学習とプログラミング学習の両方を評価できる評価方法のニーズは高まると考えられる。

表１．スクラッチを用いた教科横断的学習を評価するルーブリック

評価の観点/到達レベル	初級	中級	上級	達人
（A）教科の知識	①	②	③	④
（B）プロジェクトのデザイン	⑤	⑥	⑦	⑧
（C）プログラミングの知識	⑨	⑩	⑪	⑫
（D）学習の過程	⑬	⑭	⑮	⑯

観点 1：（A）教科の知識

初級：教科で学習した内容が含まれていない、もしくは誤った内容が含まれている ― 表 1 の①

中級：教科で学習した内容が少し含まれ、教科の学習の理解が見られる ― 表 1 の②

上級：教科で学習した内容が正しく理解されている ― 表 1 の③

達人：教科で学習した内容を結び付け、教科の内容が深く理解されている ― 表 1 の④

観点 2：（B）プロジェクトのデザイン

初級：作品に独自性がない。作品の意図が不明瞭で、内容が体系化されていない。他の児童・生徒が作品と触れ合う方法が与えられていない（例：ボタンを押すと紙芝居が再生される場合、ボタンが設定されていない） ― 表 1 の⑤

中級：他人の作品をほぼ模倣しているが、少し変更点が加えられている。作品の意図が少し不明瞭で、

内容の体系化が不十分である。他の児童・生徒が作品と触れ合う方法があるが、作品の目的に合致するために方法を変える必要がある。― 表 1 の⑥

上級：作品に独自性があり、既存のアイディアを独創的に再利用できている。作品の意図が明瞭で、内容が体系化されている。他の児童・生徒が作品と触れ合う方法が設定されており、「作品の遊び方」の説明文（インストラクション）が明瞭ある。― 表 1 の⑦

達人：作品が独創的で、学習した教科の内容としっかり関連付けされている。作品が複雑にデザインされている。他の児童・生徒が作品と触れ合う方法が、作品の目標と合致している。― 表 1 の⑧

観点 3：(C) プログラミングの知識

初級：ブロックの働きや、ブロックが連動して動くことが全く理解できていない。プログラミングが体系化されておらず、論理的でない。多数のプログラミングのエラーがある。― 表 1 の⑨

中級：ブロックの働きや、ブロックが連動して動くことが少し理解できている。プログラミングが少し体系化されており、少し論理的である。数か所プログラミングのエラーがある。― 表 1 の⑩

上級：ブロックの働きや、ブロックが連動して動くことが理解できている。プログラミングが体系化されており、論理的である。プログラミングのエラーがない。― 表 1 の⑪

達人：ブロックの働きや、ブロックが連動して動くことがより深く理解できている。プログラミングがより正しく体系化されており、より論理的である。プログラミングのエラーがなく、より高度なプログラミングの技術を活用できている。― 表 1 の⑫

観点 4：(D) 学習の過程

初級：児童・生徒は、基本的なデザインの過程（1. 問題提起、2. アイディアを膨らませる、3. 解決法を選ぶ、4. 選択した解決法で問題を解決した場合の結果の提示）を全く活用できていない。毎時間の課題の提出期限に間に合わない。作品にかかる時間配分ができていない。― 表 1 の⑬

中級：児童・生徒は、基本的なデザインの過程を活用しようと努力しているが、正確に活用できていない。毎時間の課題の提出期限に時々間にあう。作品にかかる時間配分が、毎時間ではないができている。―表 1 の⑭

上級：児童・生徒は、基本的なデザインの過程を活用できている。毎時間の課題の提出期限に間に合う。作品にかかる時間配分が、毎時間できている。―表 1 の⑮

達人：児童・生徒は、基本的なデザインの過程をより正確に活用できている。毎時間の課題の提出期限に余裕で間に合い、追加的な要素を加えることが可能である。作品にかかる時間配分がより正確で、要領よく作業が進められている。―表 1 の⑯

2.2 Dr. Scratch

Dr. Scratch（ドクター・スクラッチ）は児童・生徒のスクラッチ・プロジェクトを客観評価するために開発された無料のコンピュータ・アプリケーションである（Moreno-Leon, Robles & Roman-Gonzalez, 2015）。スクラッチ・プロジェクトにおいて CT をマスターできているかどうかを瞬時に客観評価することができる。 Dr. Scratch の使い方は、児童・生徒のプロジェクトを Dr. Scratch のホームページにアップロードすると、CT の 7 分野に関しての得点とコメントが表示される。7 分野とは（1）Flow Control（フロー制御）（2）Data Representation（データー表現）（3）Abstraction（抽象化）（4）User interactivity（インタラクティブ）（5）Synchronization（同期）（6）Parallelism（パラレリズム）（7）Logic（ロジック）で、各分野 3 点満点で、最高得点 21 点となる。点数に加えて、それぞれのレベルに応じたコメントが表示される。例えば、プログラミング初心者には、初心者を圧倒しないように基本的なコメントが表示され、

上級者には、児童・生徒のプログラミングにおける悪い習慣（bad programming habits）が検出されると、上達するためのコメントと関連した情報が表示される。さらに、到達不能コード（dead code）、重複コード（duplicated code）等も検出して表示される。現在、Dr. Scratch は英語等の 8 か国語から選べるが、日本語化はされていないため、小中学校での使用は「点数」を参考する程度の使用に限定されると考えられ、日本語化が望まれる。

2.3 Scratch Independent Project Checklist

スクラッチ・インディペンデント・プロジェクト・チェックリスト（「scratch independent project checklist」、以下、チェックリストと略す）とは、教師から提示したスクリプトさえ含んでさえいたら 4 時間の授業を使ってスクラッチ・プロジェクトを自由に作っていいという、「スクラッチ・自由プロジェクト」の場合の総括的評価に使える評価法である。チェックリストの項目は以下の 3 点で、(1) スクラッチ・プロジェクトを先生に電子メールで送る、(2) 記述式の感想文（例、プロジェクトの説明、計画した通りにプロジェクトが制作できたか等）を先生に電子メールで送る、(3) ジャーナル（前述の内省ジャーナル）を毎時間書く。児童・生徒はこの 3 点を必ず実行することが求められ、提出されたスクラッチ・プロジェクトやジャーナルは、ルーブリック等を使って教師が単独で評価する。

チェクリストは、スクラッチ・プロジェクトを一人でデザイン・制作することのできる上級者の学習を評価するのに適した方法である。上級者は、今までのプログラミング学習の過程で多種多様の評価方法（例、ルーブリック、ピア―・フィードバック等）を経験してきているため、教師と上級者の間には評価に関して「一定の共通認識」が成立していると考えられる。よって、上級者は教師単独による総括的評価に不満を抱くことはなく、教師にとっても 3 つの提出物（プロジェクト・感想文、ジャーナル）を事前に提示しておいたルーブリック等を使って評価する効率的な作業である。日本の教育現場でも、上級者であれば小学生でもこのチェックリストは応用できると考えられるが、「一定の共通認識」を持てるように、児童・生徒が上級者になるまでに多種多様な評価方法を経験させる必要がある。

３．形成的および総括的評価

形成的評価および総括的評価の両方に使える評価方法も提案されている。

3.1. Project Portfolio Analysis

一般的にポートフォリオとは、「子どもの作品や自己評価の記録、教師の指導と評価の記録などを系統的に蓄積していくもの。」を意味している（遠藤，2015）。スクラッチ・プロジェクトにおいては、スクラッチのオンライン・コミュニティに自分のプロジェクトを投稿することで、プロジェクトをオンラインで蓄積することができる。ここで投稿されたプロジェクトは Scrape（スクレープ）（http://happyanalyzing.com/）というコンピュータ・アプリケーションを使ってプロジェクト内で使用されたブロック（命令）の種類や頻度を分析することで、児童・生徒が CT を身に着けることができたか評価することができる。ただ、この分析方法は完成したプロジェクトのみを評価することになり、完成に至る学習過程を評価の対象にしないことになる（Brennan & Resnick, 2012）。

一学級の平均児童・生徒数が 20 を超える日本で、担任の教師 1 人で全ての児童・生徒のスクラッチ・プロジェクトの学習評価するのは時間的に厳しく、前述の Dr. Scratch や Scrape などのアプリを使っての機械的な評価を否定することはできない。ただ、アプリを使った評価だけでなく、本稿にある他の評価方法（例、ジャーナル、ルーブリック等）を併用し、児童・生徒の学習の過程も評価の対象にすることが望ましい。

3.2 Artifact Based Interviews

Artifact とは一般的に「芸術品」と訳されるが、本稿では完成したスクラッチ・プロジェクトを指す。前述の Scrape が児童・生徒が CT を身に着けることができているか機械的に評価する一方で、完成した作品について教師が児童・生徒を面談するのが Artifact Based Interviews（アーティファクト・ベースド・インタビューズ、以降 ABI と略す）である。面談では、教師が児童・生徒に制作したスクラッチ・プロジェクトについての説明、制作の経緯、制作の過程等を質問する。この評価方法の長所は学習過程を評価することができることであるが、その一方で、面談は時間がかかり、児童・生徒が学習過程を（覚えていないため）詳細に答えることができなかったり、正直な回答をしないなどの欠点がある。

Brennan らが行った ABI（2012）は、児童・生徒一人につき 60 分〜120 分かかり、日本の教育現場で同じレベルの ABI を行うことは非現実的であるが、自分のプロジェクトについて「自分の言葉」で発表させることは、前述の “write to learn”（学ぶために書く）のような学習効果が期待される。そこで、本稿では ABI の代替策として児童・生徒による作品ー発表会を提案する。スクラッチ・プロジェクト完成後、児童・生徒が 2〜3 分程度で自分のプロジェクトや学習過程等をクラスメートや教師の前でプレゼンテーションする。プロジェクト制作の事前や形成的評価時にどのコード（例、アルゴリズム）を使わないといけないかデザイン仕様（Design specification）を提示しておくと、目標とする CT が習得できたかプレゼンテーションで評価することができる。

3.3 Computational Thinking Rubric

Computational thinking rubric（「コンピューテーショナル・シンキング・ルーブリック」、以降 CTR と略す）は、スクラッチ・プロジェクトの 3 分野（デザイン、プログラミング、学習プロセス）を児童・生徒に自己評価させるために開発されたルーブリックである（Ross-Kleinmann, 2013）。前述の表 1 「スクラッチを用いた教科横断的学習を評価するルーブリック」の第一項目「教科の知識」を外した形となる。このルーブリックは学習の過程で形成的評価、および学習の終わりに総括的評価にも使うことができる。

CT 習得のための評価に限定した CTR は、教科の内容を含まない。CTR を導入することで、児童・生徒の CT 習得がどの段階にあるか一目瞭然となり、児童・生徒の自己評価を促すとともに、教師はその結果を指導や評価に反映させやすくなる。CTR が投稿されている ScratchEd のオンライン・コミュニティには、CTR を含めた全てのルーブリックは日本語化されておらず、日本語化されることが望まれる。

４．おわりに

本稿では 2020 年から日本の小中学校で必修化されるプログラミング教育に関して、「プログラミング的思考」を「スクラッチ」という言語を通して学ぶ場合の、アメリカの教育現場で使われている形成的および総括的評価を紹介し、日本の教育現場での応用の可能性について議論した。次期指導要領にあるプログラミング教育とは、プログラミング言語を覚えることではなく、コンピュータに意図した処理を行うよう指示することができることを意味し、特に小学校では身近な生活でコンピュータが活用されていることや、問題の解決には必要な手順があることに気づくことである（文部科学省，2016）。また教科担任制でない小学校では、プログラミングの専門性を有しないで担任が各教科の学習の中でプログラミング教育を行うことになる。教科の学習に含まれる以上、プログラミング学習や学習の過程を何らかの評価をしていく必要があり、NHK のウェブサイトで公開されているワークシートのようなものだけでは十分に評価できないと考えられる。本稿で取り上げた 10 種類の評価方法は、プログラミング教育先進国であるアメリカの小・中・高等学校で実際に使われているもので、それぞれ日本語化して日本

の教育現場に応じた形になるよう変更すれば、日本の教育現場で使える可能性が高い。

引用文献

遠藤貴広(2015). 学力評価の方法　西岡加名恵・石井英真・田中耕治(編) 新しい教育評価入門　有斐閣 pp.113-142.

西岡加名恵(2015). 教育評価とは何か　西岡加名恵・石井英真・田中耕治(編)新しい教育評価入門　有斐閣 pp.1-22.

日本放送協会：Why!?プログラミング，http://www.nhk.or.jp/gijutsu/programming/origin/scratch/ playworld.html, 2017 年 5 月 17 日アクセス.

文部科学省：小学校段階におけるプログラミング教育の在り方（議論の取りまとめ）（2016），http://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/houkoku/1372522.htm, 2017 年 10 月 30 日アクセス.

文部科学省：文部科学統計要覧（平成 28 年版），http://www.mext.go.jp/b_menu/toukei/002/002b/1368900.htm, 2017 年 10 月 30 日アクセス.

Berger, R., Rugen, L., & Woodfin, L. (2014). Leaders of their own learning. San Francisco: Jossey-Bass.

Brennan, K., & Resnick, M. (2012). *Using artifact-based interviews to study the development of computational thinking in interactive media design*. Paper presented at annual American Educational Research Association meeting, Vancouver, BC, Canada.

Leon, C., MacDonald, K. (2016).　Ten diverse ways to assess student learning with Scratch, http://scratched.gse.harvard.edu/resources/beyond-rubric-methods-assessing-scratch-projects, 2017 年 5 月 21 日アクセス.

Moreno-Leon, J., Robles, G., & Roman-Gonzalez, M. (2015). Dr. Scratch: Automatic analysis of Scratch projects to assess and foster computational thinking. Retrieved from http://www.um.es/ead/red/46/moreno_robles.pdf

Randal, K. (2009). Rubric for assessing Scratch projects--DRAFT, http://scratched.gse.harvard.edu/resources/rubric-assessing-scratch-projects-draft-0, 2017 年 5 月 21 日アクセス.

Ross-Kleinmann,J. (2013). Computational thinking rubrics, http://scratched.gse.harvard.edu/resources/computational-thinking-rubrics, 2017 年 5 月 21 日アクセス.