# ソフトウェアプラットフォーム構築技術

## Software-Platform Construction Technology

滝沢　治　■ TAKIZAWA Osamu
赤石　富士雄　■ AKAISHI Fujio
早瀬　健夫　■ HAYASE Takeo

**近年のソフトウェア開発では，開発規模が増大しているだけでなく，仕様が多様化かつ複雑化している。このような状況において，開発の生産性を向上させるためには，プロダクトライン型と呼ばれる開発方法が効果的であり，特にコア資産（ここでは，プラットフォームと呼ぶ）の構築が重要である。**

**東芝が開発した発電所向け監視制御システムでは，開発当初からプロダクトライン型の取組みを行ってきており，30年以上にわたって国内外に150システム以上の納入実績がある。また現在，プラットフォームの構造的な劣化を計測する技術の研究開発を進めているが，この技術により，プラットフォームの品質を維持できるようになる。**

In recent years, the specifications of software have shown drastic increases in variation and complexity with the progress of large-scale software development. Software product line engineering methods are an effective means of improving the productivity of software development. Construction of the core assets of the software product line, which are referred to as the "platform," is of particular importance.

Utilizing software product line engineering methods and platforms, Toshiba has been developing software for the control and monitoring systems of power generation plants at the first stage of development and has applied it to more than 150 actual projects over the past 30 years. We are also promoting research and development for measurement of structural deterioration of platforms, making it possible to evaluate platform quality.

## 1　まえがき

近年のソフトウェア開発では，開発規模が増大しているだけでなく，仕様が多様化かつ複雑化している。このため，個々のソフトウェア製品をそのつど個別に作成したり，単に過去の類似製品を流用するだけでは，生産性は極めて低くなり，品質も悪化してしまう。

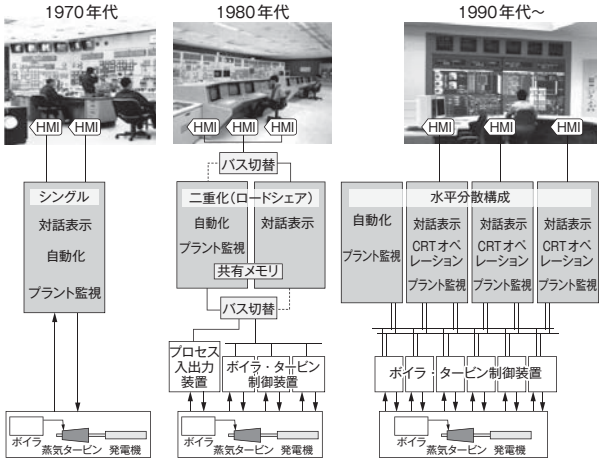
このような問題を解決するには，プロダクトライン型開発[(1)]と呼ばれる方法が効果的である。プロダクトライン型開発とは，製品の共通基盤となるプラットフォームをコア資産と定義し，そのプラットフォームに基づき個々の製品を実現する方法である。プラットフォームをあらかじめ構築しておくことで，個別に製品を開発する場合に比べ，生産性と品質を大きく向上させることができる。

東芝が永きにわたって提供してきた発電所向け監視制御システムでは，当初からプロダクトライン型開発に取り組んでおり，まずその適用例について述べる。次に，当社が研究開発を進めている，プロダクトライン型開発で今後必要となる技術について述べる。

## 2　発電所向け監視制御システムにおける取組み

### 2.1　発電所向け監視制御システム

当社は，1970年代から原子力・火力発電所向け監視制御システムを開発し，これまでに国内外に150以上のシステムを納入してきた。このシステムは，プラント内の各機器や装置と運転に必要な情報をやり取りすることにより，中央制御室での監視及び操作を可能にしている。火力発電所におけるシステム構成の変遷を**図1**に示す。




HMI：Human Machine Interface
CRT：Cathode Ray Tube

**図1. 火力発電所におけるシステム構成の変遷** ― 最新のハードウェアやOSの採用及び分散型システムの導入など，モデルチェンジを行いながら現在に至っている。

Evolution of configuration of control and monitoring systems for thermal power plants

### 2.2　発電所向け監視制御システムのプラットフォーム

発電所向け監視制御システムは，原子力と火力の違いだけでなく，発電技術の進化や発電規模の拡大に合わせ，多様な構成と機能の組合せで実現する必要がある。このようなシステムをそのつど個別に作成すると，生産性や品質が悪くなり，発電所に求められる高い信頼性と長期間の保守継続も実現できなくなってしまう。

この問題を解決するため，発電所向け監視制御システムでは，当初からプロダクトライン型開発に取り組んでいる。プロダクトライン型開発では，“プラットフォームの構築”，“プラットフォームに基づいた製品開発”，及び“プラットフォームと製品の構成管理”が重要であり，それぞれについて以下に述べる。

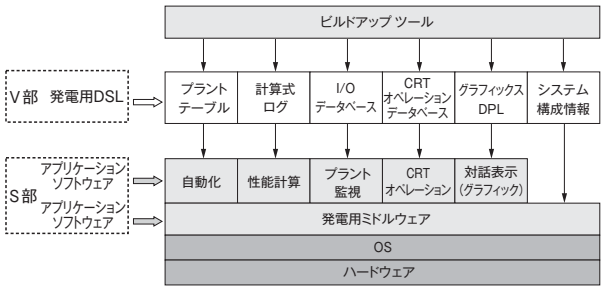
**2.2.1　プラットフォームの構築**　発電所向け監視制御システムのプラットフォームを**図2**に示す。このプラットフォームは，発電所に共通する処理を実行する標準仕様部（S（Standard）部）と，発電所により異なる入出力点や運転手順などを実現する可変仕様部（V（Variable）部）に明確に分離した構成としている。

自動化やプラント監視といった各機能は，テーブル方式の発電用DSL（Domain Specific Language：特定作業向け言語）で表現されるプラントテーブルやI/O（入出力項目）データベースなどのV部により，各発電所に合わせた仕様で定義され，これをS部のアプリケーションソフトウェアがインタプリタ方式(注1)などで処理することにより実行される。

発電用ミドルウェアは，コンピュータのOS（基本ソフトウェア）と各機能のアプリケーションソフトウェアの間に位置し，OSやコンピュータのハードウェアに依存しないアプリケーションソフトウェアの開発を可能にしている。

V部の発電用DSLは，ビルドアップツールにより，プログラミング知識がないプラントの専門家でも作成することができる。

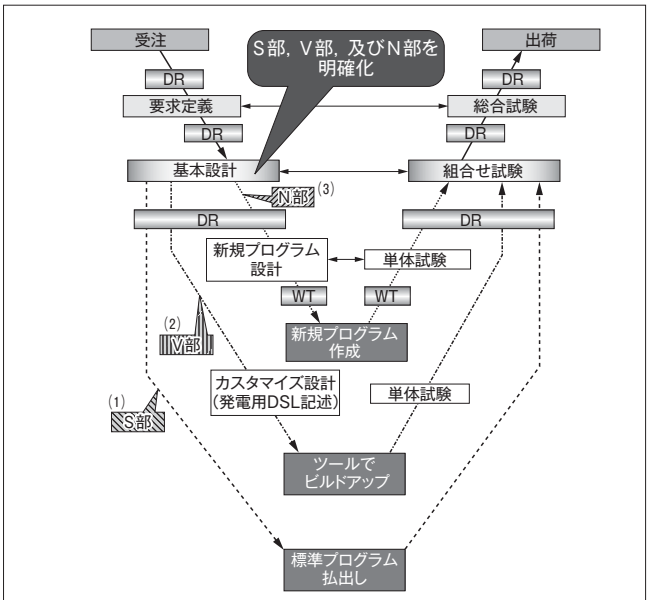
**2.2.2　プラットフォームに基づいた製品開発**　プラットフォームを実際の製品開発に適用する際の開発工程を，発電システムV字モデルとして確立した（**図3**）。製品開発では，SとVだけではなく，新規仕様部（N（New）部）として新たなソフトウェアの開発が必要になることがある。




DPL：Display Point List（画面表示項目）

**図2．発電所向け監視制御システムのプラットフォーム** ― 標準化されたプラットフォームに基づき，アプリケーションソフトウェアの組合せとDSLによるテーブル作成で構築される。

Platform of control and monitoring system for power plant




DR：Design Review
（技術，設計，及び品質部門が合同でレビュー）

WT：Walk Through
（製作者と設計部門の有識者でレビュー）

**図3．ソフトウェアの開発工程** ― S部，V部，及びN部を明確に区別した発電システムV字モデルで，ソフトウェアの開発を実施している。

Software development process using power generation system V model

発電システムV字モデルにおいては，基本設計段階でS部，V部，及びN部を明確にして開発を行う。S部であれば標準プログラムを払い出し（図3の(1)），V部であればビルドアップツールにより発電用DSLテーブルの作成及び試験を行い（図3の(2)），N部であれば新たに設計，製作，及び試験を行う（図3の(3)）。

**2.2.3　プラットフォームと製品の構成管理**　発電所向け監視制御システムのソフトウェアは，再利用と変更時の確実な水平展開を可能にするため，ソフトウェア倉庫と呼ばれるソフトウェア支援システムで一元管理している。N部についても，プラットフォームに適合した開発を行うことにより，ソフトウェア倉庫で構成管理を行い，次の製品への払出しを可能にしている。

## 3　プロダクトライン型開発に今後必要な技術

プロダクトライン型開発で今後必要となる技術について研究開発を行っている。コア資産であるプラットフォームは，ソフトウェアの多くの製品展開に耐えられるように構築することが前

(注1)　プログラミング言語で書かれたプログラムを，コンピュータが実行できる形式に逐次変換しながら実行するソフトウェアの形式。

提である。具体的には，顧客によって異なる仕様に対して柔軟に対応でき，多様化するハードウェアやOSなどソフトウェアの土台からの影響を局所化できるようにする。しかし，構築時点では対象とする業務領域全体の分析（ドメイン分析）が十分洗練されたものであったとしても，その後の予想外の顧客ニーズや市場状況の変化に伴い，プラットフォームに対する要求が変化することがある。その結果，プラットフォームを修正する可能性がある。

このようなプラットフォームを保守する過程では，プラットフォームが構造的に劣化するリスクを伴う。そこで，このような構造的な劣化を診断する技術の研究開発を行っている。この技術により最終的には，プラットフォームの品質を維持していくための費用対効果を見積りできるようになると期待される。

以下に，構造的な劣化を表す特徴的な状態を整理し，構造的な劣化を計測するためのメトリクス（指標）について述べる。

### 3.1　構造的な劣化を表す状態

プラットフォームが構造的に劣化していない適切な状態から，仕様の変更や追加によりプラットフォームを修正し保守する過程で構造的な劣化を引き起こすようすを**図4**に示す。

プラットフォームの構造は，ソフトウェアの構成要素である個々のモジュールと，それらモジュール間の依存関係で表現できる。依存関係は，単純にどのモジュールが関連しているのかという“結び付き”と，利用する側のモジュールから利用される側のモジュールへの“作用方向”で表現できる。プラットフォームに対して仕様の変更や追加が発生すると，モジュール自体やモジュール間の“結び付き”及び“作用方向”に影響を与える。以下に，個々のモジュール機能の追加や変更が行われる過程で与える影響について述べる。

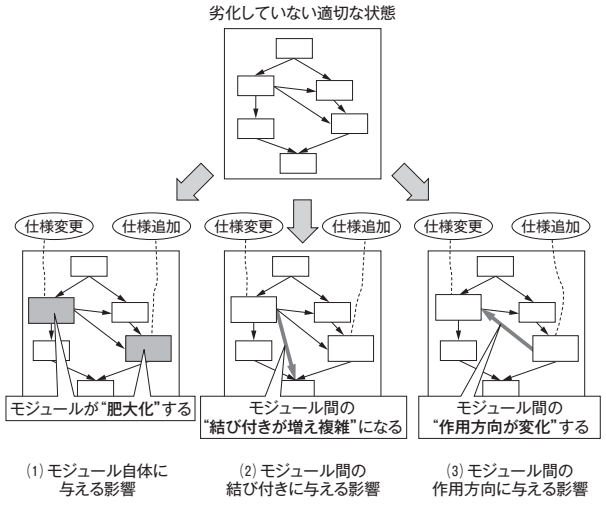



**図4．構造的な劣化を表す状態** ― プラットフォームは，仕様の変更や追加により修正し保守する過程で，構造的に適切な状態から劣化を引き起こす。
State of structural deterioration of software platform

(1)　モジュール自体に与える影響　　本来は必要のない役割をモジュールに与えることで，そのモジュールに関連のない機能や情報を作り込む可能性がある。その結果，モジュール自体が肥大化する。

(2)　モジュール間の結び付きに与える影響　　モジュール内に閉じた修正だけでなく，それまで関連のなかったモジュールと結び付く可能性がある。その結果，モジュール間の結び付きが増え複雑になる。

(3)　モジュール間の作用方向に与える影響　　モジュール内に閉じた修正だけでなく，本来想定していなかった方向の関連がモジュール間で発生する可能性がある。その結果，モジュール間の作用方向が意図しない状態に変化する。

### 3.2　構造的な劣化を把握するためのメトリクス

前節で示したプラットフォームの三つの構造的な劣化状態を把握するため，メトリクス（凝集度，結合度，安定度）を定義する。

プラットフォームの劣化に対して，モジュール間の結び付きを，あらかじめ固定された関連（静的な関連と呼ぶ）だけでなく，条件により変更される関連（動的な関連と呼ぶ）でもとらえることが重要である。例えば，C言語のプログラムでは，単純な関数の呼出しによる静的な関連だけでなく，呼び出す関数を途中で変更できる関数ポインタにより，動的な関連を扱うことができる。したがって，静的な関連だけでなく動的な関連も計測対象とする。以下に，個々のメトリクスについて述べる。

**3.2.1　凝集度**　　モジュール自体の肥大化は，モジュール内で無関係な機能や情報が含まれているかどうかで判定する。そこで，モジュール内の機能と情報との関連の多さを表す指標である“凝集度”を活用する。**図5**(a)の左部に示すように，モジュールに無関係な機能や情報が存在する状態は，凝集度が低い（劣化している）ととらえる。

既存の計測方法にはLCOM（Lack of Cohesion in Methods）[(2)(注2)]やLCOM*[(2)(注3)]などがあり，主に静的な関連を扱うものである。プラットフォームの計測では，モジュール内の動的な関連も考慮する。そのうえで，モジュール内のそれぞれの機能について，モジュール内の情報に対する参照回数に着目し計測する。

**3.2.2　結合度**　　モジュール間の結び付きの多さは，どのような参照が行われているかで判断する。そこで，モジュール間の機能や情報の参照の多さを表す指標である“結合度”を活用する。図5(b)の右部に示すように，関連するモジュールに対する参照の数が多い状態は，結合度が高い（劣化している）ととらえる。

(注2)　モジュールが保有する機能との静的な関連に基づき，凝集度（Cohesion）の欠落度合を計測するための手法。
(注3)　“LCOM”を改良した手法。

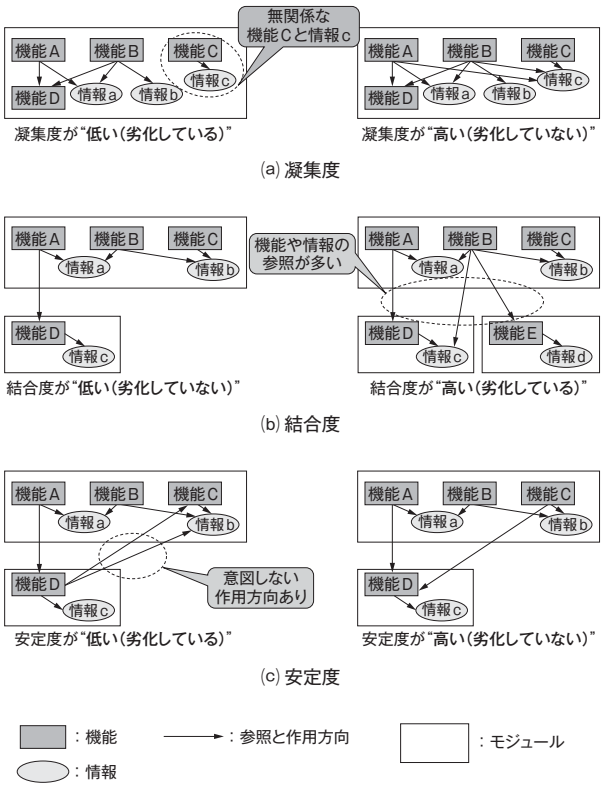



**図5. 構造的な劣化を計測するためのメトリクス** ― 構造劣化を表す三つの状態を定量的に計測するために，凝集度，結合度，及び安定度というメトリクスを定義する。

Software metrics to measure structural deterioration

既存の計測方法にはCBO (Coupling between Objects)[(2)(注4)]やMPC (Message Passing Coupling)[(2)(注5)]などがあり，主に静的な関連を扱うものである。プラットフォームの計測では，静的な関連か動的な関連かで結び付きの多さは異なるため，これらの違いで重み付けを行う。そのうえで，関連するモジュールの機能や情報への参照回数に着目し計測する。

**3.2.3 安定度**　モジュール間の作用方向の変化は，事前に定義したモジュール間の作用方向のルールを逸脱するものがあるかで判断する。ここでは，意図した作用方向である状態を“安定度が高い”と呼び，作用方向の安定を表す指標として，“安定度”を新たに定義する。図5(c)の左部に示すように，関連するモジュールに対して想定していない作用方向が存在する状態は，安定度が低い（劣化している）ととらえる。

プラットフォームの計測では，動的な関連に伴う作用方向を考慮する。そのうえで，関連するモジュールの機能や情報への想定していない参照の回数に着目し計測する。

(注4)　モジュールの静的な関連に基づき，結合度 (Coupling) を計測するための手法。

(注5)　モジュールの呼び出し (Message Passing) による静的な関連に基づき，結合度 (Coupling) を計測するための手法。

## 4　あとがき

ここでは，プロダクトライン型開発においてコア資産として共通に活用されるプラットフォームに焦点を当て，実システムでの取組みと今後必要となる技術について述べた。

適用例として述べた発電所向け監視制御システムでは，プラットフォームの実現形態は時代とともに進化させてきたが，基本的なコンセプトは開発当初から30年以上にわたって踏襲し，原子力ではPODIA$_{TM}$及びA-PODIA$_{TM}$，火力ではCOPOS$_{TM}$及びTOSMAP-DS$_{TM}$という品質の高いシステムソフトウェア製品として工業化した。こうした取組みは高く評価され，米国カーネギーメロン大学ソフトウェア工学研究所の主催で2008年9月に開催されたプロダクトラインの国際学会“ソフトウェアプロダクトライン国際会議 (SPLC) 2008”で，日本企業として初めて“プロダクトラインの殿堂 (Product Line Hall of Fame)”に登録された。

また，プロダクトライン型開発で今後必要となる技術として，プラットフォームの構造的な劣化を診断するメトリクスと，その具体的な計測方法を定義し，妥当性を実験で実証した。現在，実プロジェクトに適用して評価中である。

最終的には，プラットフォームの品質を維持していくための費用対効果を見積もり，コア資産として合理的に評価するための仕組みの実現を目指す。当社は，プロダクトライン型の開発を行うに適したシステムが多く，今後もこの方法による開発を推進していく。

## 文　献

(1) Clements, P.; Northrop, L. (前田卓雄 訳). ソフトウェアプロダクトライン. 東京，日刊工業新聞社，2003，652p.
(2) Henderson-Sellers B. Object-Oriented Metrics: Measures of Complexity. New Jersey, U.S.A., Prentice-Hall, 1995, 234p.

滝沢　　治　TAKIZAWA Osamu
電力システム社 府中事業所 原子力プロセス監視制御システム部主務。原子力施設向け監視システムの設計に従事。日本原子力学会会員。
Fuchu Complex

赤石　富士雄　AKAISHI Fujio
電力システム社 府中事業所 発電情報制御システム部主務。火力発電所向け監視制御システムの設計に従事。
Fuchu Complex

早瀬　健夫　HAYASE Takeo, D. Eng.
ソフトウェア技術センター ソフトウェア設計技術開発担当参事，工博。ソフトウェアプロダクトラインの技術開発に従事。電気学会，情報処理学会，ACM，IEEE会員。
Software Design Technology Group