# しなやかなソフトウェアシステムを目指して

本位田　真一[†1,†2]

# Towards Flexible Software Systems

Shinichi Honiden[†1,†2]

## 1. はじめに

近年，ソフトウェアが活躍する環境が広がるとともに，ソフトウェアシステムは複雑化の一途をたどり，システムとして,「想定外」の事象や環境の変化，状況の変化に「しなやかに」対応することが求められている．この「しなやかさ」を実現するために，システム制御系，人工知能系などの分野では，以前よりさまざまな研究が行われてきた．ソフトウェア工学の世界でも，古くて新しいテーマとして，動的進化 (Dynamic Evolution)，動的保守 (Dynamic Maintenance) や自己適応システム (Self-adaptive Systems) といった研究テーマで，さまざまな研究アプローチが提案されている．本講演では，それらの研究アプローチを概観し，その一つとして我々の研究内容を紹介する．

## 2. 実現のためのアプローチ

しなやかなソフトウェアを実現するには，従来のソフトウェア開発手法や実行手段，管理方法，更新手段に対して，動的なアプローチを追加する必要がある．そのような取り組みの一つとして，動的進化，動的保守や，自己適応システム[1)] に関する研究をあげることができる．動的進化や動的保守に関しては，従来のソフトウェア進化やソフトウェア保守をシステムを停止することなく実現する試みであり，進化や保守の対象やタイミングはソフトウェア開発者に委ねられる一方で，大規模な変更を扱うことが可能である．対して，自己適応システムは，変化に対して開発者は介在せず，システム自身がみずからの目的と与えられた制約を管理し，構成や振舞いを自発的に変化させることのできるシステムを指す．

例えば，自己適応システムを実現するためには，従来のソフトウェア開発と比較して，1）システムが要求を動的に管理するための記述方法やメカニズム，2）適応を考慮した汎用的なモデリング，3）振る舞いを変化させるための意思決定メカニズム，4）適応後の振る舞いの保証手段などが必要となる．1）の要求管理については不確かさや不完全性の記述手段が，4）の振舞いの保証については不確かさに対する動的な検証方法の実現が主な研究テーマとなっているが，2）のモデリングと 3）の意思決定メカニズムに関しては，情報の収集（Collect），分析（Analyze），意思決定（Decide），実行（Act）の 4 つのフェーズにより構成される Control loop と呼ばれる制御プロセスが着目されている[⋆1]．

我々は，しなやかなソフトウェアの実現を目指して，Control loop を分割配備するというアプローチから，要求記述から自己適応システムが必要とするコンポーネントとその接続を決定するアーキテクチャレベルのコンパイラ[2)] や実装フレームワーク を提案してきた．システムがしなやかに動作するためには，変化する環境と変化する要求に柔軟に対応しなければならない．要求記述とシステム構成，システム実装コードの疎で密なる連携が求められるところである．

## 参考文献

1) Betty H.C. Cheng, Rogério de Lemos, Holger Giese, Paola Inverardi, Jeff Magee, and et al. Software engineering for self-adaptive systems: A research road map. In *Dagstuhl Seminar Proceedings 08031*, 2008.

2) Hiroyuki Nakagawa, Akihiko Ohsuga, and Shinichi Honiden. gocc: A configuration compiler for self-adaptive systems using goal-oriented requirements description. In *Proc. of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS '11)*, 2011.

†1 国立情報学研究所
National Institute of Infomatics
†2 東京大学
The University of Tokyo

⋆1 自己適応システムの研究グループでは，MAPE (Monitor, Analyze, Plan, Execute) loop と呼ばれる場合も多い．