Model-based Self-Adaptive Components: A Preliminary Approach

Pedro Rodrigues, Emil Lupu

Department of Computing
Imperial College London

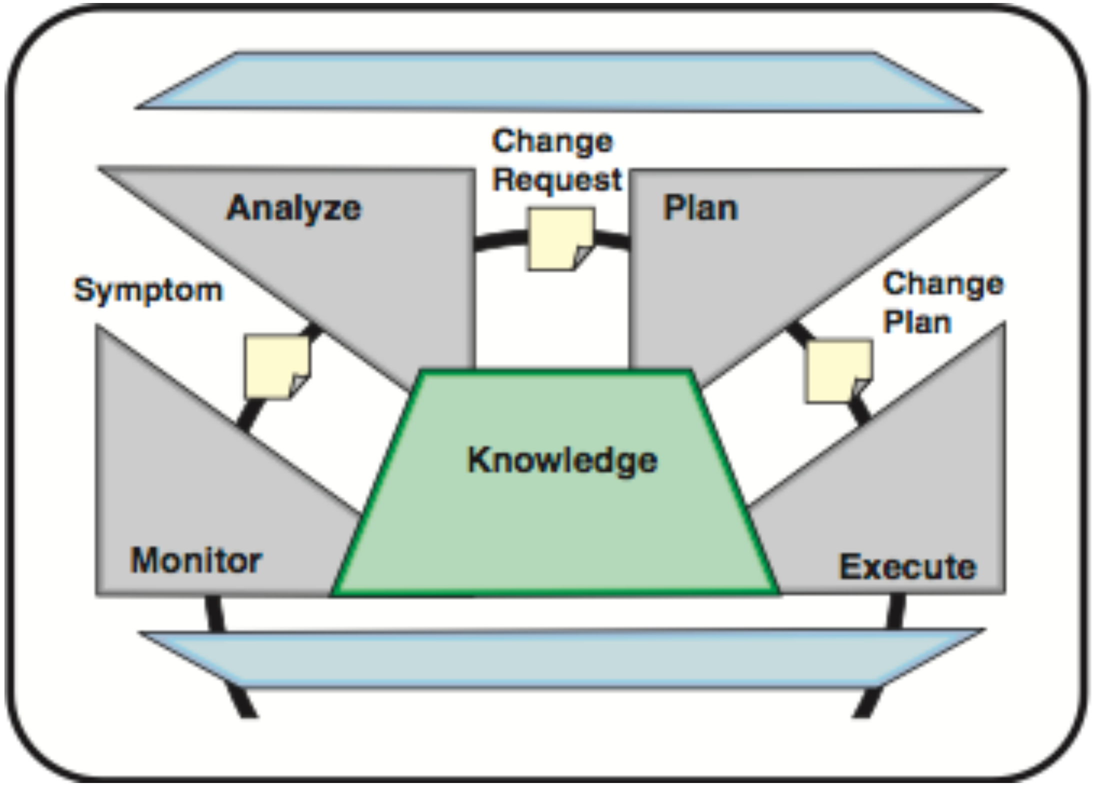
Motivation

- Modern software systems are growing in terms of:
  - scale
  - complexity
  - dynamicity
  - heterogeneity
- Only Human management
  - deficient dependability level

- Self-managing systems
  - effective approach

- Model-based adaptation
  - improves reliability
  - enhances trust

Background


Change
Request
Analyze
Plan
Symptom
Change
Plan
Knowledge
Monitor
Execute

Background

- Monitoring
  - » Main metrics
  - » Fix set of metrics to monitor
    - ▪ Detects 66% of faults with 30% of metrics
  - » Dynamic set of metrics to monitor
    - ▪ Augment when a fault occurs
  - » Reduce power consumption
  - » Control monitor impact

Background

- Decision-Making
  - Event-Condition-Actions(ECA) Policies
  - Goal Policies
  - Utility-Function Policies

  - Stitch Language
    - Modified ECA Policies
    - Tactics
      - Condition
      - Actions
        - Expected behaviour
    - Strategies
      - General conditions
      - Tree of Condition-Tactics
        - Estimated time
        - Next steps (in case of failure)

Background

- Decision-Making
  - Genetic algorithms
    - Utility-Functions
  - Planning algorithms
    - Unpredicted states
- Component-Models
  - Darwin
    - Structural view
    - Modes: behavioural extension
  - FRACTAL
    - Interfaces to reconfigure internal details
      - Structure
      - Behaviour
  - MOCAS
    - Behavioural view
      - State machines
      - Signal transitions

Background

- Self-adaptation frameworks
  - Rainbow
    - Runtime architectural model: Acme ADL
      - Components:
        - Functional and non-functional annotation
        - Expected interactions
      - Architectural constraints
    - Stitch language
  - GRAF
    - Runtime architectural model
    - Model-based Adaptation

Background - Summary

Problems facing current approaches:

- Structural adaptation
- Centralised model
- Centralised decision-making
- Adaptation costs
- Behavioural adaptation

Proposed Approach

- Overview:
  - Components with self-managing capabilities
  - Hierarchical structure
    - Handle adaptation concerns at different levels
      - Low-level: configuration parameters
      - High-level: replace components
    - Composite components manage their sub-hierarchy
    - Components specify management level for parent component

Proposed Approach

- Runtime Model
  - Structural view
    - Components: required and provided interfaces
    - Non-functional annotations
  - Behavioural model
    - Interactions between interfaces
    - Internal operations
  - Incremental interfaces
    - Behavioural evolution
  - Relevant Metrics
    - Dependencies
    - Discretisation map

Proposed Approach

- Adaptive Monitoring
  - Type of value propagation
    - Period-based
    - Interval-based
  - Set of monitoring metrics
    - ECA policies to change set
  - Updates annotations

Proposed Approach

- Decision-Making
  - » Runtime model
  - » Policy Language
    - ▪ Stitch based
    - ▪ Expected outcome
    - ▪ Cost estimation
  - » Utility-function for conflicting policies
  - » Planning for unpredicted states
    - ▪ Coordination among components
    - ▪ Decompose goals

Final Remarks

- Hierarchical structure
  - » Handle management at different levels
  - » Ease management specification
- Online reasoning
  - » Increases trust
  - » Improves dependability
- Self-managing systems
  - » Reduce management costs
  - » Delegation of system-administrators

Thank you