Vol.2021-GN-112 No.14
Vol.2021-CDS-30 No.14
Vol.2021-DCC-27 No.14
2021/1/25

ストーリーを要素の相互関係と シーンによって記述する言語の提案

上森康太郎 [1]　荻原剛志 [2]

概要：長編のストーリーでは，登場人物やアイテムなどの間の相互関係や時間的な前後関係を把握して一貫性を確認するのが難しくなる傾向がある．本稿ではストーリーを，登場人物やできごとの関係をプログラミング言語によって記述したものと捉える方法を提案する．この手法によるストーリーの記述は特定のコンテンツを前提としたものではなく，相互関係や前後関係に矛盾がないかをチェックしながらストーリーの作成，編集を行うことを主な目的とする．

キーワード：物語，ストーリー，小説，プログラミング言語，シーン，イベント，プロット

A Programming Language That Describes Stories as the Interrelationship of Elements and Scenes

KOTARO UEMORI[†1]　TAKESHI OGIHARA[†2]

1. はじめに

映画，小説，ゲーム，アニメーションなどのコンテンツは，何らかの「物語」を表現している．本稿では以下，コンテンツによって表現される物語をストーリーと呼ぶ．

ストーリーは通常，自然言語またはマンガや絵コンテのような形式で記述されている．そのため，ストーリーの規模が大きくなると，数多くの登場人物やアイテムの間の関連や時間的な前後関係などを把握するのが困難になってくる．すると，ストーリーの作成，修正に伴って，登場人物の間の関係やできごとの時系列に矛盾が起きても気づかず，最悪の場合はストーリー自体が破綻してしまいかねない．

本稿では，この問題を軽減または解消するため，ストーリーをコンピュータで処理可能なデータとして記述し，ストーリー作成を支援する手段やツールを提供することを提案する．提案では，ストーリーの記述をプログラミング言語によって行い，この言語の実行と記述内容の検証機能によって上述の問題に対応することができることを示す．プログラミング言語によるストーリーの記述は特定のコンテンツでの表現を前提としない．

意味情報をプログラミング言語で扱う場合，自然言語の意味や背景知識をコンピュータが理解できないという問題が指摘されているが，事前にストーリー要素の定義や言葉の定義を行うことで手法の適用が可能になる．これによってストーリー作成の問題点を軽減することができるだけでなく，歴史やニュースなどのような，物語に似た情報の記述を支援する効果も期待される．

本稿では以下，ストーリーをプログラミング言語を用いて記述する手法と，この提案に基づいた試作システムの概要について述べる．

2. 研究の背景

2.1 ストーリーの複雑化に伴う問題

映画や小説，ゲームなどのストーリーは，人気作であれば続編が次々と作られたり，外伝（サイドストーリー）やスピンオフと呼ばれる派生作品が，時には別のクリエータによって作成されることがある．このようにしてストーリーに現れる人物やアイテム，できごとの数が増大して行くにつれて，それらの間の関係性もまた爆発的に増加し，把握が困難になってしまう．

これによって，ストーリーを新しく作成，追加，あるいは修正した際に，人物間の関係や時系列に矛盾が発生する可能性が高くなるという問題がある．ここでいう矛盾とは，要素自体，または要素間の関係性，要素の属性が，記述がないのに出現，消失，あるいは変更されている（登場人物が急に別の場所に移動しているなど）こと，推測される関係性とストーリーの記述が異なる（親子の年齢が逆転しているなど）ことを指す．

また，ストーリー作成の際に，できごとの順番を変更したり，新しいエピソードを追加したりしたいが，すでに作成した部分と矛盾がないことを簡単に調べられないという問題も存在する．

1 京都産業大学大学院 先端情報学研究科
Kyoto Sangyo University
2 京都産業大学 情報理工学部

Vol.2021-GN-112 No.14
Vol.2021-CDS-30 No.14
Vol.2021-DCC-27 No.14
2021/1/25

2.2 ストーリーの複雑化に伴う問題

上で述べたような問題に対応するため，ストーリーの各要素と時間の経過に伴う関係の変化がコンピュータで処理可能なデータとして記述できていれば，ストーリーの進行に伴う矛盾点が発見しやすくなると期待される．さらに，主要人物なのに活躍の場がなかったり，物語の背景知識や社会環境から外れた行動やできごとを記述してしまったり，似たようなできごとの繰り返しをストーリーに取り入れてしまうなどのミスを防ぐことも期待できる．

このために，ストーリーを成り立たせている情報を記述しておく必要がある．記述すべき情報は，大きく分けて３種類あると考えられる．第１に，ストーリーの背景の情報である．これはストーリーの舞台の背景知識や社会環境といった世界観の記述（王政である， 支配的な宗教がある，魔法や近代兵器の有無，など）である．

第２は登場人物やアイテムの記述である．本稿ではストーリーに現れる人物やアイテムをストーリー要素と呼ぶことにする．これらの要素の名前や属性の記述，要素の間の関係を記述することが必要である．

第３にはストーリー自体の記述である．主要なプロット，ストーリー内のできごとやエピソード，伏線などの記述，できごとの時間的前後関係の記述，できごとによる要素間の関係の変化，属性の変化を記述することである．

また，小説として作成し始めたものを，後からコミックやゲームのような別のコンテンツとしてまとめたり，アニメ化に際してエピソードを追加，省略したりすることがありうる．したがって，ストーリーの情報を記述する場合，可能性のあるいくつかのコンテンツで利用が可能であるようにしておくことが望ましいであろう．

2.3 ストーリー情報の記述に関する先行研究

これまで，この問題をコンテンツ制作の視点から解決する研究やツールの開発が行われてきた．しかし，特定のコンテンツを制作するための支援を行う手法は存在するが，ストーリー情報の整理を主眼においた研究は未だ行われていない．ここでは代表的な事例をいくつか取り上げる．

映像コンテンツ制作を扱った研究に戀津[1]の論文がある．この研究では，ストーリーの情報を構造化することで脚本を映像に起こす際に必要な情報を容易に作成することを可能にした．また Brooks[2]は，入力されたストーリーから適切なフィードバックを行いコンテンツの制作を支援する手法を研究している．これらの研究はコンテンツ制作を支援する目的で行われており，ストーリー情報の構造化もそれに必要な情報のみを扱っている．

長久[3]は RPG（ロールプレイングゲーム）のシナリオに登場するキャラクタの行動を有限状態機械(FSM)モデルで記述することによって，ゲームの動作を検証する手法を示した．しかしこの手法には，ストーリーにおけるできごとの順序や，登場するキャラクタ間の関係性を記述する方法は含まれていない．

ストーリーの執筆を支援するツールに吉里吉里２[4], Nola[5], WorldType[6]がある．吉里吉里２はノベルゲーム制作向けに開発されており，選択肢や分岐を含んだストーリー作成に適している．Nola は小説執筆支援を目的に開発されており，登場人物や物語の起承転結を記入することができる．この２つのツールは，小説または特定のコンテンツの作成を前提としており，汎用的にストーリーの情報を記述することはできない．WorldType はこれらとは違い，特定のコンテンツに依存していない．主に物語内の宗教や社会問題などの世界観を整理するために用いられ，その物語世界の状態を理解しやすくすることができるが，プロット制作・補助機能はない．

高橋ら[7]はテキスト形式によるアニメーション脚本を、XML 形式による「構造化シナリオ」へと変換するソフトウェアを試作した。このソフトウェアはアニメーションの作成におけるビデオコンテを作成することを目的としており，人物やアイテムの相互関係を記述するものではない．

このように，これまでの手法は単にテキスト情報を複数保管したり，ただひとつの時間経過でしか物語の情報を記述することができず，できごとの前後関係を変更したり，できごとの発生に伴って要素間の関係性が変化したりすることを記述できない．また，要素やできごとをプログラムで扱うことができるデータとして記述できない．

2.4 本研究の提案

本研究では，特定のコンテンツを前提とせず，ストーリーを記述するためのプログラミング言語を提案する．この言語は，ストーリーに登場する要素（人物やアイテム）自体とそれらの相互関係の記述，および時間経過に伴って発生するできごと（シーンと呼ぶ）の記述を行う．

シーンは，要素の属性や要素間の関係を変更する操作を含むように記述できる．シーンの実行順序がストーリーの時間経過を表現し，さらにその実行に伴って要素とその相互関係が変更される．シーンは構造的に記述が可能であり，シーンの前後関係を入れ替えることも可能である．

提案する言語は，ストーリーのある時点における要素間の関係を述語論理として評価可能なデータ表現として出力する機能を持つ．これによって，シーンの順序を入れ替えたりしたことによって，要素間に矛盾が発生していないかなどを検証することが可能となる．

一方，提案言語では定量的な値を持つデータ，あるいは手続きとして表現することができないものは記述の対象としない．したがって，物語の背景知識や世界観の記述は行わない．同様に，人物の心情や行動原理，物語における価値観なども記述は困難である．

Vol.2021-GN-112 No.14
Vol.2021-CDS-30 No.14
Vol.2021-DCC-27 No.14
2021/1/25

3. シーンとストーリー要素

3.1 シーンの順序と物語時間の関係

物語（ストーリー）とは，人や事件などの一部始終について語られたものや書かれたものaとされる．ここではストーリーとは，語られるものに関するいくつかのできごとの連続であると考える．

提案する言語では，「できごと」に相当する概念をシーン（場面）と呼び，複数のシーンからなるまとまりをストーリーと呼ぶ．さらにシーンは複数のイベントから構成される．シーンは一連のイベントをまとめるコンテナとして機能する．イベントは必ずいずれか一つのシーンと関連付けられ，イベント単体で存在することはない．また，シーンに名前をつけることでストーリー全体を把握しやすくできる．図 3 はシーンとイベントの関係の例を示している．

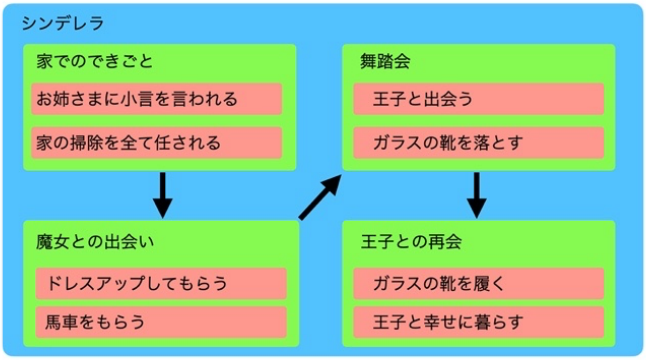

図 3　ストーリー，シーン，イベントを表す例

それぞれのシーンは物語世界での時間（以後「物語時間」と呼ぶ）の中で順序が定められているが，コンテンツの消費者に情報が提示される（物語が語られる）順序は必ずしも物語時間の通りではない．たとえば，劇的な効果を狙って，マンガの冒頭にクライマックスの場面が描かれたり，小説の途中に過去を振り返るモノローグが挿入されることがあるが，物語時間はそういった情報提示の順序とは独立した仮想的なタイムラインである．

シーンの順序は，同じ物語時間のタイミングで，別の場所で並行して物語が進行する場合など，分岐する可能性がある．たとえば，捕らえられた味方を助けるために，二人で手分けして手がかりを探すなどである．図 2 はシーンが分岐している例を示している．

シーンは入れ子にすることもできる．たとえば，「ダンジョンを攻略する」というシーンに，「スライムたちとの戦闘」「怪我をした冒険者仲間との遭遇」などのシーンが入れ子になるなどである．図 3 はシーンが入れ子になっている例を示している．

イベントは，シーンのように分岐や収束，入れ子になることはなく，シーン内では同じ物語時間に二つ以上のイベントが同時に起きることはない．

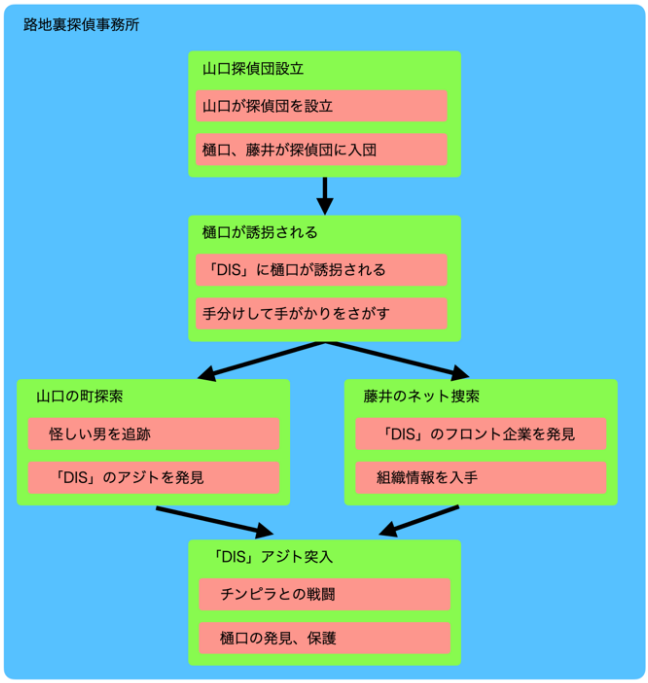

図 1　シーンが分岐している例

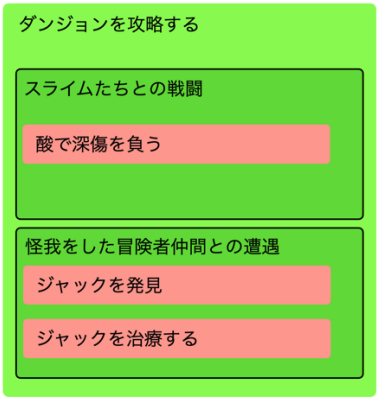

図 2　シーンが入れ子になっている例

なお，ストーリーには，物語を進行する上で重要な情報が全て明示されている．物語をコンテンツにする際に隠される情報なども全て記載しておく．これは，コンテンツの表現方法に焦点を置かず，ストーリー自体の把握を目的としているためである．

3.2 ストーリー要素

ストーリーには，登場人物，場所，思想，組織，物などの要素が登場する．これを総称してストーリー要素と呼ぶ．主人公をはじめとする登場人物や組織のほか，物語の進行

a 「大辞泉」（小学館）による定義より抜粋．

Vol.2021-GN-112 No.14
Vol.2021-CDS-30 No.14
Vol.2021-DCC-27 No.14
2021/1/25

に必要な品物をストーリー要素として記述する．たとえば「勇者の剣」が入手できなければ魔王を倒すという目的が達せられない場合，「勇者の剣」はストーリー要素として記述しなければならないだろう．

ストーリー要素は自身の状態や性質を表す属性を持っている．これを要素の属性，または単に属性と呼ぶ．属性としては，名前，職業や年齢，性別などのほか，所属する団体，経験や戦闘能力を表す値，戦闘可能／不能のような状態，所持品などを考えることができる．

ストーリー要素間の関係もストーリーに登場する．要素同士の敵対関係や利害関係がこれにあたる．例としては，先輩と後輩，仲間，恋人関係などが挙げられる．

シーンに登場しないストーリー要素も物語世界には存在しており，物語が進行したことによる影響を受ける．たとえば，シーン「主人公と師匠の特訓」からシーン「ライバルとの対決」の間に１年が経過したとすると，「主人公と師匠の特訓」にはライバルは登場しないが，ライバルの年齢は１歳増えていることが考えられる．

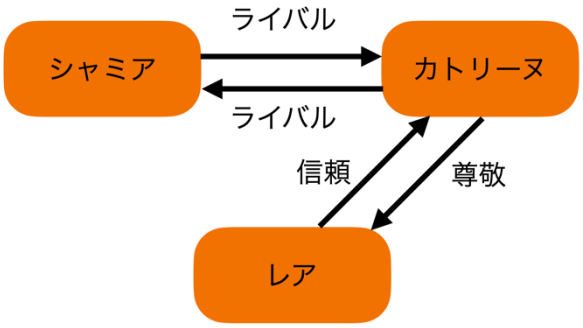

図 5　ストーリー要素と相互関係

3.3 シーンとイベントの役割

イベントは，ストーリー要素の変化とストーリー要素間の関係の変化を表す役割を持ち，登場人物の行動や，物語世界での現象など，ある特定の変化を一つだけ表す．シーンはイベントを実行することで，実際にストーリー要素の属性，相互関係を変化させる．

ストーリー要素と要素間の関係が構成するネットワーク状の構造は，静的なデータの集合として捉えることができる．たとえば，ストーリー要素を登場人物，要素間の関係を人間関係とすると，人物相関というデータの集合と考えられるだろう．この静的なデータに次々にイベントを適用することでストーリーの属性や要素間の関係を動的に書き換えることができる．図 5 はその例を示している．

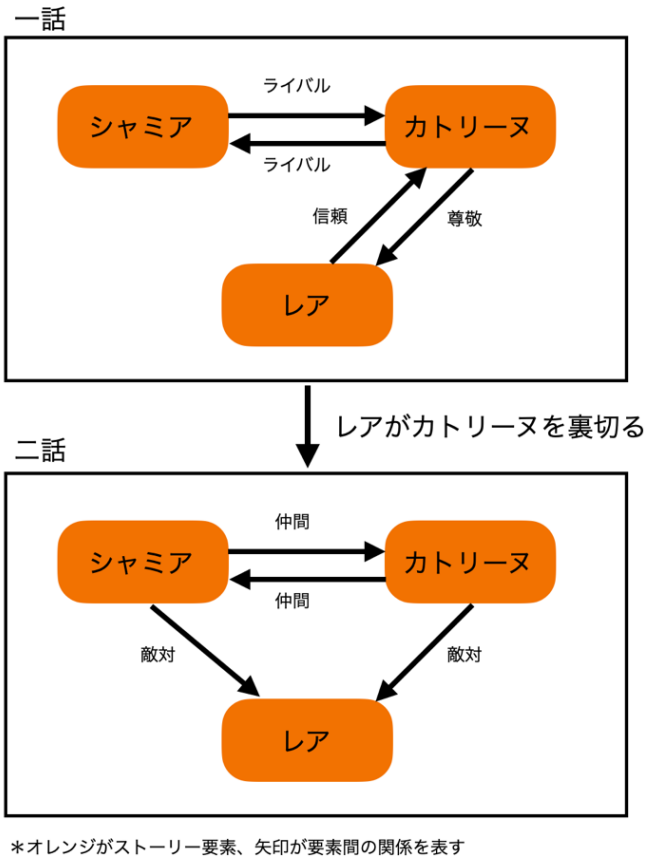

図 4　動的に変化するネットワーク状のデータの例

4. ストーリー記述言語

この章では，提案手法を実現させるために開発するプログラミング言語の概要と，試作システム「Story Designer」の記述方法について述べる．

4.1 概要

Story Designer は，シーンとイベント，ストーリー要素とその相互関係を記述するために開発した言語である．ストーリー要素の状態の遷移の表示や，シーンごとの登場人物の関係の表示，ストーリーの矛盾点の指摘などを行う機能を実装することを目的とする．現在，試作システムを python で実装している．

Story Designer は以下の情報を記述することができる．

(1)　ストーリー要素の定義
(2)　ストーリー要素の属性の定義
(3)　述語（動詞）の定義
(4)　あらすじ
(5)　ストーリー

上記のうち，(1)〜(3)の３つは，あらすじとストーリーの記述に使用するために定義され，あらすじはストーリーの矛盾を検証するために記述される．

Vol.2021-GN-112 No.14
Vol.2021-CDS-30 No.14
Vol.2021-DCC-27 No.14
2021/1/25

4.2 ストーリー要素と属性の記述

ストーリーを構造化するにあたり，ストーリー要素と要素の属性が記述できる必要がある．図 6 は，試作システムでの要素の属性とストーリー要素の定義記述方法を示している．attribute で始まる記述が要素の属性の定義，element で始まる記述がストーリー要素の定義の記述例である．

```
attribute type{
    mold : enum {
        human, creature, artifact, nature
    };
}

attribute age{
    mold : number;
    conditions{
        this >= 0;
    }
}

element momotaro {
    type : human;
    name : momotaro;
    parent : grandma, grandpa;
}

element oni {
    type : creature;
}
```

図 6　要素の属性とストーリー要素の定義例

要素の属性を定義では，型の定義と束縛される値の条件を記述することができる．ストーリー要素の定義では，要素の属性はキーとバリューのペアによって，ストーリー要素は要素の属性のリストとして表現している．

ストーリー要素は定義を行わない限りストーリーに存在することはできない．ストーリー要素はそれぞれの要素がストーリー構造化に必要不可欠な情報であり，不確実な情報や推測，推論によって定義されることがないようにするためである．たとえば，「サムが主人公の手助けをする」というイベントを記述する時に，サムを定義していなければ，人間なのか，犬なのか，組織なのか分からない．この情報を明確にしないままストーリーの記述を行うと矛盾点の検証や要素の状態の確認をうまく行えなくなってしまう．

逆に，要素の属性は定義を行わなくてもイベントによってストーリー要素に付与される場合がある．たとえば，「ジョージが死んだ」というイベントを記述したい場合に，わざわざ，新たな要素の属性「live」を定義し，その属性をどこかのシーンで「true」にするために，ジョージが生まれたイベントを記述するのは不自然である．さらに，もしジョージの生まれるイベントを記述したとしたら，全ての登場人物の誕生イベントを記述しなければうまく整合性が取れない可能性も高く，記述の手間が増える．こういった場合，「live」属性の定義は行わず「ジョージが死んだ」というイベントを発生させた際に，ストーリー要素ジョージのみに「live」属性を付与させる．他の登場人物の「live」属性は未定義となってしまうが，ジョージ以外の「live」属性はストーリー上明記するほどの重要な情報ではないことを意味する．実際の物語においても，「ウルトラマンが体感している３分の長さ」など，重要でない情報は明記されないが，それが物語を破綻させることはないのでストーリー構成上も問題はないと言えるだろう．

4.3 シーンの記述

シーンが持つ情報は以下のとおりである．なお，入れ子になるシーンを子シーンと呼称している．

- シーン名
- イベントと子シーン（複数）
- イベントと子シーンの順序
- 他のシーンとの時間的関係（複数）

イベントの順序情報をシーンが持つことにより，シーンがストーリーの時間情報をまとめて保有することができ，プログラミング時の複雑さを減らす役割を持つ．なお，試作システムでは他のシーンとの時間的関係は記述せず，記述した順番とシーンの順番が同じになるよう内部で処理する．図 7 は試作システムにおけるシーンとイベントの記述例である．

```
scene birth_of_momotaro {
    grandma / get / the_peach;
    grandpa / cut / the_peach;
    momotaro / is_born;
}

scene road_to_onigashima {
    momotaro / grow;
    momoatro / go / onigashima;
    scene increased_strength {
        momotaro / get / inu;
        momotaro / get / saru;
        momotaro / get / kiji;
    }
    momotaro / attack / oni;
}
```

図 7　シーンとイベントの記述例

4.4 イベントの記述

試作システムでは，イベントは図 7 の「grandma / get / the_peach」のように，１つの文章として記述され，文章から情報を読み取って内部的に情報を格納しておく．なお，単語を「/」で区切ることで，解析を簡略化させている．

主語や目的語には，必ずストーリー要素または要素の属性を記述する．述語は，主語や目的語に変化を与えることができる．図 9 に述語（動詞）の定義の記述例を示す．

Vol.2021-GN-112 No.14
Vol.2021-CDS-30 No.14
Vol.2021-DCC-27 No.14
2021/1/25

```
verb is_born {
    conditions {
        subject.type == human;
    }
    results {
        subject.live = true;
    }
}

verb respect {
    conditions {
        subject.type == human;
        object.type == human;
    }
    results {
        relationship(subject, object, 50)
    }
}
```

図 9　述語の定義の記述例

述語の定義は述語を実行する条件と実行の結果を記述する．述語ではストーリー要素の関係の変化を記述することも可能である．要素の関係を記述するには，図 9 の下部のように関係の変化を述語の定義内に記述し，イベントにてその定義した述語を使用することで行える．要素の相互関係は属性とは違い，２つのストーリー要素に同時に付与させる情報であり，属性とは別の表現方法で実装される．インターネット上の資源に関する情報を表現するための言語である RDF[8]の記述方法を参考に，一方向からの情報を格納する．相互関係の程度を定量的に表現するため，図 9 のように数値が格納される場合もある．

定義されていない述語がイベントに記述された場合，イベントの生成は行われ，その述語が実行されるが，述語の実行時に主語や目的語に変更は加えられない．

4.5 あらすじの記述方法

あらすじの記述方法は，いわゆる「5W1H（when, where, why, who, what, how）」の情報をそれぞれ記述する方法である．図 9 は試作システムでのあらすじの記述例を示している．この「5W1H」はイベントの内部的な表現としても利用され，それぞれイベントが処理される際の条件とストーリー要素への影響を表す．それぞれが示す条件・結果は以下の通りとなる．

- when: 時間的な条件として，特定のシーンやイベントが実行している，またはまだ実行していないことを記述する．
- where: ストーリー要素のうち，土地や場所に関する要素の状態の条件を記述する．
- why: "who" の心理的な動機など，ある特定の変化が要素に起きていることを条件として記述する．
- who: イベントに存在するべき登場人物が記述される．
- what:このイベントが実行された際の，それぞれのストーリー要素への影響を記述する．

```
outline {
    momotaro_is_born {
        where : momotaro_house;
        who : momotaro;
        what : is_born;
    }
    team_momotaro_attack_oni {
        where : onigashima;
        who : momotaro, inu, saru, kiji;
        what : attack / oni;
    }
    momotaro_get_treasure {
        where : onigashima;
        who : momotaro;
        what : get / the_treasure;
    }
    momotaro_give_villager_treasure {
        where : the_village;
        who : momotaro;
        what : give / villagers / the_treasure;
    }
}
```

図 8　あらすじの記述例

- how:上記以外のストーリー要素が関係するイベントの条件を記述する．

「how」と「why」に関しては，ストーリーを構造化するにあたり，脚本を着色しコンテンツを充実させるための情報であることが多く，記述できるようにするかは検討の余地がある．なお，今回試作したシステムには「how」「why」２つの項目は導入されていない．「when」は図 9 では記述されていないが，このように特に記述のない場合の時間的順序は記述順をあらすじの順序としている．

4.6 期待される出力

本研究の目的であるストーリー作成の検証を達成するため，以下の機能が必要と考えた．

1. ストーリーの矛盾を指摘する機能
2. 物語時間を指定してストーリー要素や要素の属性，要素の相互関係がどのような状態かを表示する機能

(1) ストーリーの矛盾を指摘する機能

ストーリーにおける矛盾は，イベントが処理される際の条件である，「5W1H」の情報で判断される．しかし，物語は「浦島太郎が窒息しない」「サザエさんは歳を取らない」などの多くの矛盾を抱えているものであり，時にはそれが物語の重要な情報となり得るので，この機能は警告を出すにとどめ，システムの停止は行わないものとする．

試作システムでは，あらすじで記述された情報とシーンによってイベントが処理される度に状態が同一であることを確認することで機能を実装している．たとえば，図 9 のあらすじの記述で「momotaro_is_born」の箇所では，where を「momotaro_house」としているが，一方，図 7 のシーン「birth_of_momotaro」内のイベント記述を見ると，場所が「momotaro_house」であるという記述がないので，警告を出力する，といった処理が行われる．

Vol.2021-GN-112 No.14
Vol.2021-CDS-30 No.14
Vol.2021-DCC-27 No.14
2021/1/25

(2) 物語時間を指定してストーリー要素などの状態を表示する機能

物語時間を指定してストーリー要素などの状態を表示する機能は，特定のシーン，イベントが実行された時点でのそれぞれの状態を表示することで実現できる. たとえば，図 7 のシーン「birth_of_momotaro」が実行された時点での要素「momotaro」と get の関係にある要素を表示させると，何も出力されない（momotaro は get を一度も行っていないため）が，シーン「road_to_onigashima」を実行し，もう一度要素「momotaro」と get の関係にある要素を表示させると「inu」「saru」「kiji」が出力される，といった処理が行われる.

シーンの分岐がない場合は，指定されたシーン，またはイベントから，物語の最初にたどり着くまで遡りながら処理を行う．分岐がある場合は，最大で分岐の数と同じだけの状態を表示する.

5. 議論

5.1 設計した記述言語について

本稿ではストーリーを，ストーリー要素と要素の関係をプログラミング言語によって記述したものと捉える手法について述べ, この手法に基づいた試作システムを開発した. 試作システムでは，登場人物やアイテム，ストーリー自体をコンピュータで処理可能なデータとして記述することを可能にしたが、ストーリー背景の記述やシーンの分岐の記述は不可能である.

物語の背景知識や世界観は，存在しないはずのものが存在するなどのストーリーの矛盾を明らかにしたり，登場人物の行動への納得感を生み出すための情報になると考えられる．全ての背景知識と世界観を定量的な情報にすることはできないが，一部のみを記述することや，単にテキスト情報としてデータに加えておくなどの機能を追加する必要を考えるべきである.

試作システムはかなり汎用プログラミング言語に近い記述方法になっており，ストーリー作成を行う誰もが利用できる状態ではないと思われる．人間が記述されたプログラム自体を読むことでもストーリーが把握できるよう，自然言語に近い形式で，ストーリー要素やシーンなどの要素ごとにまとめて記述できるように改良して行きたい.

5.2 相対的な時間の記述

本研究では，今までのプログラミング言語では表現されていなかった，情報を相対的な時間で区切って表現できるデータを扱えるプログラミング言語を提案できた．物語時間は相対時間でのみ記述しているが，これはシーンの経過時間を絶対時間で定義する必要がないためである．たとえば,「ダンジョン攻略」というシーンが，物語時間で 1 時間かかるとしても、10 分で終わったとしても、時間が物語に影響を与えない限り不必要な情報だからである．一般のプログラミング言語でも，相対的な時間を取り入れることによって新たな記述方法が生まれる可能性はあるのではないだろうか.

5.3 ストーリー要素と属性の扱い

ストーリーを記述する際に，ストーリー要素は定義が必要だが，属性の定義は必ずしも必要ではない．この点についても検討しておくべきであろう.

ストーリー要素，属性のどちらにも言えることだが，定義を必要にすると記述が増えてストーリー制作の手間が増えてしまう反面，定義を不必要にするとスペルミスの確認が大変になったり，意図しない要素の出現を可能にしてしまうなどの問題が表面化してしまう.

試作システムでは，影響の大きさを考慮して，ストーリー要素には定義を必要とし，属性は定義をしなくてもイベントに記述できるようにした.

述語についても同様に，自然言語（試作システムでは主に英語）における述語（動詞）の種類は膨大であり，ストーリーの作成過程で現れるすべての述語を定義するのは大変な労力を要する．この観点から試作システムでは定義されていない述語を使用することができ，一切ストーリー要素を変化させない述語の記述を行うことができる.

5.4 物語や現実世界の知識の利用

試作システムの内部のデータ構造は RDF[8], JSON-LD[9] を参考に開発を行った. この枠組みを活用することよって，外部のデータを取り込む機能を追加することも可能であると考えられる．たとえば，Wikidata[10]のようなインターネット上のデータを入力することで，現実世界の常識や知識を物語にも適用することができるのではないだろうか.

自然言語で書かれた物語か，簡潔に物語の情報だけが自然言語で書かれた文書をそのまま構造化すればよいのではないかという考え方もあり，実際に形態素解析を利用した研究[7]もあった．しかし，自然言語処理を自動的に行う手法の問題点として，深層学習や機械学習で意味理解と文脈理解をすることは先行研究では実現されていない[11]．よって，ある程度人間が言葉の意味を与えることで，構造化を補助することが提案手法を現在の時点で実現可能にする手段だと考えている．今後、自然言語処理の技術が発展すれば、物語をテキストからデータ化できる可能性はある.

行動の意味や物の状態を人間は言葉で表してきた．それをプログラミング言語という言語でも表すのは難しいことなのだろうか．現状のプログラミング言語は言語と呼ぶにはあまりに記述できる要素や方法が少ない．もしも、プログラミング言語が本稿のような物語という情報だけでなく，物や概念などの情報の記述が容易になれば，プログラム自体が物語を伝える言語になりうるのではないか.

Vol.2021-GN-112 No.14
Vol.2021-CDS-30 No.14
Vol.2021-DCC-27 No.14
2021/1/25

参考文献

[1] 戀津 魁，シナリオ情報構造化システムによる映像コンテンツ制作支援基盤の構築，東京工科大学 博士論文 (https://ci.nii.ac.jp/naid/500001323069/), 2017.
[2] K. M. Brooks，Programming narrative，https://dl.acm.org/doi/10.5555/832278.834387 IEEE Symposium on Visual Languages, 380-386, 1997.
[3] 長久 勝，FDR による RPG シナリオの検証，トップエスイー修了制作，http://files.topse.jp/posters/pdf/presen-nagaku.pdf, 2008.
[4] W.Dee，吉里吉里２リファレンス，https://krkrz.github.io/krkr2doc/kr2doc/contents/index.html，（2020 年 12 月に参照）.
[5] indent, Nola 小説家専用エディタツール, https://nola-novel.com/, 2020.
[6] Tasuku Oikawa, WorldType docs, http://doc.world-type.com, 2017.
[7] 高橋由樹，松本大貴，堀江夏子，後藤滋文，中村章人，金子満，塚本享治，形態素解析を利用したアニメーション脚本の XML 化とその会話編集システム，第 66 回全国大会講演論文集 3, 149-150, 2004.
[8] W3C，RDF 1.1 Primer，https://www.w3.org/TR/rdf11-primer/，(2020 年 12 月に参照)
[9] W3C，JSON-LD1.1，https://www.w3.org/TR/json-ld11/，(2020 年 12 月に参照)
[10] ウィキペディア財団，Wikidata，https://www.wikidata.org/wiki/Wikidata:Main_Page，(2020 年 12 月に参照)
[11] 新井紀子，AI vs. 教科書が読めない子どもたち，東洋経済新報社，2018.