Wiki の本質とは何か

江渡　浩一郎*

1　はじめに

人文情報学の観点から見て，Wiki は非常に重要な概念であると思われる．

Wiki はインターネット上のコラボレーション・システムであり，Wiki を使うと手軽にみんなで Web 上の情報を共同編集することができる．グループのコミュニケーションに Wiki を活用することによって，お互いの持つ知識を共有することができる．Wiki はそのようなツールである．

しかし，たしかに Wiki はそのようなコミュニケーション・ツール，あるいはコミュニケーション・システムとしての側面があるが，それだけではない．Wiki はまた，人と人のコミュニケーションの変革，または人と人との関係性を捉えなおすための思想としての側面も持っている．Wiki は，技術的な意味合いとしてのシステムと，人がとる行動への指針としての文化的なシステムとの両面を持ち，そのようなシステムと思想が一体化した「概念」として Wiki を捉えるべきである．

本論では，Wiki が発祥した経緯をその源流に遡ることによって明かにし，そこから Wiki の本質が持つ意味を明かにすることを試みる．

2　Wiki の現状

近年，Wiki をめぐる状況は大きく変化している．Wiki という言葉をしばしばメディアで目にするようになった．大きな理由の一つは *Wikipedia*[1] である．インターネット上の誰でも編集できる百科事典 Wikipedia は，本当に様々な情報が蓄積されており，まさにインターネット上の集合知を体現する存在となっている．

Wiki を扱う企業も多数現れてきた．日本でも Wiki サービスを提供する会社が登場し, Wiki サービスを提供する会社 *JotSpot*[2] は Google に買収された．Wiki についてのシンポジウムである *WikiSym* (International Symposium on Wikis)[3] や，Wikipedia についてのシンポジウムである *WikiMania*[4] など，様々な Wiki 関連のイベントが開催されるようになってきた．研究の分野にも，Wiki に独自の拡張を加えたり，Wiki 的な機能を別のメディアで実現するなど，様々な論文が現れてきている．

このように Wiki という言葉は大きく普及し，様々な領域で用いられるようになってきたが，その言葉が持つ意味の背景は，用いる人によって少しづつ異なっているように感じられる．

例えば，Wikipedia は Wiki の代表例として扱われることが多く，時には Wiki を Wikipedia の略称だと考えている人もいる．この場合には，Wikipedia は Wiki という広い概念の一つの実例にすぎないと解説をすることができる．しかし，ではその時に Wiki という概念とは一体どのような意味を持つのか．

* 独立行政法人産業技術総合研究所 / 東京大学大学院情報理工学系研究科

Wiki と近い分野の言葉として**ブログ**がある．ブログは Web 上で日記を書くためのシステムである．Wiki とブログはよく並べて語られるが，ブログは一人で書くもの，Wiki はみんなで書くものという違いがある．しかし，一人で Wiki を使うこともできるし，一つのブログをみんなで共同で書くこともできる．つまり，システムの機能の違いではなく，システムの使い方の違いとなる．ブログは主に日付けによって情報を管理し，Wiki はページ名で情報を管理するという機能の違いもある．そこから，Wiki でページ名に日付けを使えば Wiki をブログのように使うこともできることになる．このことから Wiki をブログの上位概念として理解している人もいるが，この場合はどこまでを包摂しているのかという疑問が生じる．例えば Web 上の掲示板も Wiki で実現できるとすると，Wiki は掲示板の上位概念でもあるということになる．このように考えると，Wiki という言葉は実質的には意味が無くなるくらいに広い対象を意味することになってしまう．

このように，Wiki という言葉を取り巻く状況は曖昧である．何が Wiki という言葉の根本にあるのかについての共通理解が存在していないため，Wiki という言葉を使って議論をおこなっていても，すれ違いが生じる．そこで，Wiki という言葉がどこで生まれ，どのように発展していったのかを明かにすることによって，Wiki という概念を成り立たせている根本的な原理がどのようなものなのかを考察する．

3　クリストファー・アレグザンダーによるパターンランゲージ

Wiki の発祥の背景には，実は**クリストファー・アレグザンダー**の活動がある．そこでまずクリストファー・アレグザンダーの活動について解説する．

1964 年，クリストファー・アレグザンダーは「**形の合成に関するノート**」[5][6] という博士論文をまとめた書籍を出版した．これはデザインという主観的に行われるとされるプロセスを，数学的な手続きの集合に還元することを試みる，デザイン理論についての論文である．例えばヤカンという製品のデザインを行う際には，ヤカンが満すべき条件がある．例えば，水を入れられること，持ち手がついていることなどである．それらの条件さらに細かい条件に分けられていく．また同時にそれら機能的な条件だけではなく，見た目を良くするデザインにおいても同じく条件の集合として捉えることができる．それらの条件の間は，相反したり相関したりする関係を持つ場合がある．デザインの問題をそのような細分化された条件の集合に還元し，そこから最も条件を満す解を導き出すプロセスだと考えた．これはまさしくデザインに対する構造主義的なアプローチと言える．デザインを数学的に分析し，パラメータの集合に置き換える．そこから最適解を探す問題へと置き換えたわけである．

ここでデザインする対象を「ヤカン」ではなく「タウン」へと置き換えれば，都市計画を行うためのプロセスとして用いることができるようになる．しかし，実際にこのアプローチを都市に対して行った結果，この試みは破綻することになる．都市はヤカンのような製品とは違い，もっと複雑なプロセスで作られるからだ．「**都市はツリーではない**」[7] では，自然に出来上がった自然都市と，都市計画によって作られた人工都市との違いを分析し，全ての人工都市は不可避的にツリー構造を持ったものになることを明かにした．人間の認識能力には限界があり，全ての制約条件の集合を同時に考えることは不可能なのだ．自然都市には人工都市には無い，セミラティス構造を持つ．それは，都市に関わる複数の主体が同時に都市の生成に関与し続ける結果生まれた構造なのである．

パターンランゲージ(Pattern Language)[*1]は，そのような都市計画における議論を経て，一つの建築もまた

[*1] Pattern Language の日本語表記は「パタン・ランゲージ」「パターン言語」などと揺れているが，本文中に記す理由により「Language」は「ランゲージ」のままがよいと判断し，一語でパターンランゲージと表記することにした．

都市計画と同様に，参加型のプロセスで作られるべきだと論じている．アレグザンダーは，都市計画の分析において得られた視点を元に建築を捉えなおした．彼は，建築の利用者と実際に建築を設計する建築家との関係に着目した．建築家は利用者からどのような建築を使いたいかの意見を聞き，その意見を反映させた建築を建てる．しかし実際には利用者が実際に建築が建てられる前にどのような建築がいいかを言うのは困難であり，結果として建築家が提案する建築を利用者がそのまま利用するという状況になりがちである．建築家は自分の美的センスや，自分が実現したいと考える建築を優先し，利用者はただそれを使うだけ，つまり建築家優先で利用者不在の建築がたくさん生まれていた．

それは言わば，失敗した都市計画と同じである．従来のトップダウン型の都市計画，最初に計画があってそれを実現するだけの都市計画では，都市の住民の意見は十分に反映されない．その従来型の都市計画における反省を，いかに形のあるものにしていくか．そのために考え出されたのがパターンランゲージである．パターンランゲージは，利用者と設計者が共通で利用できる事例集のようなものであり，互いが同じパターンランゲージを使って対話することによって，共通の土台を作り上げることができる．そのように共同で設計をすることによって，利用者が設計に参加できるようにしたのがパターンランゲージである．つまり，パターンランゲージの根幹には「利用者の設計への参加」がある．

パターンランゲージという言葉は誤解を招きやすい．パターンランゲージという言葉は，「パターン」と「ランゲージ」というどちらも一般的な概念の組み合わせであるため，一見理解しやすそうに見える．しかし，この「パターンランゲージ」はその字面からすぐに理解できる概念だと考えてはならない．

「パターン」という言葉は，通常単に「繰り返しによる模様」や「模倣」といった意味でとられる．「パターン化している」などと使われる場合は否定的な意味になる．しかしパターンランゲージにおける「パターン」は，単なる繰り返しという意味ではない．自然界を見まわすと，自然にできた形には，ある一定のルールによって生成された形を見付けることができる．例えばヒマワリの花の中に規則的に並ぶ形を発見したり，泥が乾いていく時にできるひびわれに法則を発見したりする．そのような，自然の法則の中に潜む，形態を発生させるようなルールを称してパターンと呼んでいる．

また「ランゲージ」という言葉も，通常は文字で表現される言語という意味に捉えられるが，そうではない．自然言語は様々な単位の部品の集合から成り立っている．文字という最小単位から始まり，単語，文，段落，節，文章，本，全集，百科事典と様々な大きさの単位がある．それぞれの単位においても，例えば主語，述語，助詞，動詞，形容詞など様々な種類がある．個々の部品間の関係も有機的であり，様々な部品が相互に接続されることによって多種多様な意味が生まれる．そのような，様々な大きさ・種類の部品が集まることによって多様な意味が生まれるような構造全体を指して「ランゲージ」と呼んでいる．

そのようなことから，建築設計において繰り返し出現するような形態を，自然言語における辞書のような形に組織化して，再利用できるようにまとめたものがパターンランゲージであるということになる．

このパターンランゲージへの発展は，ある意味「転回」である．それまでの「形の合成に関するノート」における議論では，デザインを数学的な構造に分析し，詳細な条件の集合に還元し，その条件を満す値の集合として良いデザインを捉えるという実験を続けていた．そしてその理論をまた都市へもあてはめようとしていた．しかしそれは不可能な実験だった．そこで彼は，数学的な条件の集合としての要素に還元するのではなく，複数の主体が相互に関係を続けることから生まれる形に注目することになった．そこで作り出されたのがパターンランゲージである．このパターンランゲージにおける一つのパターンも，ある条件の集合として捉えることができるが，重要な違いは，数学的な条件の記述ではなく，自然言語における辞書のような記述へと変わったことである．自然言語における辞書には自ずと初めからその解釈にはゆらぎが生じる．それを解釈する人間の曖昧さを許容するような方法へと転回したのだった．ここで，意図的に曖昧な手法を選び直したという

ことを理解する必要がある．

このパターンランゲージの背景となる理論は，5 冊からなる著作集の一冊目の本「時を超えた建設の道」[8][9] にまとめられている．その二冊目の本「パターン・ランゲージ」[10][11] は，パターンを集めたカタログ，一種の辞書のような本である．これらの本は，パターンランゲージを理論的にまとめた本であり，最初は逆にとっつきにくいかもいれない．三冊目の「オレゴン大学の実験」[12][13] のような具体的な事例集から見ていく方がわかりやすいだろう．またアレグザンダーの建築にかかわってきた日本の建築家中埜博氏の記録「パターン・ランゲージによる住まいづくり」[14] は，日本の建築状況においてパターンランゲージを実践した大変わかりやすい記録となっており，参考になる．

例えば「原寸大の設計」という手法が紹介されている．ある場所に家を立てる際に，図面上で配置を考えるが，それだけでは実際にどのような配置になるのかが一般の人にはわかりにくい．そこでその図面を持って現場に行き，地面に杭を打ったり線を引くことで原寸大の図面を書いていく．そのように原寸大に広げることによって見えてきた問題点を，その場で修正する．内装も同様に，家具と同じ大きさの段ボールを配置するなどして，どのように見えるかを確認する．このようにして，図面で設計したものを現場で確認しながらその場で設計を直していく手法である．

このように，パターンランゲージは実際には単なる辞書のようなものではなく，このような建築家と利用者をつなぐための様々な工夫の集積として機能するものである．建築はソフトウェアと違って，一度建ててしまったものはそう簡単には変えられない．そのため，建築では設計と施工を簡単には行き来することはできないが，そのような建築という世界においてもなんとか設計と施工を行き来することができるように工夫したものが，パターンランゲージである．

4　パターンランゲージのプログラミングへの応用

1986 年, 第一回目の *OOPSLA* (Object-Oriented Programming, Systems, Languages, and Applications), オブジェクト指向プログラミングに関する学会が開催された．

オブジェクト指向とはプログラミングにおける方法論の一つである．それまでのプログラミング言語は，主に手続きに注目して記述するものであり，手続き型プログラミング言語と呼ばれていた．オブジェクト指向とは，対象となる物の性質に注目して記述することによってプログラムを理解しやすくするという方法論である．このオブジェクト指向という当時新しく出てきた方法論は，ソフトウェア開発を大幅に改善する手法として熱狂的に迎えられていた．ちょうどそのような時期に第一回目の OOPSLA が開催され，現在オブジェクト指向の大家と呼ばれるような開発者も多数参加していた．その発表者の中に，ケント・ベックとウォード・カニンガムの二人がいた．

第二回目にあたる 1987 年の OOPSLA において，ケント・ベックとウォード・カニンガムの二人は「**オブジェクト指向プログラムのためのパターンランゲージの使用**」という論文を発表した [15]．彼らはこの論文で，プログラミングへのパターンランゲージの導入を提唱した．

従来の手続き指向型だったプログラミングにおける設計の方法論から，オブジェクト指向の設計に移るには，設計における方法論そのものを新しくする必要がある．特に，ユーザ・インタフェースの設計に重きを置く必要がある．そのために，建築の分野で提唱されていたパターンランゲージを，オブジェクト指向プログラミングに取り入れてみたというのがこの論文の趣旨である．実際には論文というよりテクニカルペーパーであり，2 ページだけと非常に短い．実践を行ってみての簡単な記録となっている．

ちょうど同じころに GUI が登場してきた．ソフトウェア設計の際に，実現すべき動作 (ロジック) と，それ

を操作するための手法に分けて設計するが，GUI によるグラフィカルなインタフェースの登場によってインタフェース設計が従来よりもはるかに重要になった．インタフェースの設計は，利用者がどのようにそのシステムを利用したいかという意見を取り入れることが重要となる．もちろん，システム設計には技術的な制約があり，その制限内で使い易さを追及する必要がある．これはちょうど建築における問題設定と同じである．パターンランゲージが建築の分野で行ったのと同じように，利用者と設計者の間の垣根を取り除く実験を，ソフトウェア開発の現場においても行ったということになる．論文より引用する．

> アレグザンダー氏は，家やオフィスというものは，実際にそこにいる人たちの手によって設計され，作られるべきだと提案している．氏がこう結論付けたのは，ある特定の構造への要求を一番よく知っているのは，彼ら自身だからだ．[15]

このように，利用者が望む建築をどのようなものにするべきかを一番よく理解しているのは利用者自身であるのと同様に，ソフトウェア開発においても利用者がどのようなシステムを利用したいのかを一番よく知っているのは利用者自身であるはずだからだ．パターンランゲージは，設計者と利用者との共通言語として機能し，お互いがこの言語を使うことによって，共通の土台で設計していくことができるようになる．

彼らはまずインタフェースにおけるパターンとして，下記 5 つのパターンを提案した．

1. タスクごとのウィンドウ（Window Per Task）
2. ウィンドウに対してペインはできるだけ少なく（Few Panes Per Window）
3. 標準ペイン（Standard Panes）
4. 短いメニュー（Short Menus）
5. 名詞と動詞（Nouns and Verbs）

このパターンをアプリケーション設計のチームに提供したところ，彼らは 1 日ほどでよくできたインタフェースを記述することができた．

インタフェース設計の後に，そのシステムをユーザが認識する世界におけるオブジェクトへと分割していく．次にそれをファイルフォーマットなどの低レベルのプログラムへとコーディングしていく．このようにユーザの世界からコードの世界へと順に細分化していく形でパターンを考えていった．

5 HyperCard によるパターン・ブラウザ

論文が公開された 1987 年，Apple 社から *HyperCard* というソフトウェアが発売された．その 3 年前の 1984 年には，Apple 社より GUI を搭載したパソコン *Macintosh* が発売された．HyperCard は，その GUI を最大限に生かしたソフトだった．

HyperCard はカード型データベースソフトと呼ばれる．例えば図書カードのように，一枚のカードに書名，著者，出版社などを記録して文献目録を作ることがある．そのようなカードの形に情報をまとめたデータベースを簡単に作れるようにしたソフトである．*HyperTalk* というスクリプト言語を備えており，カードによるデータと共に，カードを操作するインタフェースなどの動的な仕組みを埋め込むことができた．そのようにデータとそれを操作するインタフェースを組で配付することができ，それを**スタック**と呼んでいた．

ケント・ベックは当時 Apple 社に勤務しており，アラン・ケイをリーダーとする Vivarium プロジェクトに従事していた．同時期にビル・アトキンソンによって開発が進められていたのが，コードネーム *WildCard*[16] であり，後に HyperCard という商品名で発売されるソフトウェアだった．

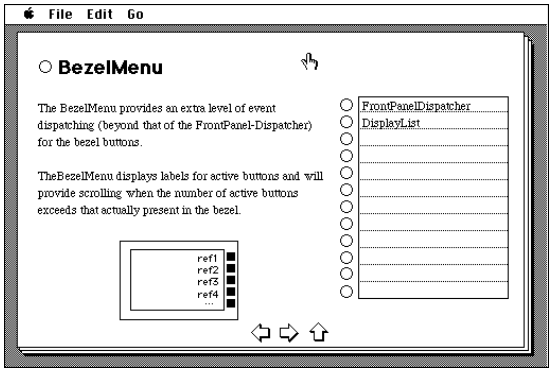

図 1　HyperCard によるパターン・ブラウザの画面

ウォード・カニンガムは，この HyperCard に衝撃を受けた．彼は，HyperCard を使って，自分が発見したプログラミングにおけるパターンを収集・整理するためのパターン・ブラウザを作り始めた [17]．実はこのパターン・ブラウザが発展して出来たのが，今の Wiki なのである [18]．

図 1 がこの HyperCard によるパターン・ブラウザの画面である[*2][19]．カードの左上には「BezelMenu」というタイトルがあり，その下に説明文章，図が挿入されている．右側には関連するカードへのリンク一覧が示されていて，クリックするとそのカードに飛ぶ．メニューから Edit を選ぶことで，WYSIWYG でそのカードを編集できる．

この HyperCard のスタックを複数人で共有する実験も行った．誰かが何かを編集すると，自動的にその編集記録が RecentChanges というカードに残り，リンクがはられる．そのため，その RecentChanges を見れば，他人がどのように編集したのかを知ることができる．

このように，HyperCard のスタックとして情報を複数人で共有しつつ，パターンの収集・編集を行っていった．

6　WikiWikiWeb の誕生

1991 年に *WWW* (World Wide Web) が誕生した．1993 年には *NCSA Mosaic* が誕生し，WWW は普及への道を歩みはじめた．それとほぼ同じ時期の 1993 年 12 月に，*CGI* (Common Gateway Interface)[20] という，Web サーバで動的な処理を行うための仕組みを備えた *NCSA httpd* 1.0 が公開された．

1994 年，ウォード・カニンガムは Perl と CGI を使って最初の *WikiWikiWeb* を作り，1995 年 3 月 25 日に彼の Web サイトである *c2.com* で公開した [21]．彼は 5 月 1 日に，パターンを議論するためのメーリングリストでこの Wiki をアナウンスした [22]．

彼はこの Web サイトに *Portland Pattern Repository* という名前をつけていた．彼が住んでいたポートランドの名前を冠したパターン貯蔵庫という意味である．しかし，Portland Pattern Repository はコンテンツにつけられた名前であり，編集できる仕組みを備えた Web サイトとしての名前が必要だった．そこでつけたのが WikiWikiWeb という名前である．このとき始めて，現在 Wiki と言われる手軽にページを編集できる仕組みを備えた Web サイトが出現した．

[*2] この HyperCard のスタックそれ自体を入手することができればより深い考察を行えると考えられるが，現時点で利用可能な資料はこのスクリーンショットしかない．

HyperCard によるシステムからの大きな違いは，インターネットに接続されていることである．そのため，パターン記述の場であると同時に，コミュニケーションの場としても機能するようになった．不特定多数のユーザがパターンを収集する際に，どのようにパターンをまとめていくのか．その収集する方法そのものについてのパターン自体もまたパターンとして記述していくことになった．このようにして，Wiki はパターン収集の場であると同時に，どのようにパターンを収集していくかのパターン，つまりメタなパターンの実践の場としても機能することとなった．

この最初の WikiWikiWeb を実現するためのソフトウェアは，*WikiBase* と呼ばれている．Perl による CGI スクリプトであり，ソースコードは 331 行と非常に短い．当時のソースコードは今も入手可能であり，自分のマシンで動かしてみることができる [23].

7 Wiki による文芸的プログラミング

この最初の Wiki の誕生には，興味深いメタな構造が隠されている．

ドナルド・クヌースは，**文芸的プログラミング** (Literate Programming)[24] という概念を提唱した．文芸的プログラミングとは，コードを説明する文章中にプログラムそのものを埋め込むことによって文章とプログラムを一体化できるようにするという概念である．Knuth は実際にこの概念に基いた *WEB* という処理系を作り，その処理系を用いて *TEX* を作った．

ウォード・カニンガムはこの文芸的プログラミングに影響を受け，Wiki ページ中にプログラムを埋め込み，Wiki とプログラムを一体化できるようにする *HyperPerl* というシステムを作った．まず，ある Wiki サイトの Wiki ページにコードの断片を分散して書いていく．出発点となるページを指定し，そこからページ間接続のリンクを辿るようにして次から次へとページを移っていき，それらのページに埋め込まれているコードを抽出していく．最終的には一つの Perl のスクリプトを生成する．

WikiBase は，実はそのようにして HyperPerl を使って書かれていた．つまり，世界最初の Wiki エンジンである WikiBase は，Wiki による文芸的プログラミング環境 HyperPerl で書かれていた．つまり，Wiki は Wiki を使って書かれていたというメタな関係にあった．

もちろんこれは卵と鶏の関係にある．ちょうど C 言語で C コンパイラを書く手順と同じように，本当に最初の Wiki は手で書いたのだろう．その基盤の上に HyperPerl を構築し，その HyperPerl を用いて WikiBase を再構築したものと考えられる．

この HyperPerl の Wiki サイトには，現在もまだアクセスすることができる [25]．実際に，Wiki ページ一つ一つに WikiBase のコードの断片が分散して書かれており，それらのページを順にたどることによって WikiBase のコードが生成される様子を説明の文章と共に読むことができる．

誰でもコードを編集できるのならセキュリティの面が気になるが，非常にコンパクトなコードの断片が説明文章と共に記述されているため，何かしようとしてもその意図はすぐに見破られてしまい，悪意のあるコードを入れることは難しいだろう．非常に短くシンプルなコードであるために，そのコードのシンプルさそのものが，最大の防御になっていたと考えられる．

このように最初の WikiWikiWeb のコードである WikiBase は，説明文章と共に誰でも編集できる形で公開されていたため，同じ仕組みを自分のサイトで動かしたり，自分なりに改造することは難しくなかった．このようにしてウォードの WikiWikiWeb と同じような機能を持つ Web サイトが登場し，それらは Wiki サイトと呼ばれるようになった．また，WikiWikiWeb と同じような機能を持つソフトウェアを Wiki クローンや Wiki エンジンと呼ぶようになっていった．そのようにして，固有名詞としての WikiWikiWeb は徐々に一般

名詞としての Wiki へ変化していき，現在では小文字の wiki と表記されるようになっていった．

8 Wiki はプログラムである

HyperPerl では，一つの Wiki サイトにコードの断片がちらばっていて，それが最終的に一つのプログラムになる．つまり，一つの Wiki サイトが一つのプログラムとなっているわけだ．このことから，Wiki とプログラムの関係について見えてくるかもしれない．元の C2 Wiki に戻って考えてみよう．一つ一つの Wiki ページは，WikiName と呼ばれる形式でページ名がついている．このページ名の形式は，オブジェクト指向におけるクラス名の形式と妙に似ていると思ったことはないだろうか．このように解釈すると素直に理解できるかもしれない．C2 Wiki という Wiki サイト自体もまた，一つのプログラムなのであると．そこで作られるのはいわゆるプログラムではないが，一つのプログラムのような存在を，みんなでよってたかって更新しているのだと．

Wiki はあまりに柔軟なシステムなので，それが一体何なのかが捉えにくいのだが，このように考えてみると理解できてくるかもしれない．つまり Wiki とは，一つのプログラムをみんなで共同編集するシステムだったのだと．

Wiki を更新していると，情報をどのように整理しようか迷うことがあるかもしれない．そのときは，このことを頭にいれるといいかもしれない．一つの Wiki サイトは，一つのプログラムであるかのように考えて更新する必要がある．まるでプログラムを共同編集するように Wiki を共同編集することによって，始めて Wiki のポテンシャルが生きてくるのだ．

9 エクストリーム・プログラミング

パターンランゲージのプログラミングへの応用は，一つは**デザインパターン**と呼ばれる流れへとつながる．「オブジェクト指向における再利用のためのデザインパターン」[26] では，プログラミングにおいて繰り返し表われる記述が辞書のように集められている．このデザインパターンは現在ではプログラマーの間の共通言語として機能するにいたっている．しかし，当初考えられていた，ユーザと開発者を結ぶ共通言語としての理念は失われてしまっているように見える．

デザインパターンを超え，もう一つの流れとして *XP* (エクストリーム・プログラミング, Extreme Programming) が生まれた [27]．これは再びケント・ベックとウォード・カニンガムの二人が提唱するソフトウェア開発における方法論である．このエクストリーム・プログラミングが提唱する手法の中に，ユーザの設計への参加が含まれている．それは，開発しようとするシステムがどのように動くべきなのかを一番よく知っているのはユーザだからである．ここではまさしくパターンランゲージの主題が繰り返されている．

10 Wiki と XP の理念

ケント・ベックは「エクストリーム・プログラミング」[27] 第二版の第 23 章「時を超えたプログラミングの道」において，建築設計について語っている．彼の曾祖父は大工だった．彼の家族が引っ越すたびに，自分の家族のための家を設計して建てていた．アレグザンダーの夢は，そのような空間設計の能力を取り戻させることだった．

彼はまたアレグザンダーの失敗についても書いている．建築家という職業はあまりに昔から存在しており，

社会的な役割が固定化してしまっている．建築家は自分たちの力を放棄せず，また利用者は要求する術を知らなかった．そのためアレグザンダーは最終的には建築家と利用者の間のバランスをとることに失敗したと．

しかし，ソフトウェア開発における開発者と利用者の関係は，まだ固定した関係には至っていない．誕生して間もないコンピュータの世界であれば，その開発者と利用者という社会的な関係を含めて新しく定義しなおせるかもしれない．ケント・ベックは「新たな社会構造を作る機会がある」と語っている．つまり XP の本当の目的は「新たな社会構造を作る」ことにある．

Wiki の本当の目的も，XP と同じようにパターンランゲージの応用から生まれている．Wiki はソフトウェア開発とは直接の関係が無い分野においても使われるようになってきているが，利用者が同時に設計者でもあるというコンセプトはそこでも生き残っている．建築の世界では受け入れられなかった設計者と利用者の間のバランスをとるという試みが，この Wiki の世界では受け入れられているように見える．

このことから，「**パターンランゲージの概念を応用した Web 上のシステム**」というのが Wiki の本質であると考えられる．同じようにパターンランゲージから発展した概念として，XP があり，両者は同じ出発点を持っているという意味で，兄弟のような関係にあると言えるだろう．

11 おわりに

本稿では，ウォード・カニンガムが Wiki を作り上げた経緯を出発点とし，どのように Wiki が発展していったのかを概観した．またその前提としてのパターンランゲージに着目し，そのパターンランゲージにおいて考えられていた理念が，Wiki にどのような形で残っているのかを記述した．Wiki は単なるシステムではなく，利用者と設計者のバランスをとるという目標設定を内在した概念であり，その意味で単なる技術的な機能の集合には還元できない．また Wiki とプログラミングには非常に深い関係があり，Wiki サイトを構築することとプログラムを構築することには強い相関関係があることがわかった．

最後に，柄谷行人による「隠喩としての建築」改訂版あとがきから引用する．

> 建築のモダニズムは重工業的段階に生じたものである．単純化すれば，彼らの問題はいわば，いかにしてスチールとコンクリートとガラスを芸術に取り込むかということであった．それに対して，今日われわれが出会っているのは，ポスト工業的段階での最先端情報技術である．そこで若い建築家たちがヴァーチャル建築にひきつけられるのは無理もない．そして，それが新たな可能性を与えることは疑いがない．しかし，それがかつてのモダニズムのようなインパクトを与えないのはなぜか．そこに，かつてのモダニストがもっていた倫理性と社会変革のヴィジョンが致命的に欠けているからである．[28]

現代の情報技術の発展は目覚ましいが，そのように発展した技術は，果して適切に他の分野にも影響を与えているのだろうか．最先端の情報技術を生かした建築があるとすると，それはどのようなものになるか．そのような建築は，人の住む社会をどのように変えていくことができるか．本論は背景にはこのような問いがある．

そのようなことを考えつつ，行き付いた先が Wiki だった．Wiki を出発点として，情報技術を生かした建築について考え直す．今後はそのような活動へと幅を広げていきたいと考えている．

謝辞

本論文は，WalWiki 作者の塚本牧生氏，SHIMADA こと島田慶樹氏との共著の論文から発展し生まれた [29]．また，未来心理編集長の萩原徹太郎氏には，本論文の前身にあたる「なぜそんなにも Wiki は重要なのか」[30] の執筆にあたり，多大なる協力を頂いた．ここに感謝の意を表する．

参考文献

[1] Wikipedia: . http://www.wikipedia.org/.

[2] JotSpot: . http://www.jot.com/.

[3] WikiSym: . http://www.wikisym.org/.

[4] Wikimania: . http://wikimania2007.wikimedia.org/.

[5] Alexander, C.: *Notes on the Synthesis of Form*, Harvard University Press (1964).

[6] クリストファー・アレグザンダー：形の合成に関するノート，鹿島出版会 (1978).

[7] Alexander, C.: A City is not a Tree, *Architectural Forum*, Vol. 122 (1965). http://www.patternlanguage.com/archives/alexander1.htm.

[8] Alexander, C.: *The Timeless Way of Building*, Harvard University Press (1979).

[9] クリストファー・アレグザンダー：時を超えた建設の道，鹿島出版会 (1993).

[10] Christopher Alexander, e. a.: *A Pattern Language: Towns, Buildings, Construction*, Oxford University Press (1977).

[11] クリストファー・アレグザンダー他：パタン・ランゲージ―環境設計の手引，鹿島出版会 (1984).

[12] Christopher Alexander, e. a.: *The Oregon Experiment*, Harvard University Press (1975).

[13] クリストファー・アレグザンダー他：オレゴン大学の実験，鹿島出版会 (2000).

[14] 中埜　博：パタン・ランゲージによる住まいづくり，井上書院 (1988).

[15] Beck, K. and Cunningham, W.: Using Pattern Languages for Object-Oriented Programs, *Technical Report No. CR-87-43* (1987).

[16] noop: . http://c2.com/cgi/wiki?WildCard.

[17] noop: . http://c2.com/doc/etymology.html.

[18] noop: . http://c2.com/cgi/wiki?WikiWikiHyperCard.

[19] noop: . http://c2.com/cgi/wiki?BezelMenu.

[20] McCool, R.: The Common Gateway Interface (1993). http://hoohoo.ncsa.uiuc.edu/cgi/.

[21] noop: . http://c2.com/cgi/wiki?WikiHistory.

[22] noop: . http://c2.com/cgi/wiki?InvitationToThePatternsList.

[23] WikiBase: . http://downlode.org/wiki/wiki.

[24] Knuth, D. E.: *Literate Programming*, Center for the Study of Language and Information (1992).

[25] noop: . http://c2.com/w4/wikibase/wiki.cgi?HyperPerl.

[26] Gamma, E., Helm, R., Johnson, R. and Vlissides, J.: Design Patterns: Abstraction and Reuse of Object-Oriented Design, *Proceedings of ECOOP'93* (1993).

[27] Beck, K.: *Extreme Programming Explained: Embrace Change*, Addison Wesley (1999).

[28] 柄谷行人：定本 柄谷行人集 第 2 巻 隠喩としての建築，岩波書店 (2004).
[29] 江渡浩一郎，塚本牧生，島田慶樹：Wiki 概念の多様性，*Proceedings of LinuxConference 2006* (2006).
[30] 江渡浩一郎：なぜそんなにも Wiki は重要なのか，Mobile Society Review 未来心理 Vol.7，モバイル社会研究所，pp. 50–57 (2006).