序章

計算機システムの能力が向上するにつれて、その上で稼働するソフトウェアも複雑で大きなものとなってきている。そのようなソフトウェアを短期間で効率良く生産するためのアプローチとして、オブジェクト指向に基づくプログラミングが有望視されている。

オブジェクト指向をベースとしたプログラミングには、"a-kind-of" 関係に主眼をおいたクラスを対象とする手法と、"a-part-of" 関係を中心としたインスタンスを対象とする手法があり、これらの技術を用いたプログラムの部品化とその再利用が、ソフトウェアの生産効率を高めるものとして期待されている。しかし、クラスライブラリ (Smalltalk-80[25]、Booch Component[28] など) のようなクラスを対象とした手法では、拡張性が高い反面、クラス構造を熟知しなければ使いこなせないという難点がある [81]。一方、コンポーネントウェア (IntelligentPad[88] など) に代表されるインスタンスを対象とした手法では、プログラミングが容易な反面、拡張性の貧弱さが問題となっている [80][57]。そのため、現在のオブジェクト指向をベースとした開発フレームワークだけでは、多様なアプリケーションを短期間で開発するという目的には必ずしも十分ではない。

本論文では、従来のオブジェクト指向をベースとした開発フレームワークの問題点を検討した上で、それを解消するための方策の１つとしてクラスを対象とした拡張が可能なコンポーネントアーキテクチャを提案し、その実現例である Nuts フレームワークについて解説する。

クラスを対象とした拡張が可能なコンポーネントとは、プログラム構築をコンポーネントの結合によって容易に行えると同時に、サブクラス化によって拡張が容易なコンポーネントのことを言う。筆者は、このようなコンポーネントに基づくアーキテクチャを考案し、Nuts と名付けた。Nuts フレームワークとは、Nuts コンポーネントを用いたプログラミングの枠組をいう。

さらに本論文では、サブクラス化に依らずにプログラムの拡張が可能な新しいプログラム拡張手法として、コンポーネント差分プログラミングと呼ぶ手法を提案する。従来は、差分プログラミングによるプログラム部品の再利用は、クラスを対象とした再利用に限定されていた。しかし、クラスを対象とする手法は、クラス構造を熟知したプログラマでなければ難しい。これに対し、コンポーネント差分プログラミングは、扱いの簡単なブラックボックス的なコンポーネントの利用において差分プログラミングを可能とする手法である。

コンポーネント差分プログラミングでは、拡張機能の差分だけを組み込んだコンポーネント (ベクターコンポーネント[1]) をプログラム構造中に挿入することにより、コンポーネン

[1] ベクター (Vector) とは、悪性細胞にインターフェロン等の薬物遺伝子を運び込むウイルスの総称

トをベースとした差分プログラミングを可能にする。コンポーネント差分プログラミングを用いることで、拡張機能の追加、変更、削除をコンポーネント単位で簡単に行える。拡張機能の運搬を担うベクターコンポーネントは、Nutsコンポーネントの一種であり、通常のコンポーネントと同様の手法でフレームワーク中に組み込める。

以下、第1章では、オブジェクト指向プログラミングの概要と従来のオブジェクト指向をベースとした開発フレームワークを取り上げ、その問題点を整理する。

第2章では、ホワイトボックス的な拡張が可能なコンポーネントを提案する。また、コンポーネントを用いた差分プログラミングの概要について説明する。

第3章では、Nutsのコンポーネントアーキテクチャが規定する構造／制御モデルについて説明する。

第4章では、Nutsのクラス定義を通して、Nutsのアーキテクチャがどのように実装されているかを詳説する。

第5章では、Nutsの構造／制御モデルを拡張した透明コンポーネントの導入について説明する。

第6章では、透明コンポーネントを用いたコンポーネント差分プログラミングについて解説する。

第7章では、Nutsを用いたアプリケーション開発事例群を示す。

第8章では、Nutsに基づくアプリケーション開発事例として、オブジェクト指向に基づく物流業務支援アプリケーションフレームワークNavimosについて説明する。

第9章では、第7,8章の開発事例をもとにソフトウェア開発におけるNutsの有効性を評価する。また、デザインパターンや既存開発フレームワーク、プログラム拡張手法と比較する。

最後に結論を記す。