Using Quantitative Analysis to Implement Autonomic IT Systems

Radu Calinescu and Marta Kwiatkowska
Computing Laboratory, University of Oxford

UNIVERSITY OF OXFORD

ICSE 2009

Outline

Motivation, background

Development of autonomic IT systems

Case studies

Effectiveness, limitations, applications

Conclusions and future work

Motivation

Key challenges for today's software systems

- demanding non-functional requirements
  – performance, dependability, utility, . . .
- need to adapt to changing scenarios
  – workload, environment, objectives, . . .

Motivation

Key challenges for today's software systems

- demanding non-functional requirements
  – performance, dependability, utility, . . .
- need to adapt to changing scenarios
  – workload, environment, objectives, . . .

can be analysed formally
(for a fixed scenario)
using *quantitative verification*

Motivation

Key challenges for today's software systems

- demanding non-functional requirements
  – performance, dependability, utility, ...
- need to adapt to changing scenarios
  – workload, environment, objectives, ...

can be addressed through adding self-* capabilities in *autonomic IT systems*

can be analysed formally (for a fixed scenario) using *quantitative verification*

Approach

Integrate quantitative verification and autonomic systems

- using formal, quantitative runtime analysis to support multi-objective adaptation
- using new method to implement autonomic IT systems

can be addressed through adding self-* capabilities in *autonomic IT systems*

can be analysed formally (for a fixed scenario) using *quantitative verification*

UNIVERSITY OF OXFORD

Advantages over existing approaches

Integrate quantitative verification and autonomic systems

- using formal, quantitative runtime analysis to support multi-objective adaptation
  - adaptation decisions based on exhaustive analysis of nonlinear behaviour instead of heuristics
  - verified quantitative properties derived from high-level, user-specified system objectives
- using new method to implement autonomic IT systems

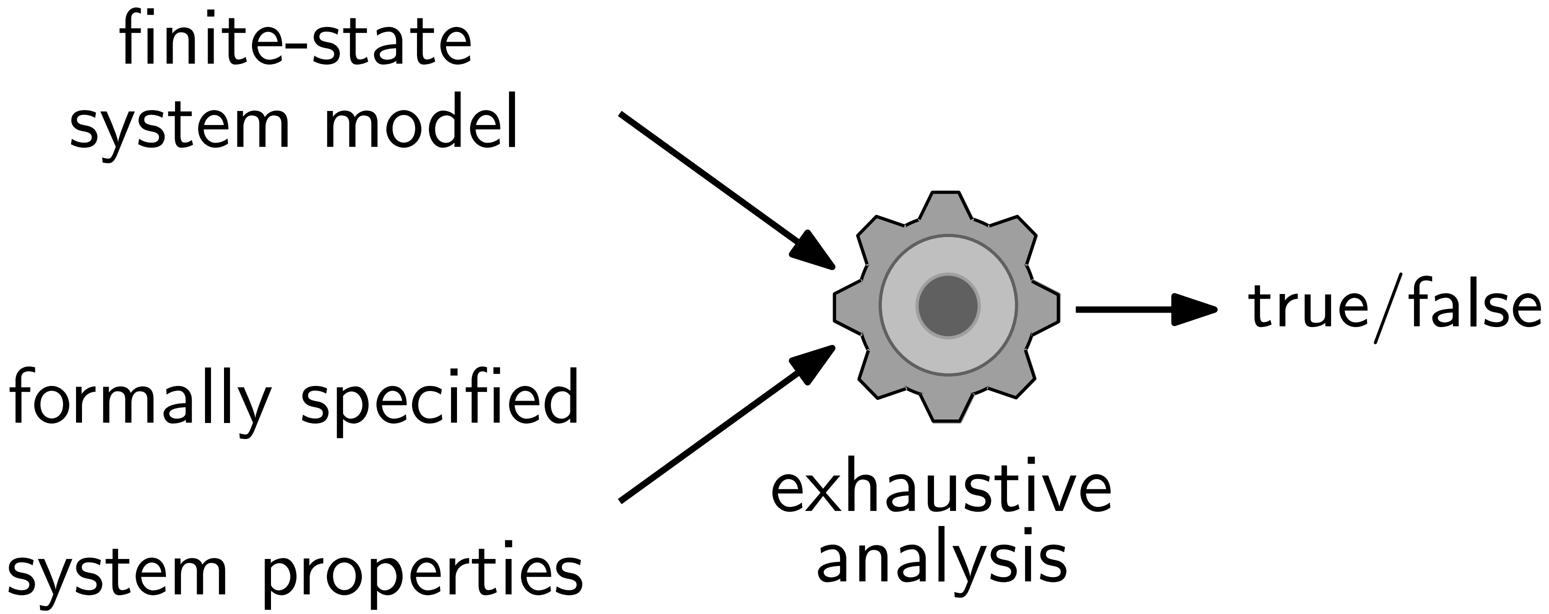
Advantages over existing approaches

Integrate quantitative verification and autonomic systems

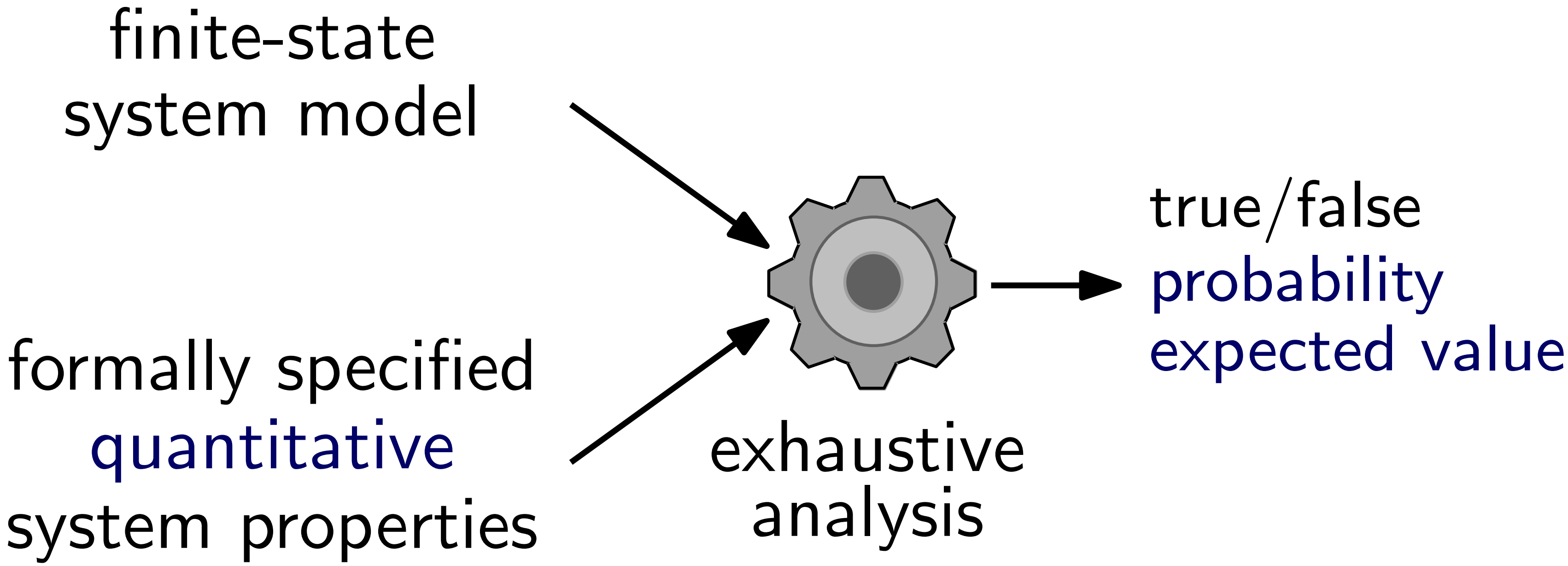
- using formal, quantitative runtime analysis to support multi-objective adaptation
  - adaptation decisions based on exhaustive analysis of nonlinear behaviour instead of heuristics
  - verified quantitative properties derived from high-level, user-specified system objectives
- using new method to implement autonomic IT systems
  - reduced development time/effort through partial automation

[R Calinescu, M Kwiatkowska – CADS*: Computer-Aided Development of Self-* Systems, FASE/ETAPS'09]

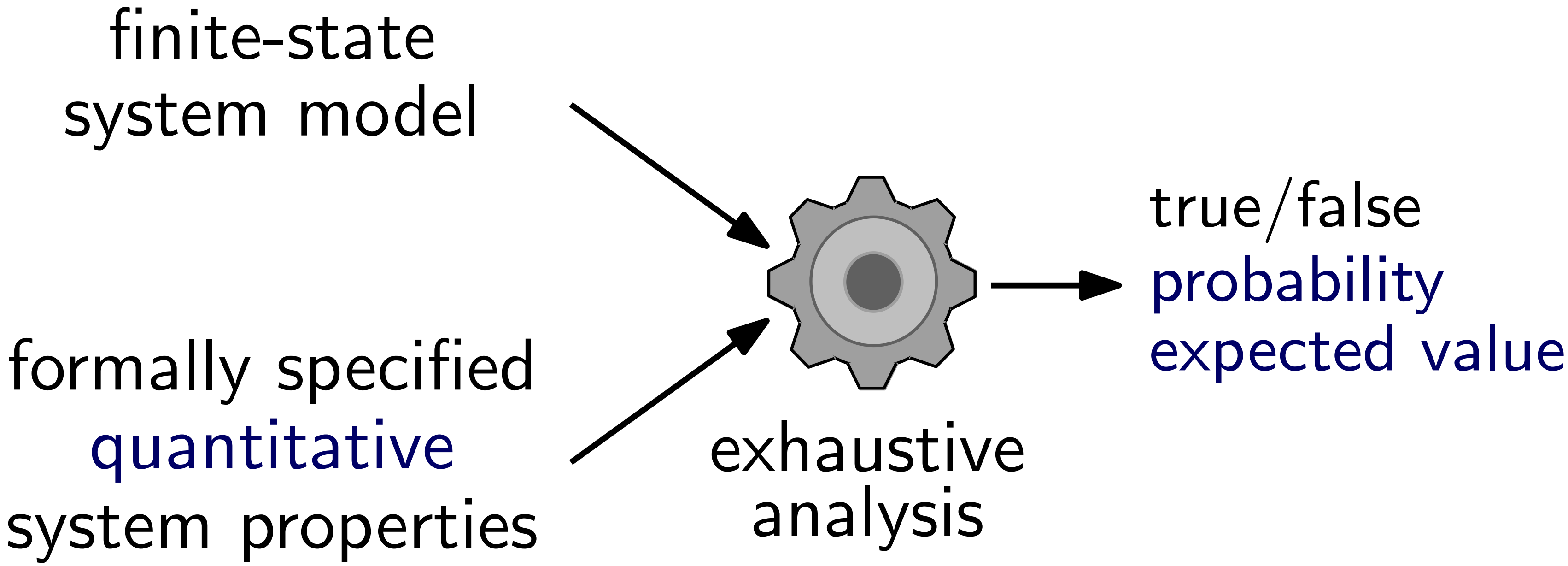
Background: formal verification



Background: quantitative verification



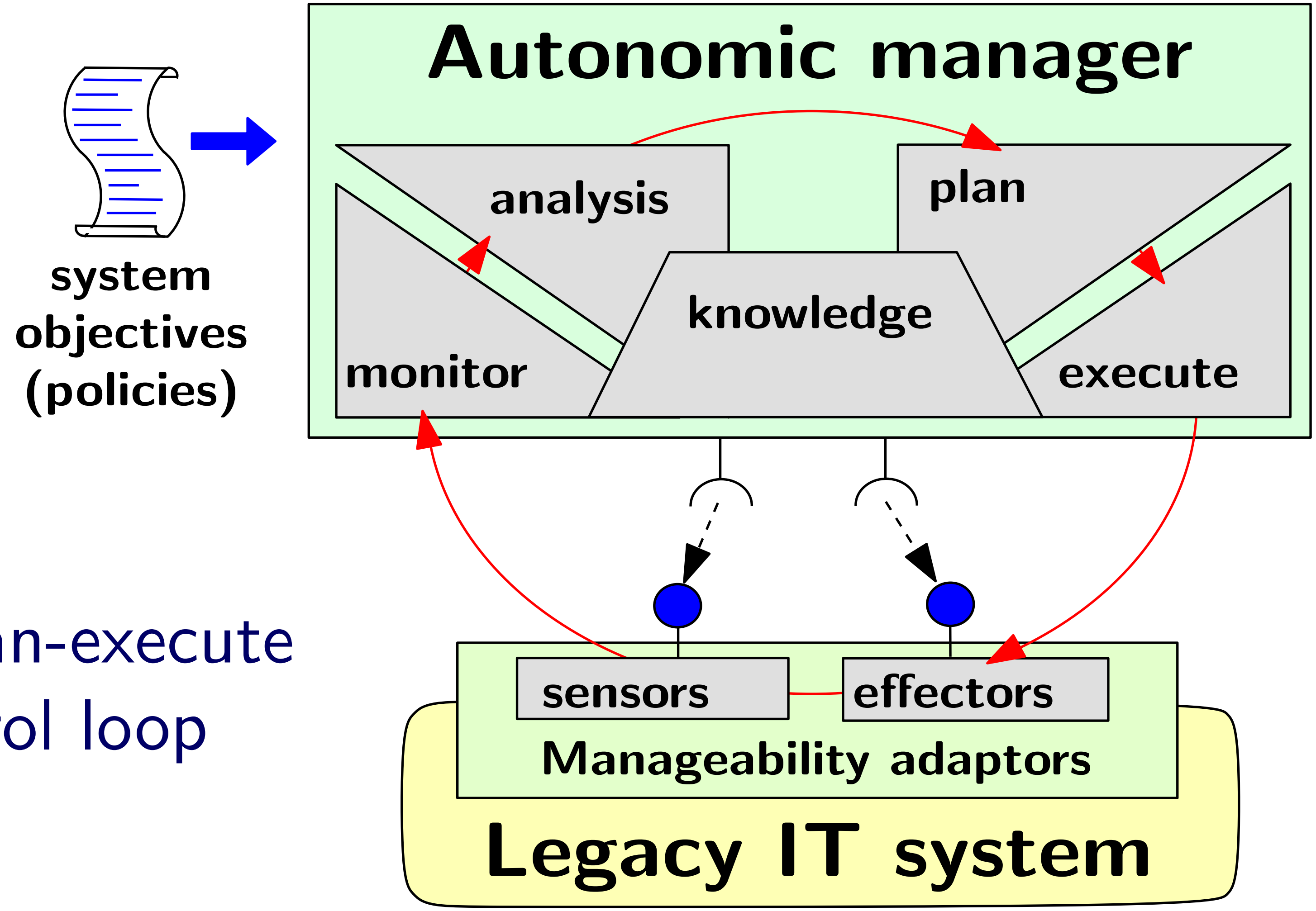
Background: quantitative verification



Supported by probabilistic model checkers

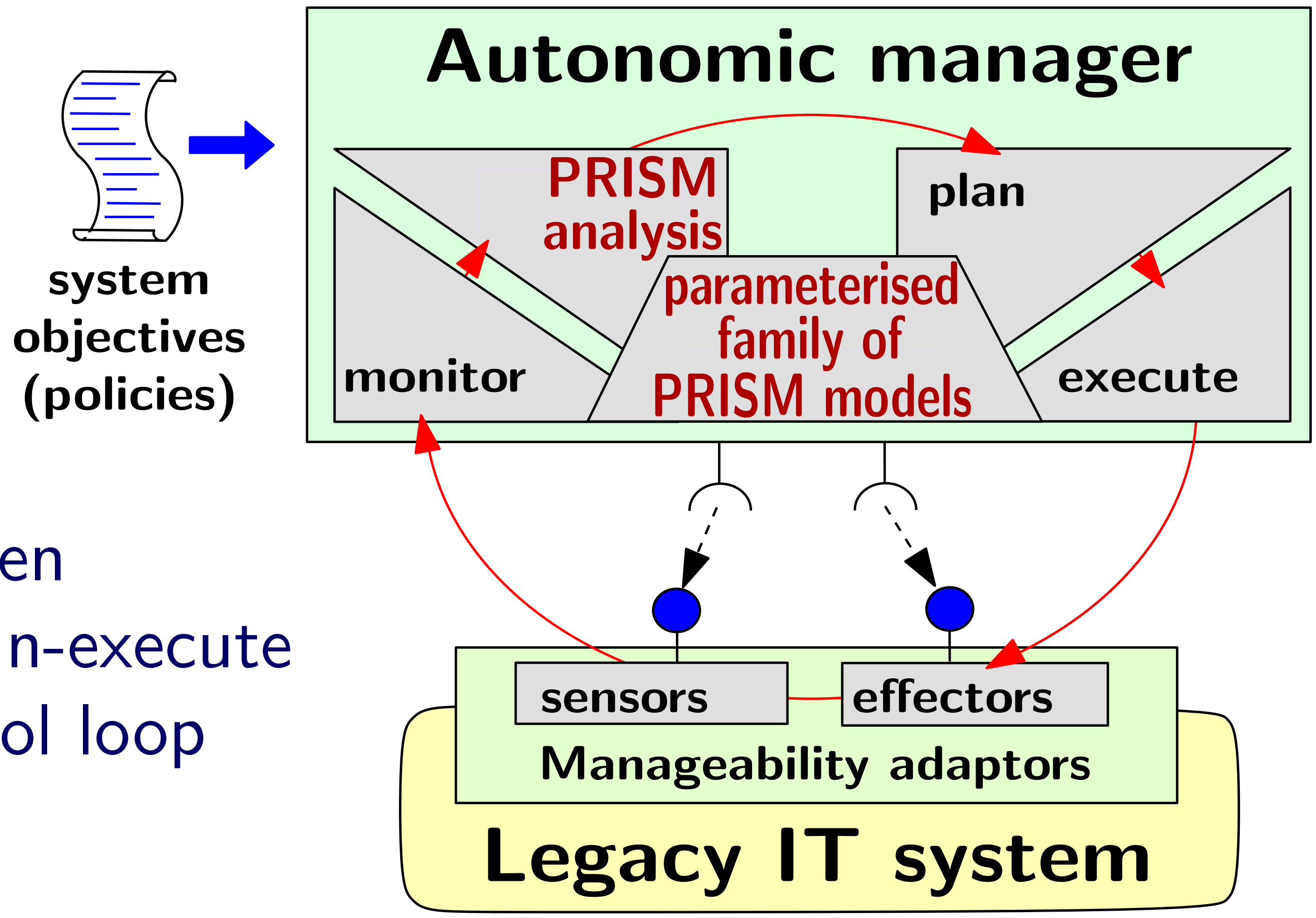
- PRISM (Univ. of Birmingham and Oxford, 2001–)
  - discrete-/continuous-time Markov chains, Markov decision processes
  - probabilistic temporal logics with costs/rewards & expectations
  - multiple verification of parameterised sets of models (*experiments*)
  - applied to many case studies across application domains

Background: autonomic IT systems



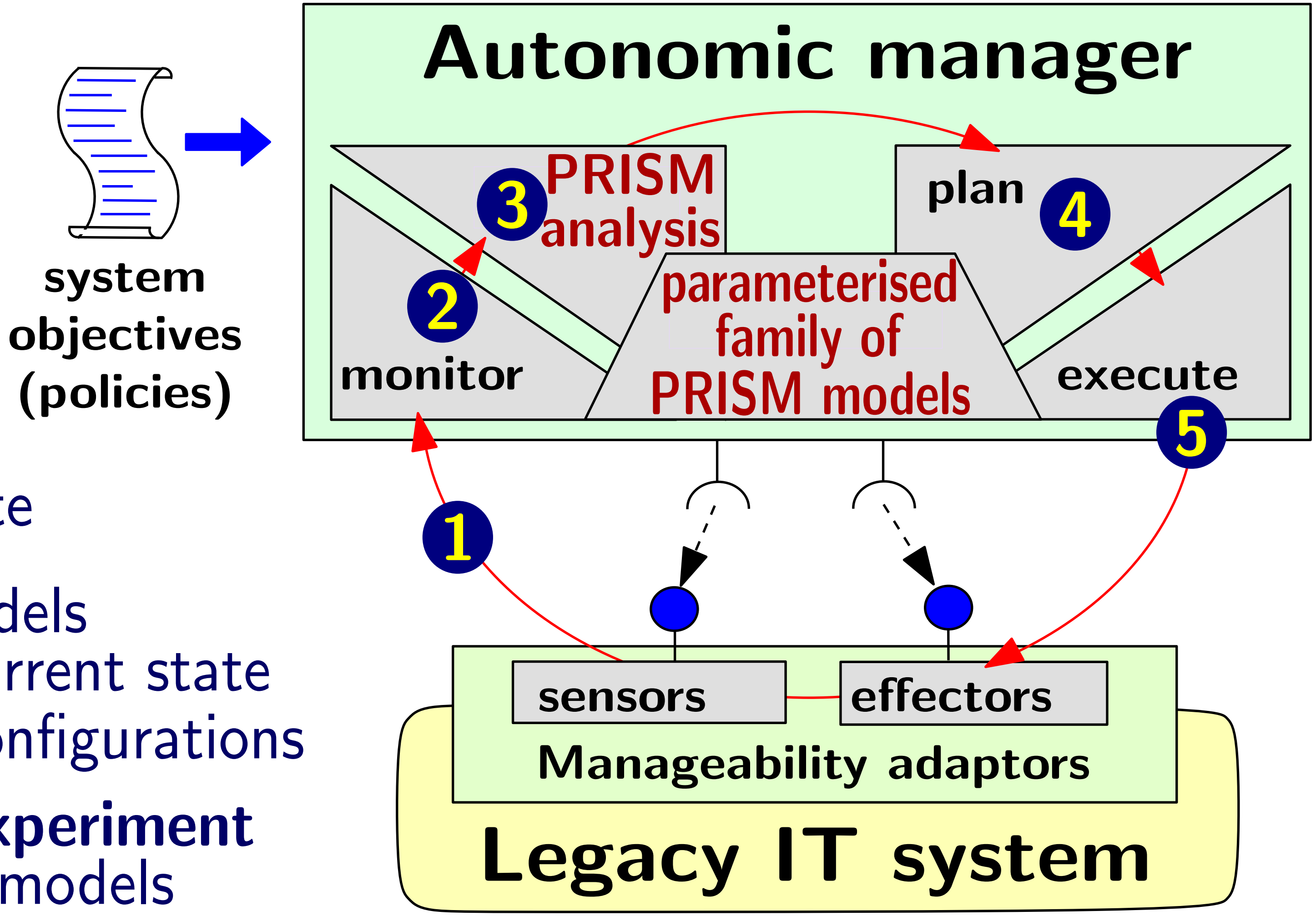
monitor-analyse-plan-execute
autonomic control loop

Integration



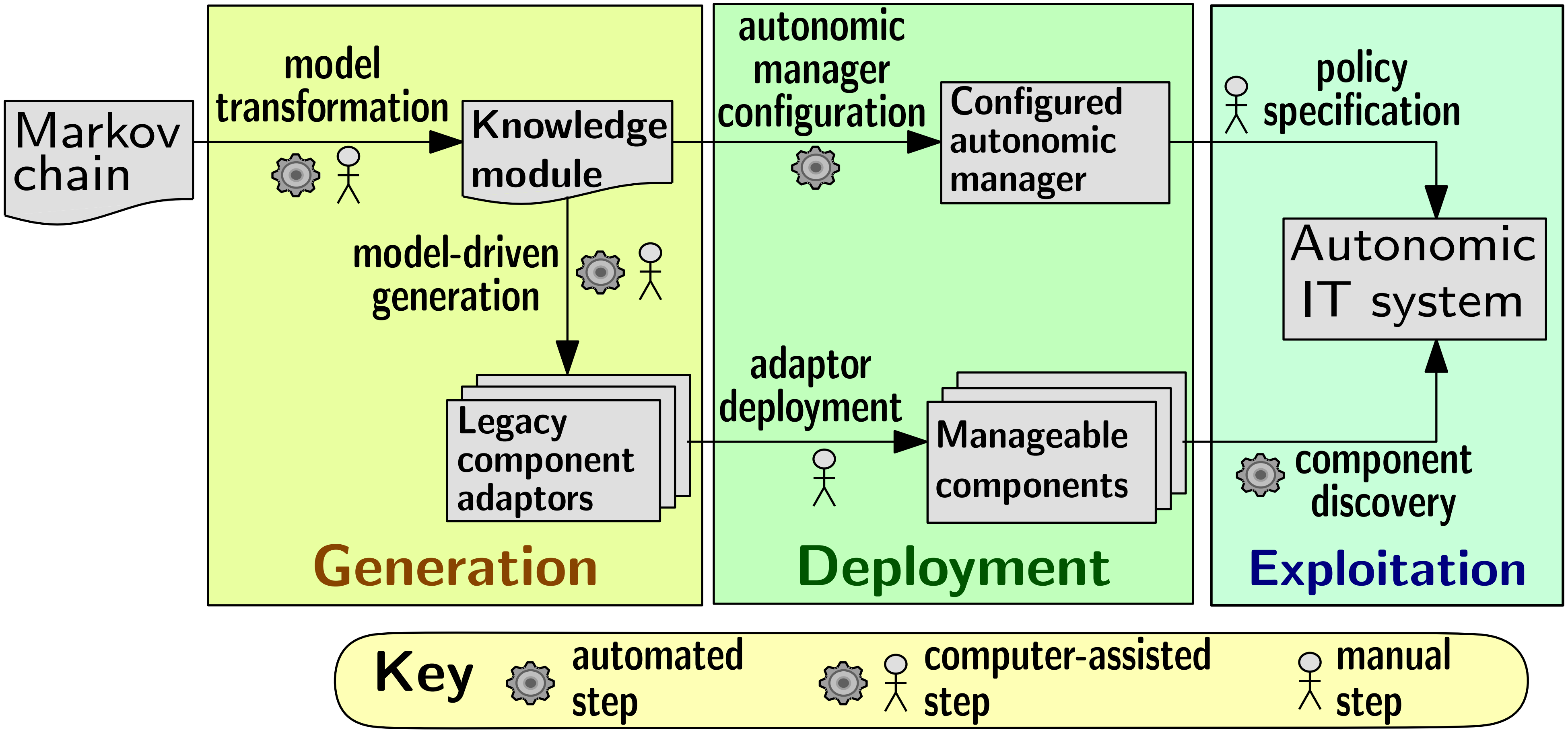
PRISM-driven
monitor-analyse-plan-execute
autonomic control loop

Integration

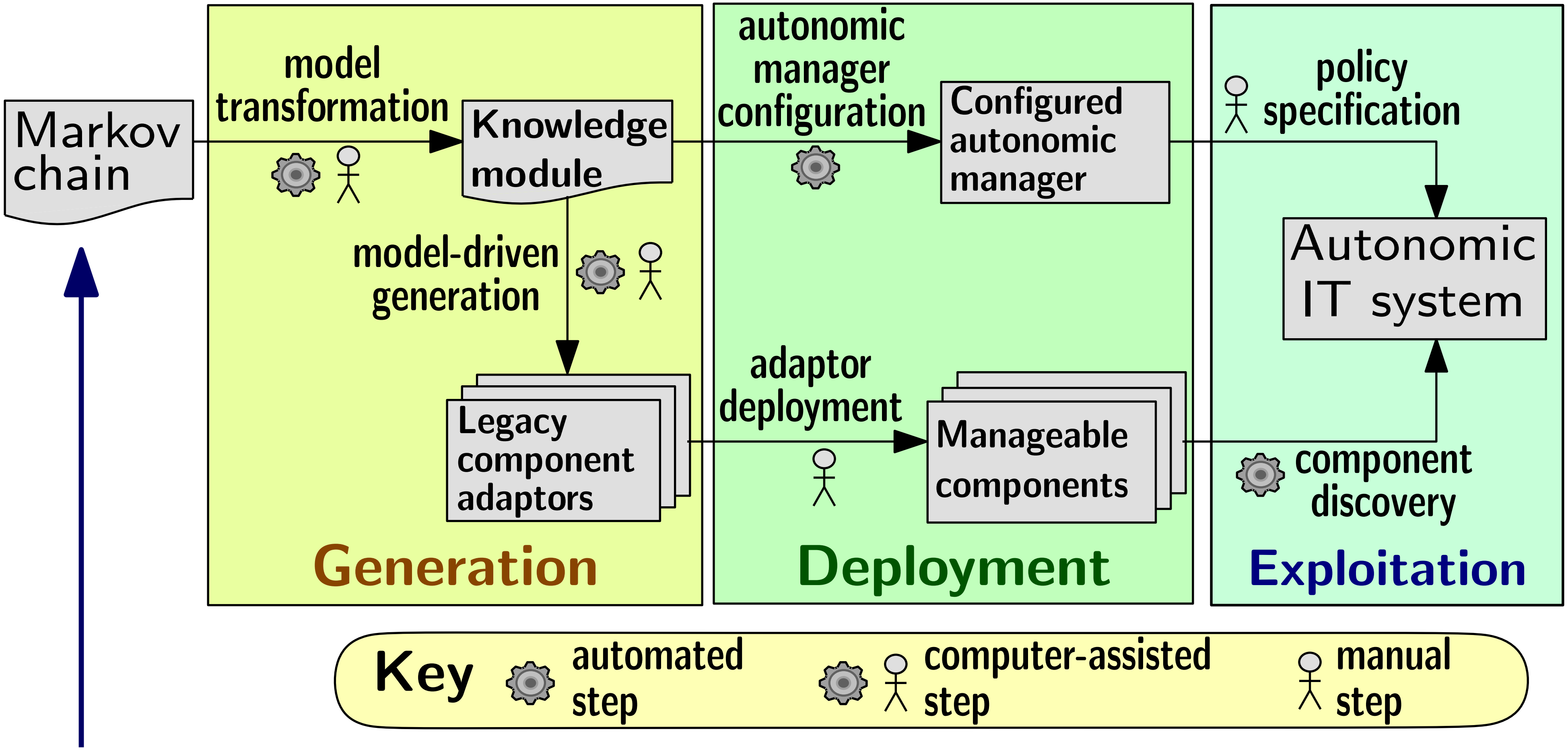


1. monitor system state
2. select subset of models corresponding to current state & set of possible configurations
3. carry out PRISM **experiment** to analyse selected models
4. map analysis results to system objectives & decide "best" configuration
5. enforce chosen configuration

Development method



Development method: generation

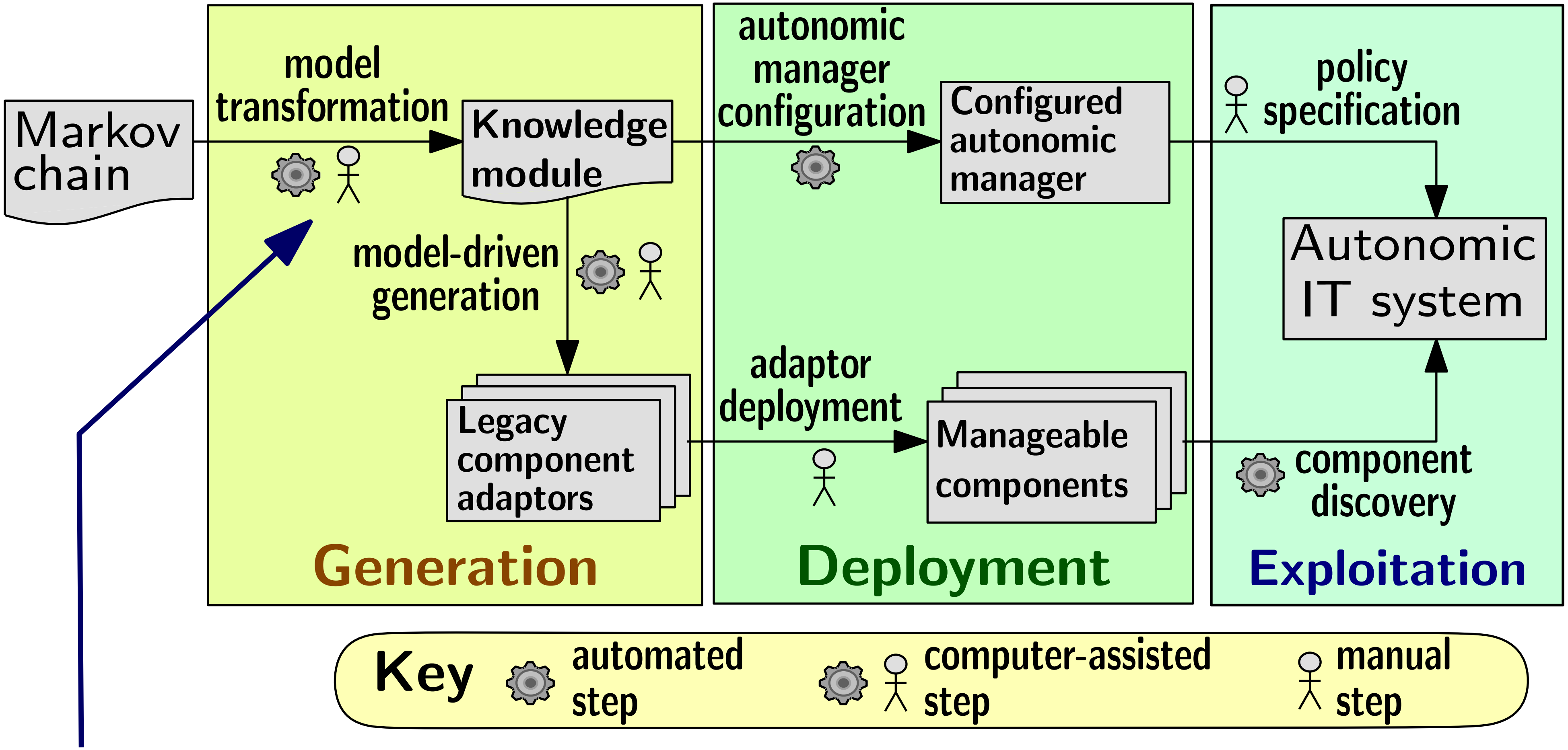


PRISM discrete-/continuous-time Markov chain

– available from the formal verification of the system

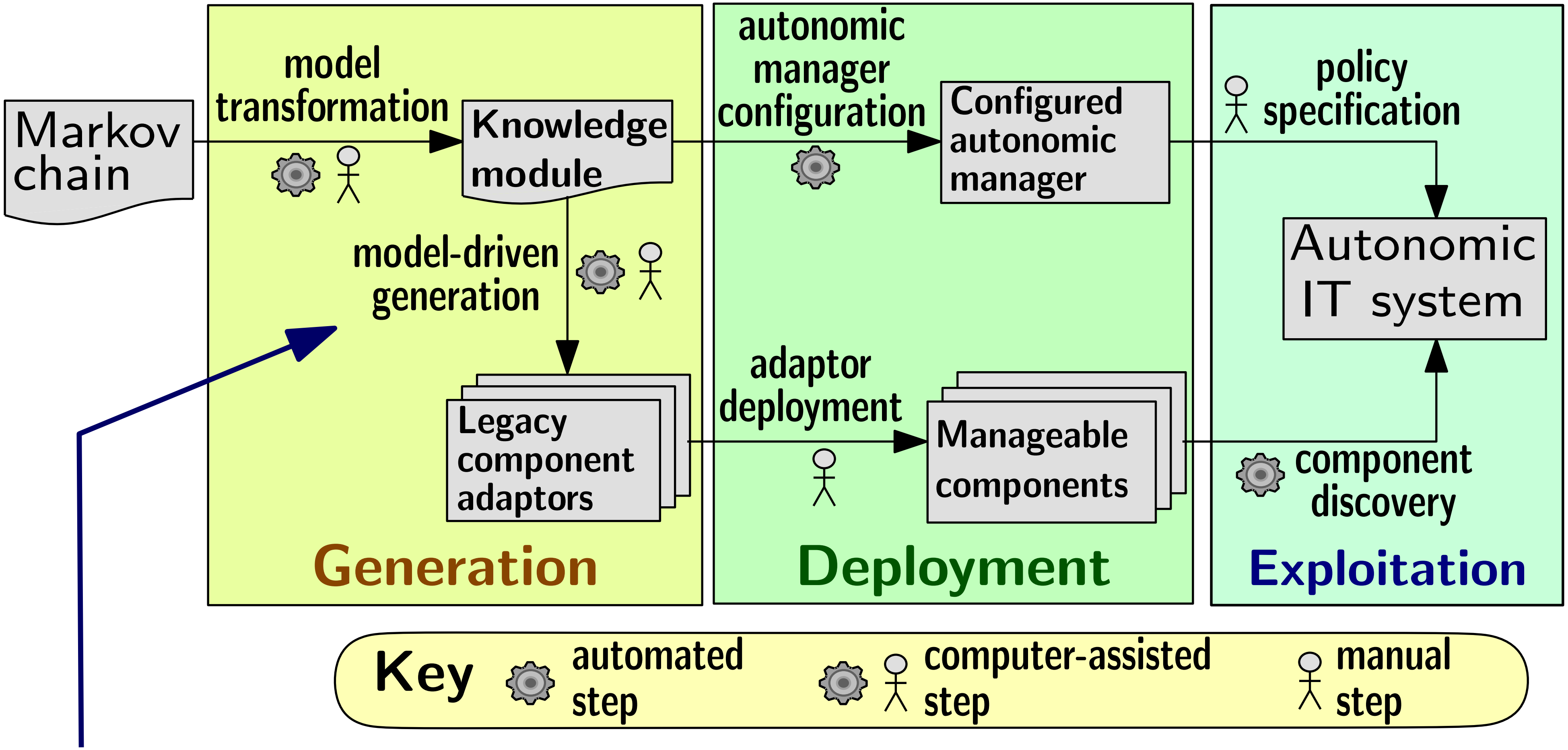
– newly developed

Development method: generation



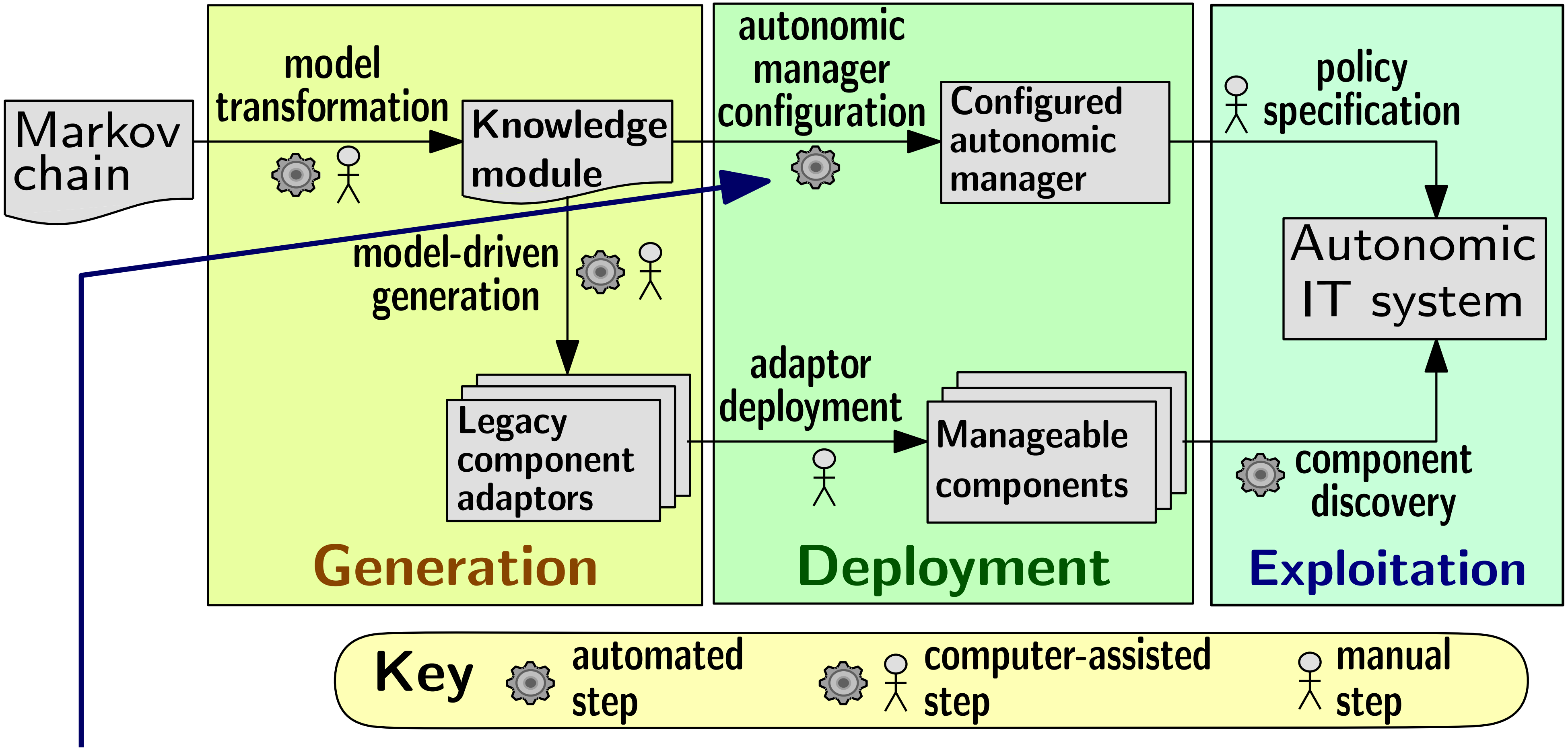
Automated transformation, except for the partition of the Markov chain parameters into state and configuration parameters

Development method: generation



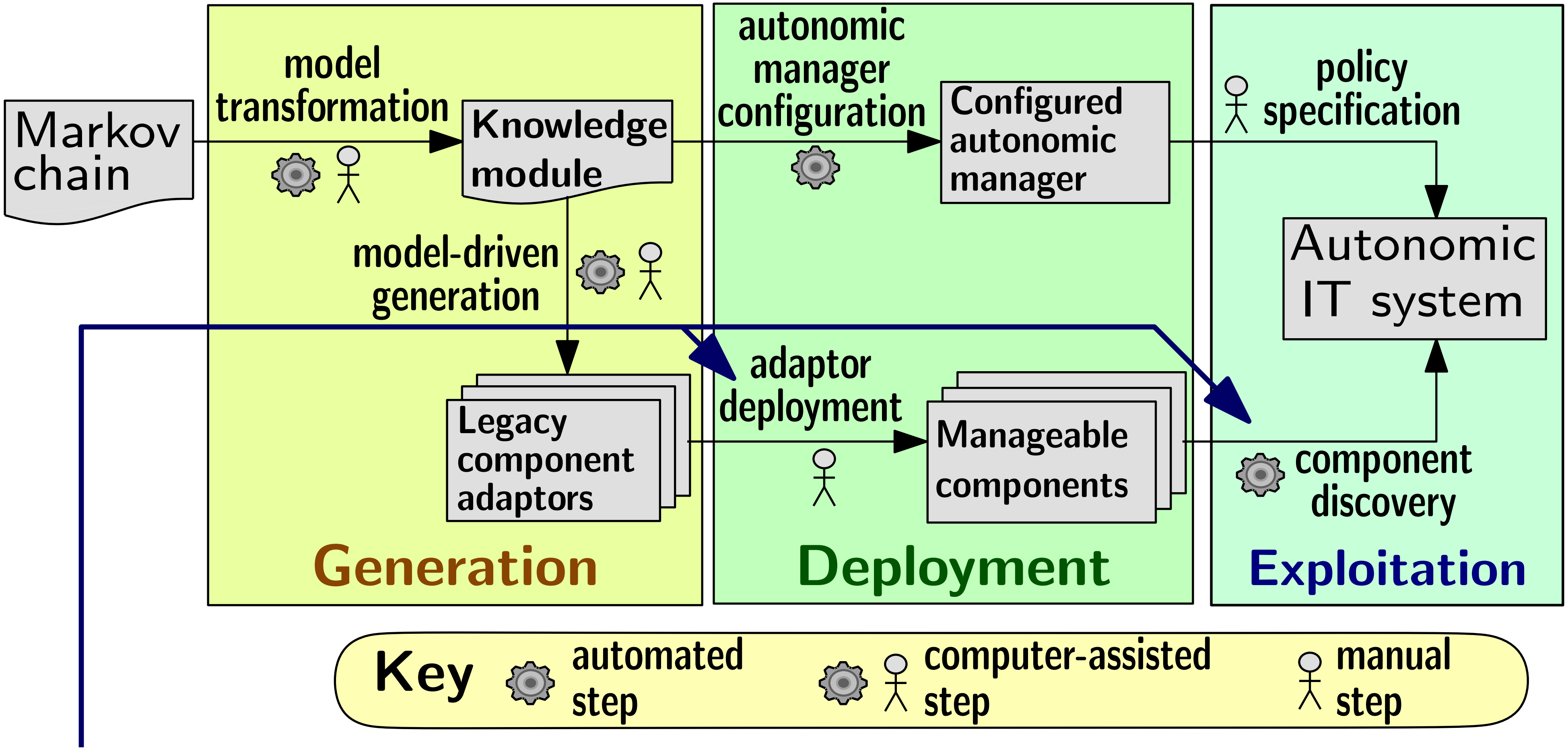
Off-the-shelf tools (XSLT engine, data type generator) used to generate most adaptor code

Development method: deployment



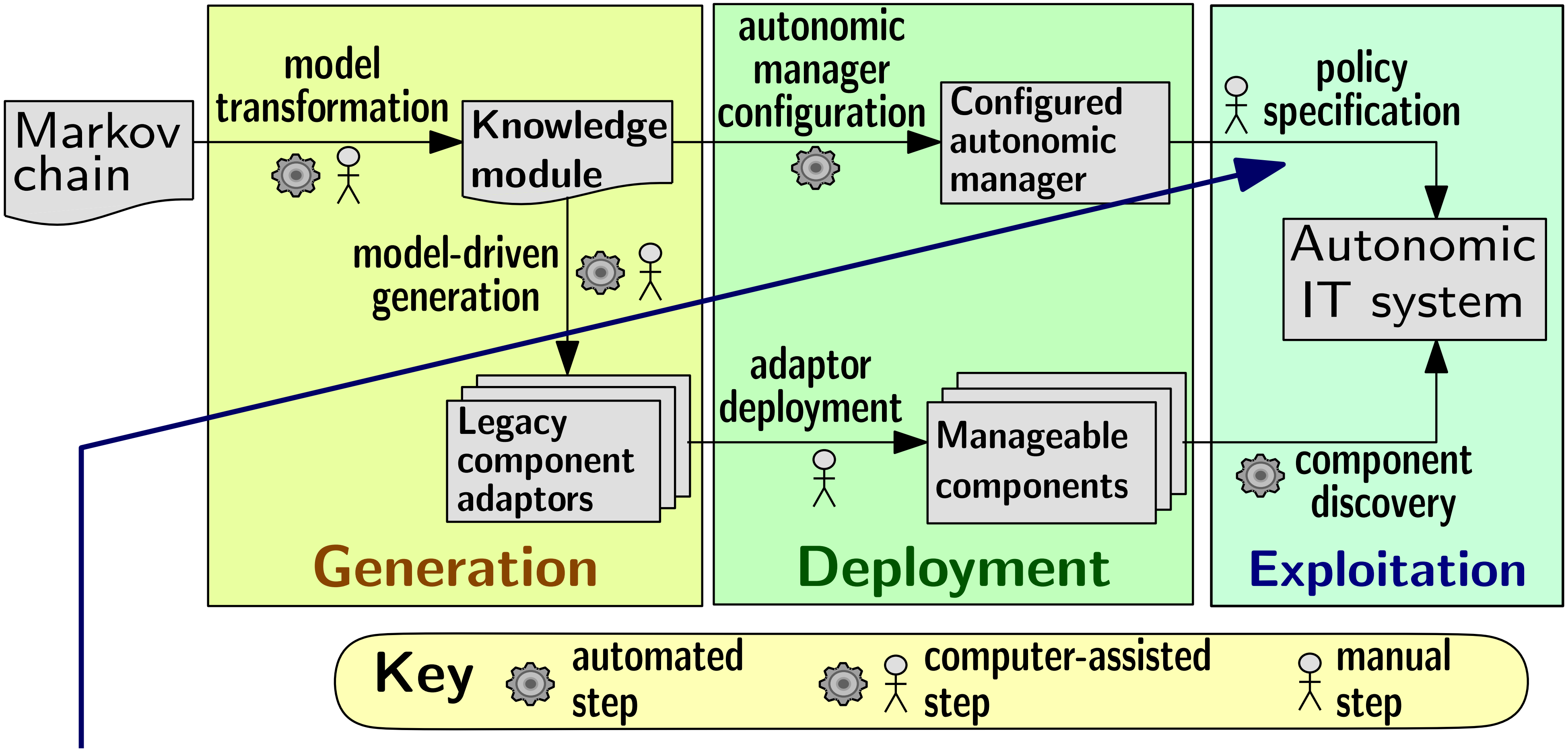
Knowledge module supplied at runtime to autonomic manager instance

Development method: deployment



Adaptor deployment leads to automatic component discovery by the autonomic manager

Development method: exploitation



System objectives specified by system administrator

– multi-objective system *utility* defined in terms of quantitative properties associated with Markov chain

Case study 1: adaptive power management

Power-management enabled devices (disk drives, processors, ...)

- states with different power consumptions and service rates
- state transitions controlled by power manager component

Case study 1: adaptive power management

Power-management enabled devices (disk drives, processors, ...)

- states with different power consumptions and service rates
- state transitions controlled by power manager component

Adjust configuration of power manager dynamically

- to achieve user-specified goals (power use, response time)
- to reflect workload changes (request inter-arrival rate)

Case study 1: adaptive power management

Power-management enabled devices (disk drives, processors, ...)

- states with different power consumptions and service rates
- state transitions controlled by power manager component

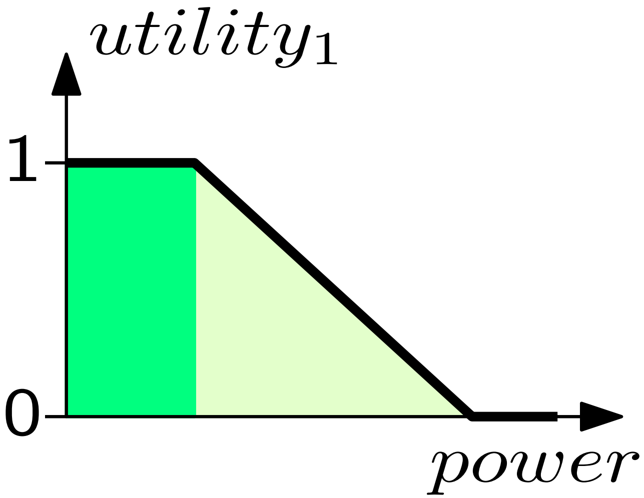
Adjust configuration of power manager dynamically

- to achieve user-specified goals (power use, response time)
- to reflect workload changes (request inter-arrival rate)

Application: adaptive power management of simulated disk drive

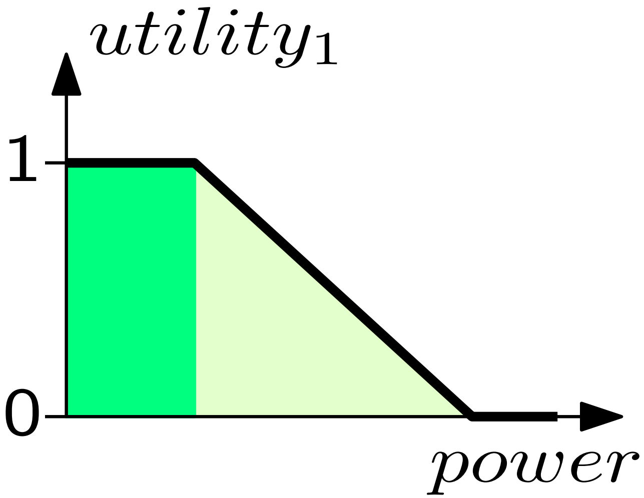
- using existing CTMC model of a three-state Fujitsu disk drive [Qiu, Wu and Pedram, 1999]

Case study 1: adaptive power management

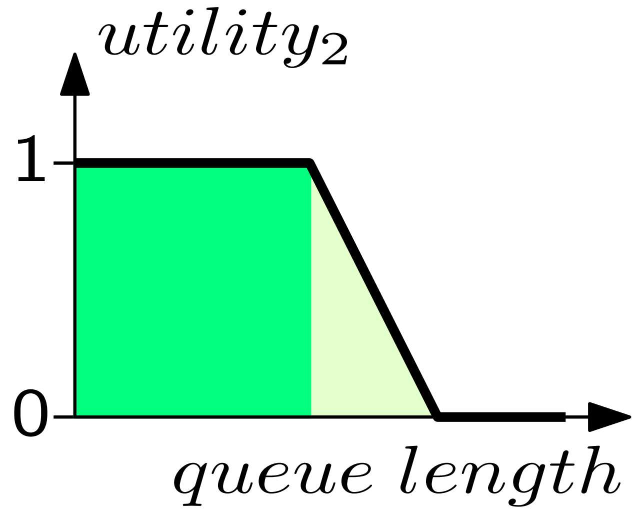


power-related
objective

Case study 1: adaptive power management

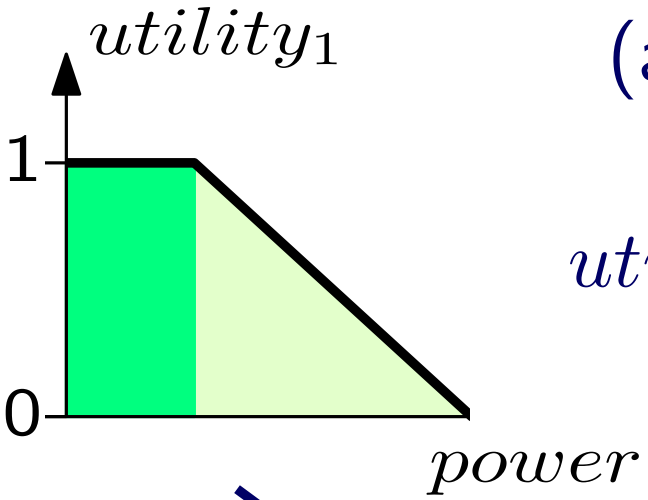


power-related objective



performance-related objective

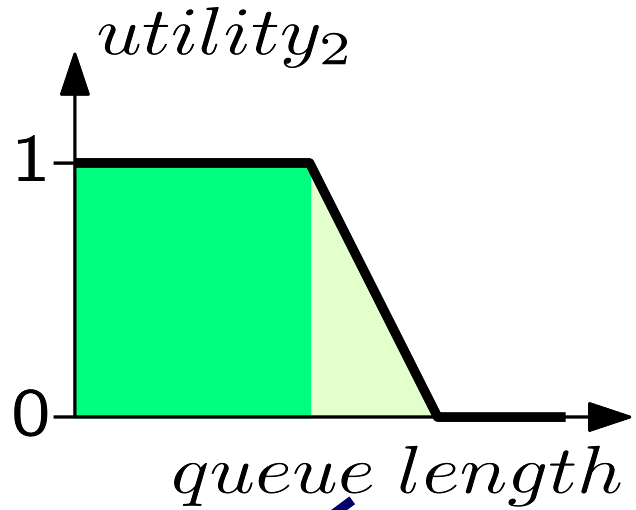
Case study 1: adaptive power management

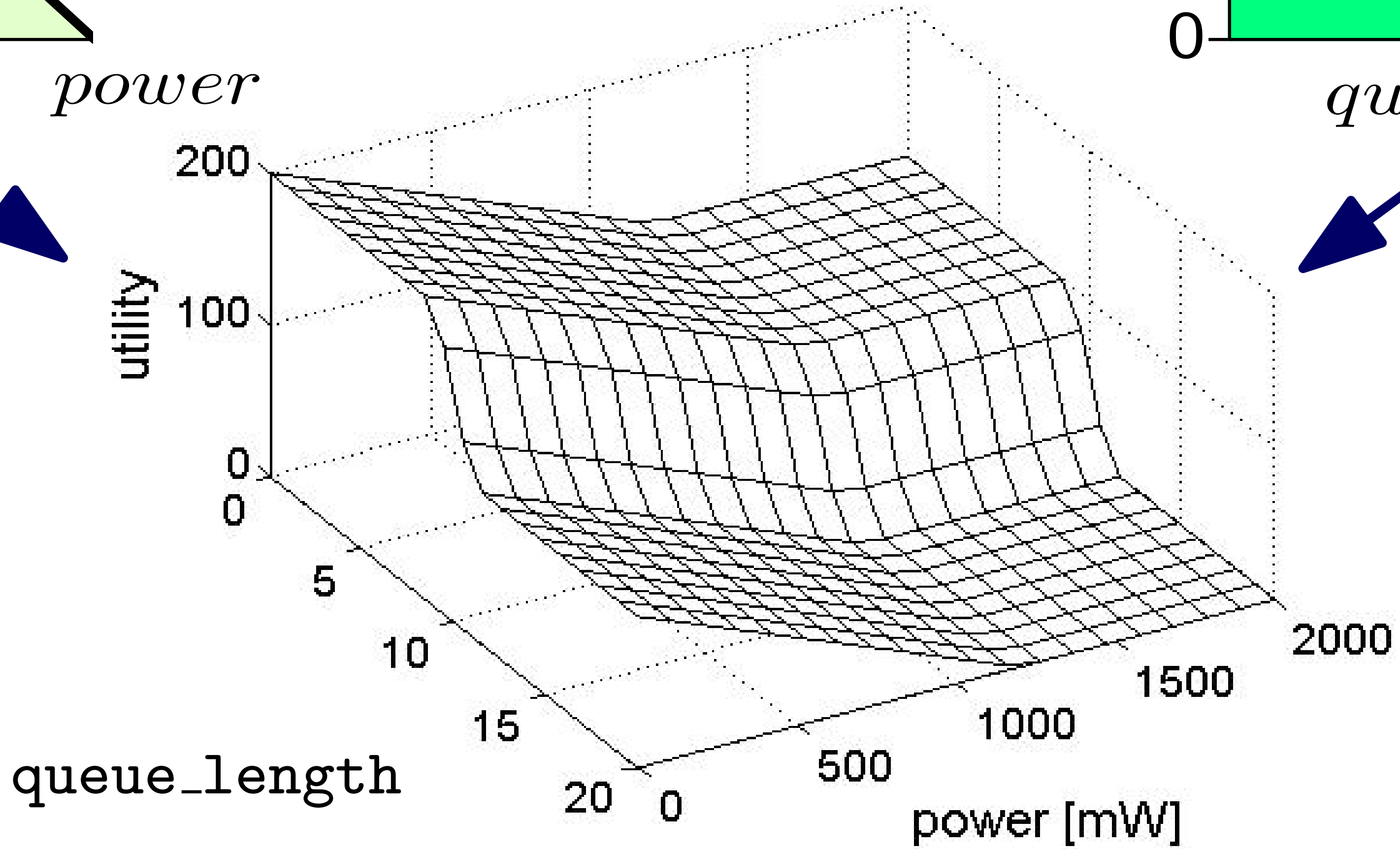


(autonomic computing)

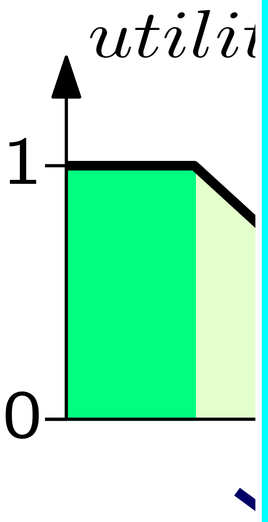
policy: maximise

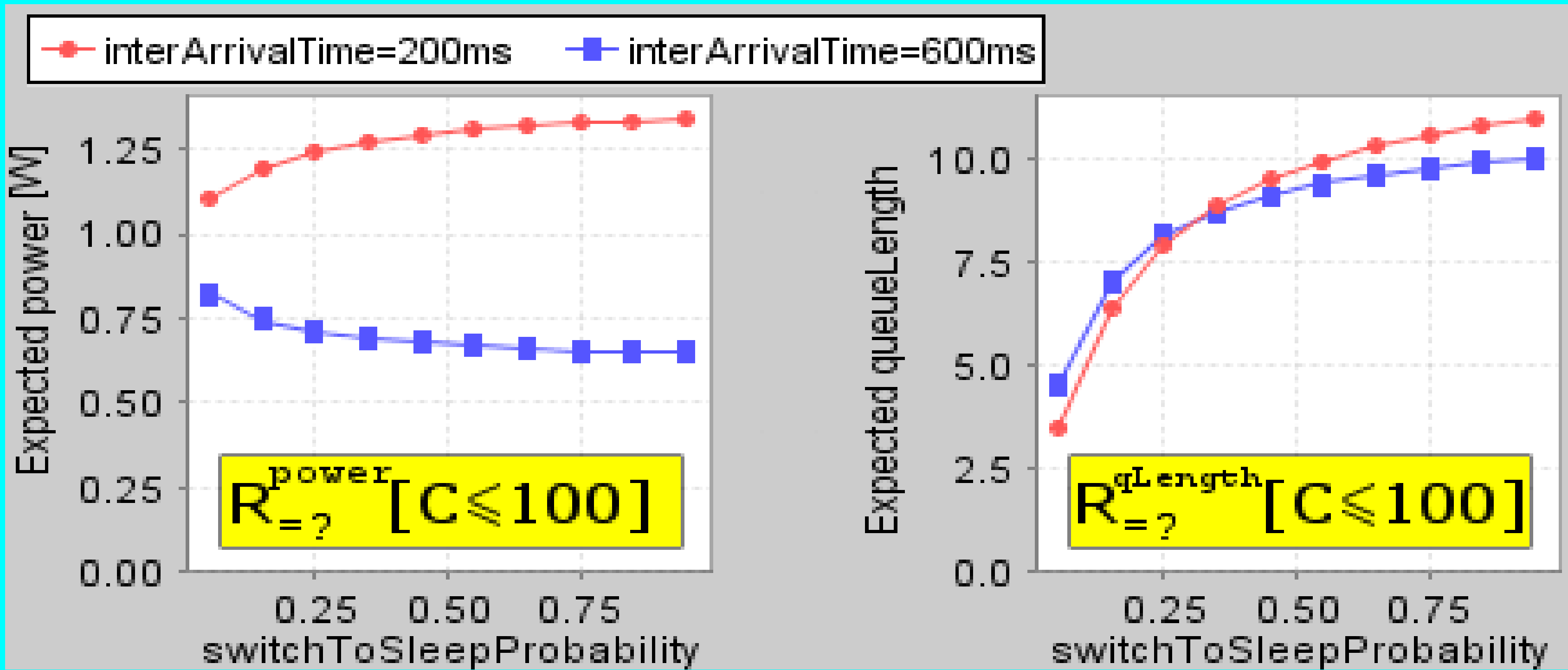
$$utility = \sum_{i=1}^{2} w_i utility_i$$



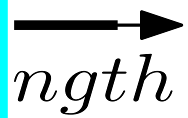


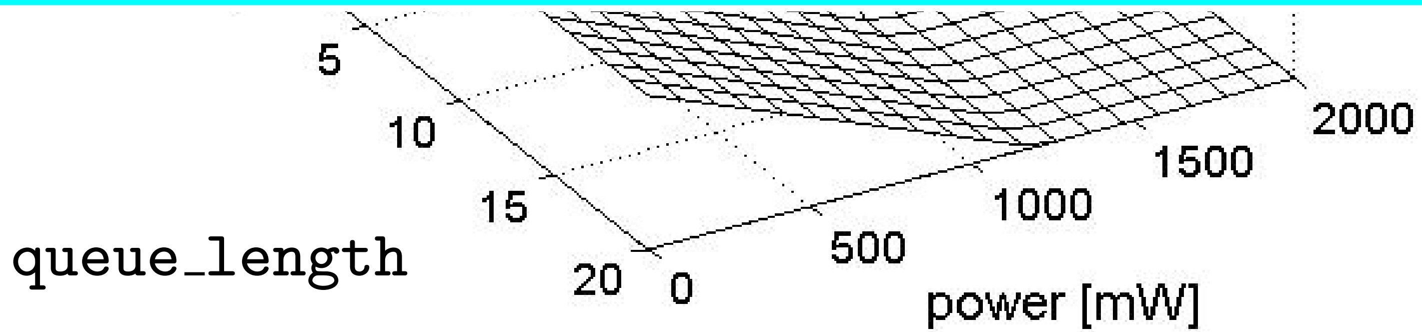
Case study 1: adaptive power management





PRISM *experiments*: multiple verification of quantitative properties across a parameterised family of system models

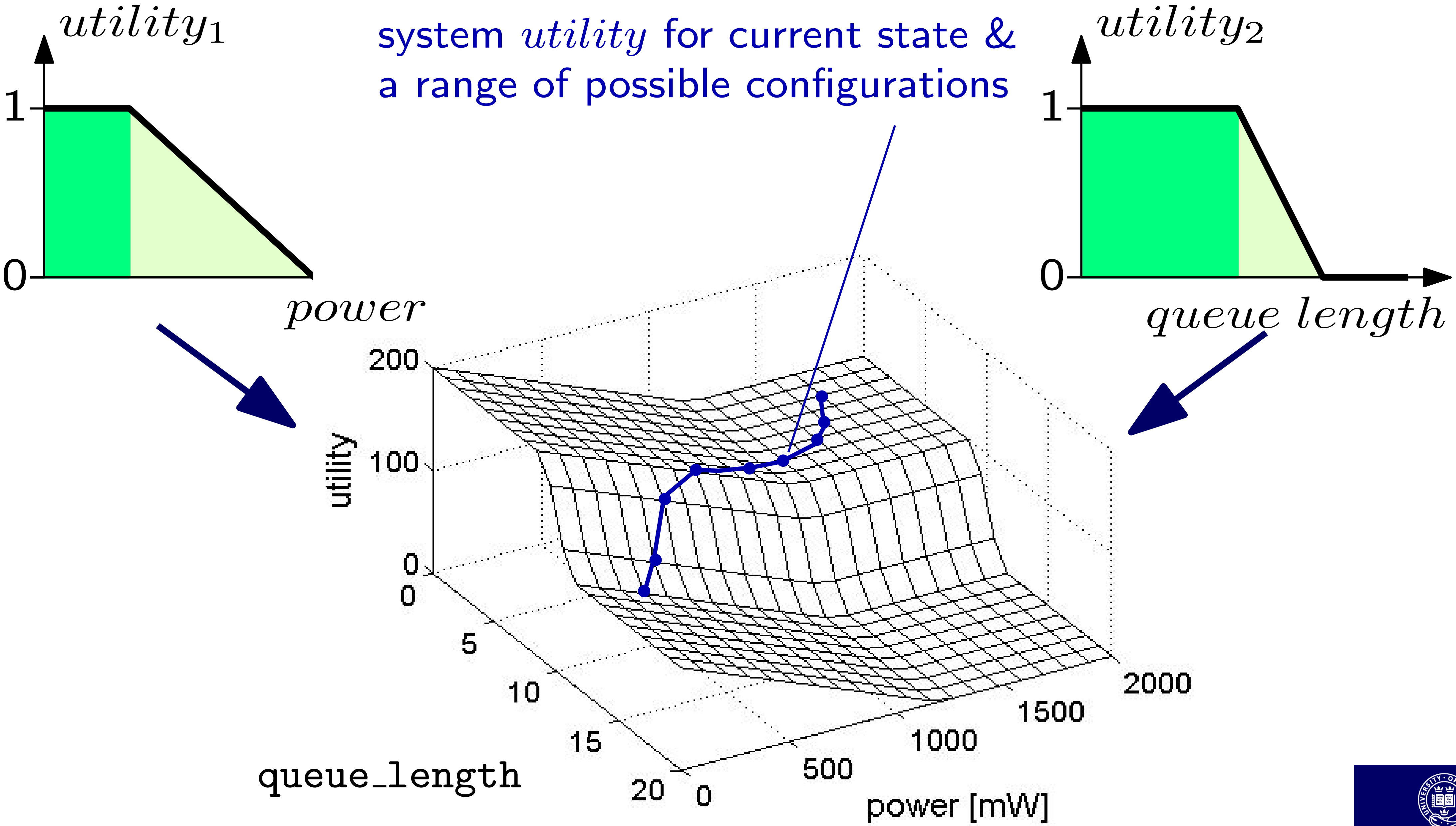




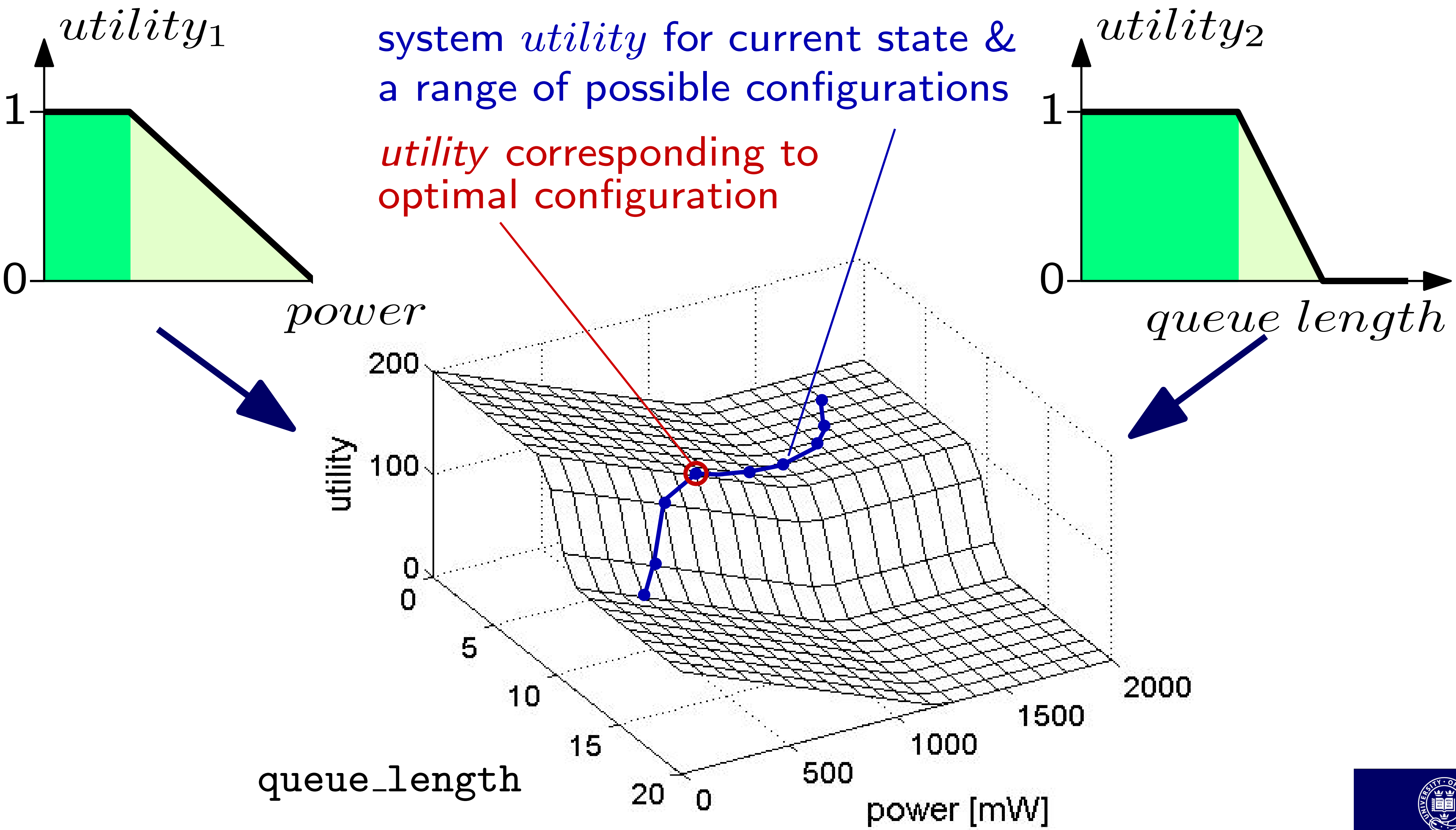
UNIVERSITY OF OXFORD

Case study 1: adaptive power management

system *utility* for current state & a range of possible configurations



Case study 1: adaptive power management



Case study 2: cluster availability management

Multi-objective policy for set of data-centre clusters:

1. achieve target cluster availabilities (user-specified probabilities that clusters are allocated enough servers) in the presence of
   - failures and repairs of data-centre components
   - variable target availabilities & numbers of required servers
2. consider cluster priorities when servers are insufficient
3. minimise overall number of allocated servers

Case study 2: cluster availability management

Multi-objective policy for set of data-centre clusters:

1. achieve target cluster availabilities (user-specified probabilities that clusters are allocated enough servers) in the presence of
   - failures and repairs of data-centre components
   - variable target availabilities & numbers of required servers
2. consider cluster priorities when servers are insufficient
3. minimise overall number of allocated servers

Global policy: decision of a suitable system configuration requires the joint consideration of the quantitative analysis results for all clusters (cf. *local*, device-level policy for the adaptive power management case)

Case study 2: cluster availability management

Multi-objective policy for set of data-centre clusters:

1. achieve target cluster availabilities (user-specified probabilities that clusters are allocated enough servers) in the presence of
   - failures and repairs of data-centre components
   - variable target availabilities & numbers of required servers
2. consider cluster priorities when servers are insufficient
3. minimise overall number of allocated servers

Application: 3-cluster, simulated data-centre (see paper)

- using existing CTMC model of a workstation cluster [Haverkort, Hermanns and Katoen, 2000]

Effectiveness and applicability/limitations

Case study performance and overheads

- consistently bettered existing, heuristic approaches (power management) or delivered guaranteed optimal solution for system with non-linear behaviour (availability management)
- single-figure percentage CPU and memory overhead on average desktop server
- sub-second response time local policy (power management)
- up to 30s response time global policy (availability management)

Effectiveness and applicability/limitations

Applicability/limitations

- for systems whose components exhibit probabilistic/real-time behaviour
- when system objectives can be expressed in terms of performance-/ dependability-related non-functional properties
- subject to ability to perform analysis timely & with acceptable overheads (see later for ways to mitigate high response time/overheads)

Effectiveness and applicability/limitations

Applicability/limitations

- for systems whose components exhibit probabilistic/real-time behaviour
- when system objectives can be expressed in terms of performance-/dependability-related non-functional properties
- subject to ability to perform analysis timely & with acceptable overheads (see later for ways to mitigate high response time/overheads)

Application domains

- resource allocation in IT systems in the presence of failures, variable demand & objectives (VM allocation for cloud computing)
- capacity planning (ongoing project, with industrial collaborator)
- context-aware software systems

Conclusion

Software is increasingly required to adapt dynamically to changes in system state, objectives and environment

Quantitative analysis techniques & tools reached a level of maturity that enables their runtime use to achieve such adaptiveness
(subject to ability to perform analysis timely and with acceptable overheads)

Future research

Techniques to reduce analysis time and overheads

- incremental quantitative analysis of considered system configurations
  - changes in system state are often incremental, hence use verification results for the current state to speed up the analysis for the next state

    [AN Langville, CD Meyer – Updating Markov chains with an eye on Google's PageRank, SIAM J. Matrix Anal. Appl. 27(4), 2006]

Techniques to learn/update system model

- extend existing automata learning algorithms [Biermann&Feldman 1972; Angluin 1987] and devise new algorithms for learning probabilistic/ real-time models
  - to keep model in sync with changes in system behaviour
  - to devise system model starting from given structure

Thank you

Questions?

Oxford Quantitative Analysis and Verification Group
http://www.comlab.ox.ac.uk/activities/qav