コンテキストを考慮した Web アプリケーションのためのアーキテクチャに関する研究

2014SE099 竹内優斗　2014SE101 田中涼

指導教員：野呂昌満

1　はじめに

カテゴリ検索を行う Web アプリケーションの多くは，カテゴリが静的に定義されており，このことが Web アプリケーションの使いづらさの原因となる．ユーザが自分の求める情報を理解していても，Web サイトの構造上，求める情報になかなかたどり着けないことがある．静的に定義されたカテゴリに対して動的にショートカットリンクを付加することで，EC サイトのような Web アプリケーションが稼動している Web サイトが使いやすくなると考えた．検索履歴に応じて，Web ページを動的再構成し，ユーザごとに最適な画面表示を行うことを目指す．動的とは，状態や構成が状況に応じて変化したり，状況に合わせて選択できたりする柔軟性を持っていることを指す．

現行の Web サイトの問題点として，カテゴリが直感的に理解できない場合に検索の手戻りがあり，ページ遷移の回数も多くなる点が挙げられる．通販サイトでは膨大な商品を取り扱うので，階層が深くなってしまう．階層を深くならないようにすると，ページ内のカテゴリが多くなり，求めている情報を探し出すのに時間がかかる．キーワード検索を用いても適切なページがすぐに発見できることが保証されている訳ではない．

本研究の目的は，ユーザの検索履歴に応じた Web アプリケーションの動的再構成による使いやすさの向上である．Web ページの動的再構成のポリシーを柔軟に変更できるアーキテクチャを構築する．すなわち，インタラクティブソフトウェアにおけるコンテキストを考慮したアーキテクチャを設計する．プログラムの構造やソフトウェアの記述が簡素化できるコンテキスト指向による実現が妥当であると考えた．ユーザごとに最適な画面表示を構築するために，検索履歴をコンテキストとしてとらえ，コンテキスト指向アーキテクチャを定義する．検索履歴をコンテキストとしてとらえることで，ページ遷移に関わるコードにおいて条件判定をする必要箇所が減り，それらの条件を変更した場合にもソフトウェアの保守が容易になる．設計したアーキテクチャに基づいて開発した Web アプリケーションが稼動する Web サイトでショートカットの動的追加の可能性について考察する．

ユーザの検索履歴をコンテキストとし，ショートカットリンクを動的に作成するために，本研究室で提案されている CSA/I-Sys[**?**] を拡張し，コンテキストを考慮したインタラクティブソフトウェアのためのアーキテクチャを設計する．CSA/I-Sys は，既存の MVC アーキテクチャの派生を統一的に取り扱ったものとして定義されている．MVC アーキテクチャの派生として MVVM パターンがあり，Model，View，ViewModel に分けられる．CSA/I-Sys を MVVM パターンの一般形としてとらえる．ネットショッピングを事例とし，ユーザの検索履歴をコンテキストとして着目する．これをアスペクトとして分離し，CSA/I-Sys を再設計する．コンテキストに応じて動的にショートカットを作成することによって，ユーザごとに最適な画面表示が構築でき，検索などのユーザの余分な操作を排除できることを確認する．ショートカットの追加は MVVM パターンの ViewModel を書き換えることで実現できる．

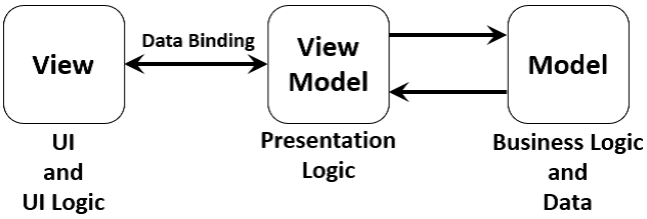

図 1　MVVM パターン

2　背景技術

2.1　MVVM パターン

MVVM パターンとは，アプリケーションのビジネスロジックとプレゼンテーションロジックをユーザインターフェースから分離することを目的としたソフトウェアアーキテクチャパターンである．MVC アーキテクチャの派生として MVVM があり，Model，View，ViewModel に分けられる．View はユーザインターフェースとユーザインターフェースロジックをカプセル化し，ViewModel はプレゼンテーションロジックと状態をカプセル化し，Model はアプリケーションのビジネスロジックとデータをカプセル化したものである．Model は安定的で不変であるという立場から，View が変更される場合に，Model と View の中間的な存在として ViewModel を用いることによってそれらの関係を独立に記述することができる．ViewModel が Model のパラメータを参照し，データ・バインディングによって ViewModel の値を View に反映させることで View を変える．MVVM パターンを図 1 に示す．

2.2　CSA/I-Sys

CSA/I-Sys は，アーキテクチャとアプリケーションの設計及びコードの理解，変更を容易にし，ライブラリやミドルウェアを，大きな粒度で変更する枠組みを提供する．メタアーキテクチャとしての参照アーキテクチャ，及び，具

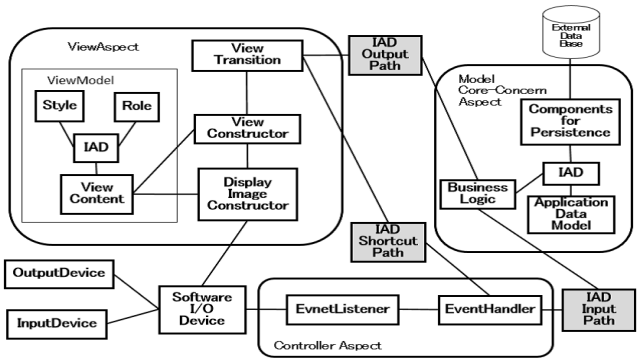

図 2 CSA/I-Sys のアプリケーションアーキテクチャ

象アーキテクチャをアスペクト指向アーキテクチャとして設計されている．CSA/I-Sys は既存の MVC アーキテクチャの派生を統一的に説明可能とするという観点から設計されており，参照アーキテクチャを詳細化したアプリケーションアーキテクチャとして図 2 のような構造となっている．ここで，あるアーキテクチャを説明可能とは，アスペクト指向アーキテクチャとして定義した共通アーキテクチャにおいて，特定の横断的関心事のいくつかを指定して織込み，そのアーキテクチャが生成されることを指す [?]．以下が，本研究に関わる各コンポーネントの説明である．

- EventListener：外部イベントを内部イベントに変換し，EventHandler に通知する
- EventHandler：EventListener から受け取った内部イベントを ViewTransition に渡す
- ApplicationDataModel：データの構造を決める
- ViewTransition：画面毎の再利用を可能にする
- ViewConstructor：画面部品を再利用可能にする

2.3 アスペクト指向技術

アスペクト指向は，横断的関心事をアスペクトとしてとらえ，独立したコンポーネントに分離する技術である [?]．オブジェクト指向において，複数のオブジェクト間にまたがり適切にモジュール化できないプログラムの振舞いのことを横断的関心事 (CrossCutting Concern:CCC) と呼ぶ．アスペクト指向技術を用いることで，ソフトウェアの開発効率や保守性が向上する．

2.4 コンテキスト指向プログラミング [?]

コンテキスト指向プログラミング (Context-Oriented Programming:COP) とは，コンテキストに依存した振舞いをモジュールとして定義するためのプログラミング方法である [?]．オブジェクト指向プログラミング (Object-Oriented Programming:OOP) では達成が困難な関心事の分離を可能にするアスペクト指向プログラミング (Aspect-Oriented Programming:AOP) などのプログラミング方法がある．AOP とは別のアプローチで OOP の限界を補うものとして COP があり，横断的関心事の中でもコンテキストに依存した振舞いに注目したプログラミング方法である [?]．コンテキストを何らかの方法で抽象化し，それに依存した振舞いの切り替えを統制的に行う言語要素を提供する．以下に，コンテキスト指向プログラミングにおける主な構成要素を挙げる [?]．

- 層 (Layer)：コンテキストに応じて変化する振舞いをモジュール化したもの
- オリジナルメソッド (Original Method)：プログラムの内，コンテキストに依存しない振舞いを実装したメソッド
- 部分メソッド (Partial Method)：基本メソッドの振舞いを変更するメソッド
- 活性化 (Activation)：プログラムに新たな層を追加すること
- 非活性化 (Deactivation)：活性化された層をプログラムから取り外すこと

3 コンテキストを考慮したアーキテクチャ

3.1 アーキテクチャ設計指針

ユーザの操作履歴に応じてショートカットを作成したい．ショートカットを作成する際に，複数のコンポーネントの振舞いを変更する必要があるので，これは横断的関心事となる．履歴をコンテキストとしたコンテキストアウェア実現が自然と考えた．CSA/I-Sys はアスペクト指向に基づき設計されているので，アスペクトとコンテキストを統一的に扱う．PBR パターンを適用して ShortcutAspect として分離する [?]．ショートカットのリンク先とどのページに付加するのかはコンテキストによって異なるので，ショートカットの作成はコンテキストに依存した振舞いである．ShortcutAspect は検索履歴に応じて振舞いが変化することから，コンテキスト指向技術を適用することでコンテキストとレイヤとしてモジュール化する．変化の要因となる記述をコンテキストとして独立させることにより，ソフトウェア全体の変更を容易にすることを指針とする．

3.2 コンテキストの定義

コンテキストはプログラムから観測することのできる外部環境やシステムの内部状態であり，本研究においては，ユーザの検索履歴とする．検索履歴をコンテキストとしてとらえることで，検索元ページと検索先ページが抽出でき，ユーザの求める Web ページが構築できると考えた．

3.3 ShortcutAspect の定義

ShortcutAspect は，ApplicationDataModel のページ構成要素を参照し，Context for Shortcut に応じて活性化するレイヤを切り替える (図 3)．ShortcutAspect は，Context for Shortcut，ShortcutCreationBehavior の 2 点のコンポーネントで構成されている．Context for Shortcut

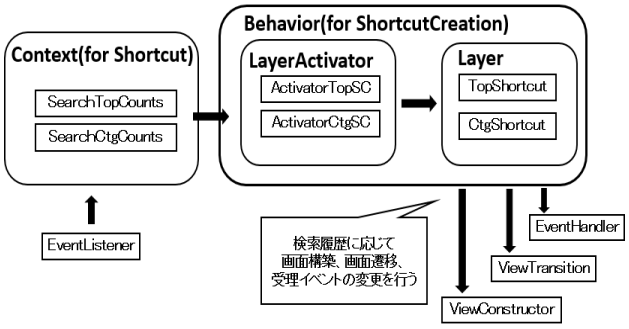

図 3　ショートカット作成のための ShortcutAspect

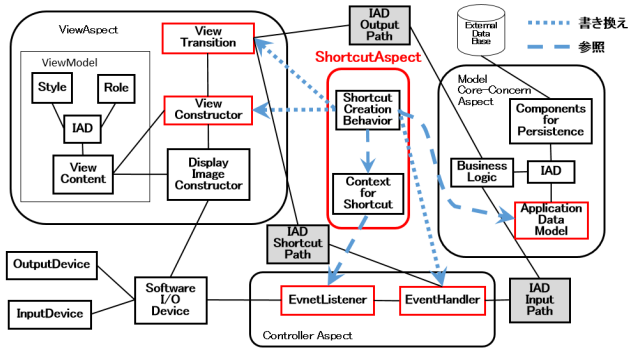

図 4　コンテキストを考慮したアーキテクチャ

は EventListener に送られてくるコンテキスト情報を参照し，LayerActivator が Context for Shortcut のコンテキスト情報を参照する．LayerActivator は参照したコンテキスト情報に応じて起動し，Layer を活性化させる．Layer は活性化した層に応じて画面構築，画面遷移，受理イベントの変更を行う．ショートカットリンクを付加したページを構築するために ViewConstructor，それに伴うページ遷移を実現するために ViewTransition を用いる．コンテキストによる振舞いは上記のコンポーネントによる振舞い，またはその組合せで実現できる．

3.4　コンテキストを考慮したアーキテクチャ設計

ユーザの検索履歴に応じてショートカットを作成するために CSA/I-Sys に ShortcutAspect を織込み，ViewConstructor 及び ViewTransition の振舞いを変更させる．コンテキストを考慮したアーキテクチャの構造を図 4 に示す．EventListener はユーザからの外部イベントを受けとるので，Context for Shortcut は EventListener に送られてくるコンテキスト情報を参照する．ShortcutCreationBehavior は Context for Shortcut に応じて ViewConstructor 及び ViewTransition のインスタンスに変更を加える．

4　事例に基づく考察

4.1　事例の設計

本研究では，ネットショッピングを対象にどのページからどのページへ，何回検索を行ったかによってショートカットを作成する事例を設計した．検索モデルとして，トップページで行う検索であるトップ検索と，カテゴリページ内で行う検索であるカテゴリ検索を定義した．ViewConstructor のインスタンスに対してリンク要素を追加し，ViewTransition のインスタンスに変更を加えることで，リンクとそれに伴うページ遷移が実現できるので，動的にショートカットを生成することが可能である．具体的な事例としては，トップ検索とカテゴリ検索によるショートカットを作成する事例を考える．表 1 では各検索と Context for Shortcut，Activator，ShortcutCreationBehavior，CSA/I-Sys 上の関連部分との関係を表している．Context はプログラムの様々な実体の実行に影響を与えるもので，Activator は振舞いが起こるきっかけ，Behavior は振舞いを表す．

表 1　各検索と Context for Shortcut，Activator，ShortcutCreationBehavior との関係

	トップ検索による ショートカット作成	カテゴリ検索による ショートカット作成
Context for Shortcut	検索元ページと 検索先ページの トレースを組とした集合	検索元ページと 検索先ページの トレースを組とした集合
Activator	同一カテゴリページ へ規定回数 トップ検索する	同一カテゴリページ から規定回数 カテゴリ検索する
Shortcut Creation Behavior	トップページに検索先 カテゴリページへの ショートカットを作成	カテゴリページに検索元 カテゴリページへの ショートカットを作成
CSA/I-Sys 上の関わる部分	EventListener EventHandler ApplicationDataModel ViewTransition ViewConstructor	EventListener EventHandler ApplicationDataModel ViewTransition ViewConstructor

トップ検索によるショートカット作成の場合，トップページからの検索回数 (SearchTopCounts) を Context for Shortcut とする．Activator は特定のカテゴリページへの検索回数が 3 回以上のときで (ActivatorTopSC)，TopShortcut レイヤを活性化し，画面構築，画面遷移，受理イベントの変更を行うことで，トップページにショートカットを作成する．カテゴリ検索によるショートカット作成の場合，カテゴリページからの検索回数 (SearchCtgCounts) を Context for Shortcut とする．Activator は特定のカテゴリページからの検索回数が 3 回以上のときで (ActivatorCtgSC)，CtgShortcut レイヤを活性化し，画面構築，画面遷移，受理イベントの変更を行うことで，カテゴリページにショートカットを作成する．各検索におけるショートカット作成の事例において，検索履歴に加えるかの判定として PageRank の判定と閲覧時間の判定を行う．PageRank とは各カテゴリページに定義され，トップページから静的に定義されたリンクを何回辿るかによって定義を行う．トップ検索における検索履歴に加えるかの判定として，検索結果ページの PageRank 判定を行い，2 以上の場合検索履歴に加え，2 未満なら加えない．カテゴ

リ検索における検索履歴に加えるかの判定として，検索元ページの PageRank 判定を行い，3 以上の場合検索履歴に加え，3 未満なら加えない．各検索において，検索結果ページの閲覧時間が 5 秒以上の場合検索履歴に加え，5 秒未満なら加えない．検索履歴に加えるかの判定をした後，トップ検索の検索履歴を SearchTopCounts に，カテゴリ検索の検索履歴を SearchCtgCounts に加える．カテゴリ検索によるショートカット作成の場合，ショートカットを作成するページは検索元カテゴリページから特定できる PageRank1 のカテゴリページとする．

この事例は MVVM の ViewModel を書き換えることで実現できる．このとき，Model は店舗を木構造で表したもので，ViewModel はページ構成要素，View は各 Web ページである．

4.2 考察

提案するアーキテクチャの検証として，ユーザの検索履歴に応じて Web ページにショートカットを動的に付加する事例を設計した結果，コンテキストに応じてポリシーの変更を柔軟に行えることが確認できた．提案するアーキテクチャに基づいて Web アプリケーションを開発する際，ショートカットを作成するためのリンク先ページとリンク元ページがカテゴリ分けされているという前提に立っているので，Model の形式としてカテゴリが定義されている必要がある．従って，エントリのカテゴリ分けがされていないブログ上などで稼働する Web アプリケーションに対しては，提案するアーキテクチャを適用できない．ユーザの検索履歴に応じてショートカットを作るのではなく，新たなページを作る際も，ViewModel だけを書き換えることにより実現できる．検索が複数回行われた際に，毎回 ViewModel を書き換えるのではなく，検索履歴に応じたショートカットを集めたページを構築することも可能となる．検索ではなく，ページ遷移の履歴をコンテキストとすることで，深い階層のページから浅い階層のページへの遷移を複数回行ったユーザに対して，浅い階層のページへのリンクを作成することも可能となる．これらの例は，Model と View の関係を独立に記述する ViewModel を書き換えることで実現が容易になるので，MVVM パターンを用いた．

ページ遷移に関わるコードは，その全てに条件判定が必要になり，それらの条件が変更した場合にソフトウェアの保守が困難になる．オブジェクト指向を用いた場合，ページ遷移にかかわるコードが散在してしまうので，コンテキスト指向を用いることで，プログラムの構造やソフトウェアの記述を簡素化できた．コンテキストとアスペクトを統一的に取り扱うことで，コンテキストに依存した振舞いに対しても変更が容易になる．MVC パターンを用いた場合，Model をそのまま View として表示することになり，Model に変更を加えない限り，View の変更が行えない．MVVM パターンを用いることで，変更したい View にあたる Model として ViewModel を作ることで View の変更が可能となった．

提案するコンテキストを考慮したアーキテクチャに基づいて Web アプリケーションを開発すれば，変更箇所が独立したモジュールとなり，変更が容易になる．ユーザの検索履歴に応じて Web ページを動的再構成でき，ユーザの余分な操作を排除できると考えられる．

5 まとめ

本研究では，Web ページの動的再構成のポリシーを柔軟に変更できるアーキテクチャの構築を目的とし，コンテキストを考慮したアーキテクチャを設計した．我々は，ページ遷移の回数が多くなるという問題を特定し，ページ遷移を定義している部分を動的に変化させれば余分な操作が排除できると考えた．設計したアーキテクチャに基づき，Web ページに動的にショートカットを作成する事例を挙げることで，ユーザの余分な操作を排除できることを確認した．コンテキストによる振舞いを実現するためのアスペクトを本研究室で提案されている CSA/I-Sys に織込むことにより，Web ページの動的再構成のポリシーを柔軟に変更することが容易になる．今後の課題は，他の解決策の調査，比較が挙げられる．

参考文献

[1] A. Esaka, M. Noro, and A. Sawada, “Design of Common Software Architecture as Base for Application Generator and Meta-Generator for Interactive Systems,” *IEEE*, vol. 2, pp. 323-328, 2017.

[2] M. Appeltauer, R. Hirschfeld, M. Haupt, and H. Masuhara, “ContextJ: Context-oriented Programming with Java,” *Computer Software*, vol. 28, no. 1, pp. 272-292, 2011.

[3] R. Hirschfeld, P. Costanza, and O. Nierstrasz, “Context-oriented Programming,” *Journal of Object Technology*, vol. 7, no. 3, pp. 125-151, 2008.

[4] 江坂篤侍, 野呂昌満, 沢田篤史, 繁田雅信, 谷口弘一, “コンテキストアウェアネスを考慮した組込みシステムのためのアスペクト指向アーキテクチャの設計，” ソフトウェア工学の基礎ワークショップ論文集，vol.24，pp.3-12，2017.

[5] 紙名哲生，“文脈指向プログラミングの要素技術と展望，” コンピュータソフトウェア，vol.31，no.1，pp.3-13，2014.

[6] 紙名哲生，青谷知幸，増原英彦，玉井哲雄，“ユースケースを用いた文脈指向ソフトウェア開発，” ソフトウェアエンジニアリングシンポジウム，pp.1-8，Sep. 2011.

[7] 千葉滋，“アスペクト指向ソフトウェア開発とそのツール，” 情報処理，vol.45，no.1，pp.28-33，2004.