Welcome to SEENG 480B / CSC 485B / CSC 586B
Self-Adaptive and Self-Managing Systems

Dr. Hausi A. Müller
Professor
Department of Computer Science
University of Victoria

http://courses.seng.uvic.ca/courses/2013/summer/seng/480b
http://courses.seng.uvic.ca/courses/2013/summer/csc/485b
http://courses.seng.uvic.ca/courses/2013/summer/csc/586b

1

Announcements

- Marking
  - A3 graded
  - Marks are posted on website
- A4
  - posted
  - Due Thu, Aug 6
- A4 Group Presentations
  - Tuesday, Aug 6
  - In class
- Review for final exam
  - Wed, Aug 7
  - Last day of classes
- Final exam
  - Tue, Aug 13, 9:00-12:00 am in ECS 124
  - Closed books, closed notes
  - Materials: entire course
  - Format: like midterm

2

Crib Sheet for Final Exam

- **Crib sheet**: a concise set of notes used for quick reference
  H.A. Müller and N.M. Villegas: Runtime Evolution of Highly Dynamic Software, in *Evolving Software Systems*, T. Mens, A. Serebrenik, and A. Cleve (eds.), Springer, 38 pages, July 2013. In Press.
- Summarizes a significant part of this course
- You will have access to a hard copy during final exam
- Contains answers to selected final exam questions

3

A3 Marking Guide

Part I (50 marks)
Mention the following in one form or another:
- Discuss that utility is for both client and server.
- Mention that the client & server have a lower and upper bound on these utilities.
- Identify that utility functions can be used to define an SLA for a particular service in a business scenario.
- Use an example to illustrate how utility functions can be used.
- Mention how adaptive systems can be used to negotiate SLAs, based on utility, for a client automatically.
- Explain how adaptive systems can be used to enforce SLAs.
- Mention that if something is too cheap clients may not use it because it looks too good to be true

Part II (50 marks)
- Define a simple resource control problem. **(10 marks)**
- Design a simple PID controller for this resource control problem. **(10 marks)**
- Simulate your PID controller using Matlab. **(15 marks)**
- Write a tutorial or software engineering or computer science undergraduate students on how to build a simple PID controller using Matlab. **(15 marks)**

Assignment 4

Part I

In Part I (a) you are to write a summary of the following paper:

H.A. Müller and N.M. Villegas: Runtime Evolution of Highly Dynamic Software Systems," in *Evolving Software Systems*, T. Mens, A. Serebrenik, and A. Cleve (eds.), Springer, 39 pages, July 2013. In Press.

In Part I (b) you are to write a recommendation on how to improve the paper:

The answers to this question should fit into approximately 3-4 typeset pages.

Do not copy verbatim from any source. Cite your sources.

Additional motivation: This paper summarizes a significant part of this course self-adaptive and self-managing systems. You will have access to a hard copy of this paper during the final exam. The answers to selected final exam questions can be found in this paper.

5

Assignment 4 — Groups

Part II - Group Project (3-4 people per group)

Control theory offers several reference models for realizing *adaptive control* where the target system but also the controller is adjusted over time guaranteeing global stability and convergence. Two famous models are *reference adaptive control (MRAC)* and *model identification adaptive control (MIAC)*.

In Part II you are to design an *innovative* application that uses an MRAC or MIAC reference model. Immerse yourself in adaptive control and then design a truly innovative application that could be platform for a company.

Groups

G1	Derek Roberts, Gareth Johnson, Ali Alsaihaty, Noe Hwang
G2	Alessia Knauss, Daniel Conti, Tom Gibson, David Clarke
G3	Daniel Mow, Mohammed Alghamdi, Mustafa Abualsand
G4	Nina Taherimakhsousi, Pratik Jain, Nitin Goyal
G5	Andi Bergen, Pauline Redding, Angela Rook, Fares Almotlag
G6	Nick Phura, Xiyu Bi, Cale McNulty, Heng Wu
G7	Curtis St. Pierre, Muhammad Azam, Gordon Meyer
G8	Carlos Gomez, Lorena Castaneda

6

Graduate Student Research Paper Presentations

- Garlan, D., Cheng, S.-W., Huang, A.-C., Schmerl, B., Steenkiste, P.: Rainbow: Architecture-Based Self-Adaptation with Reusable Infrastructure. *IEEE Computer* 37(10):46-54 (2004) **— Angela Rook, July 23**
- Kramer, J., Magee, J.: Self-Managed Systems: An Architectural Challenge. In: *ACM /IEEE International Conference on Software Engineering 2007 Future of Software Engineering (ICSE),* pp. 259-268 (2007) **— Pratik Jain, July 23**
- Oreizy, P., Medvidovic, N., Taylor, R.N.: Runtime Software Adaptation: Framework, Approaches, and Styles. In: *ACM/IEEE International Conference on Software Engineering (ICSE 2008),* pp. 899-910 (2008) **—Alessia Knauss, July 24**
- Brun, Y., Di Marzo Serugendo, G., Gacek, C. Giese, H. Kienle, H.M., Litoiu, M., Müller, H.M., Pezzè, M., Shaw, M.: *Engineering Self-Adaptive Systems through Feedback Loops.* SE for Self-Adaptive Systems, pp. 48-70 (2009) **— Samra Ramandeep, July 24**
- Kaushik, R.T., Cherkasova, L., Campbell, R.H., Nahrstedt, K.: Lightning: self-adaptive, energy-conserving, multi-zoned, commodity green cloud storage system, *ACM International Symposium on High Performance Distributed Computing (HPDC 2010),* 332-335 (2010) **— Andi Bergen, July 26**

7

Graduate Student Research Paper Presentations

- Villegas, N.M., Müller, H.A., Tamura, G., Duchien, L., Casallas, R.: A framework for evaluating quality-driven self-adaptive software systems. In: *Proc. 6th Int. Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2011),* pp. 80-89 (2011) **— Lorena Castaneda, July 30**
- Ebrahimi, S., Villegas, N.M., Müller, H.A., Thomo, A.: SmarterDeals: a context-aware deal recommendation system based on the SmarterContext engine. *CASCON 2012:* 116-130 (2012) **— Nina Taherimakhsousi, July 30**
- McKinley, P.K., Sadjadi, M., Kasten, E.P., Cheng, B.H.C.: Composing Adaptive Software. *IEEE Computer* 37(7):56-64 (2004) **— Carlos Gomez, July 31**
- Tewari, V., Milenkovic, M.: Standards for Autonomic Computing, *Intel Technology Journal,* 10(4):275-284 (2006) **— Nitin Goyal, July 31**

8

Graduate Student Research Paper Presentations

Great Job!

9

Guidelines for Grad Presentations

- Format of presentation
  - Presentation 15-20 mins
  - Q&A 5 mins
  - Practice talk (!)
- Slides
  - High quality
  - Submit slides 2 days before presentation to instructor for approval
  - Submit final slides 1 day after presentation for posting on website
- Talk outline
  - Motivation
  - Problem
  - Approach
  - Relation to what we heard in the course so far
  - Contributions of the paper
- Avoid plagiarism!!
  - Prepare your own talk
  - Critical

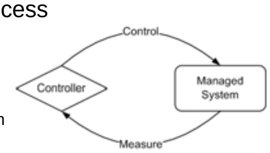
10

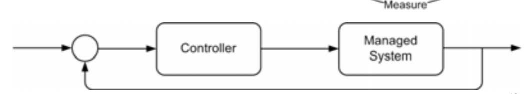
How was your experience? What would you do differently?

11

Feedback Control System

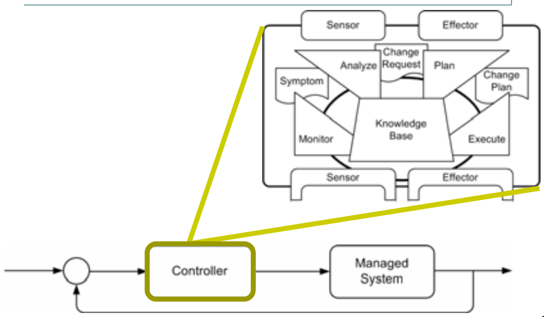
- **Merriam-Webster's Online Dictionary**
  the return to the input of a part of the output of a machine, system, or process
  - producing changes in an electronic circuit to improve performance
  - an automatic control device to provide self-corrective action





12

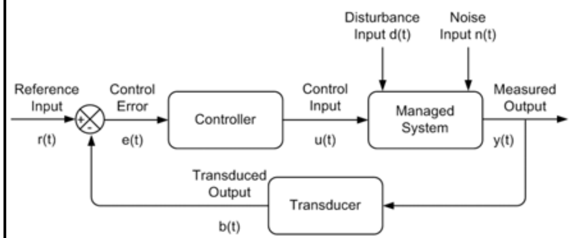
Controller as an Autonomic Element



13

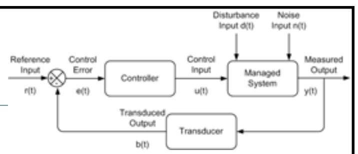
Realization of a Dynamic Architecture

- Feedback control system with disturbance and noise input



14

Realization of a Dynamic Architecture



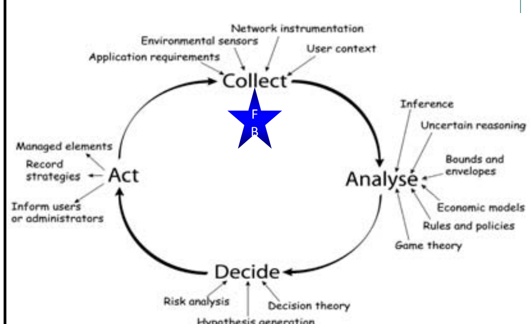
- Reference input
  - Goal, objectives, specified desired output
- Control Error
  - Reference input minus transduced output
- Control Input
  - Parameters which affect behavior of the system—number of threads, CPU, memory
- Disturbance input
  - Affects control input—arrival rate
- Controller
  - Change control input to achieve reference input—design is based on a model of the managed system
- Managed system
  - Dynamical system, process, plant—often characterized by differential equations
- Measured output
  - Measurable feature of the system—response time
- Noise input
  - Affects measured output
- Transducer
  - Transforms measured output to compare with reference

Controller Algorithm based on Managed System Model

- *"All models are wrong, some models are useful."*
  - generally attributed to the statistician George Box
- The design of the controller algorithm is based on a model of the managed system or process
- Approaches
  - Analytical modeling: physical and mathematical laws
  - Experimental modeling: data fitting from observed input and output
- The control algorithm changes $u(t)$ based on the error $e(t) = r(t) - b(t)$
  - Proportional—if $e(t)$ is high, then $u(t)$ should be high
  - Integrative—eliminates transients; sum of all previous errors
  - Derivative—anticipate the trends; rate of change of the error
  - PID—computation based on the error (proportional), the sum of all previous errors (integral) and the rate of change of the error (derivative)

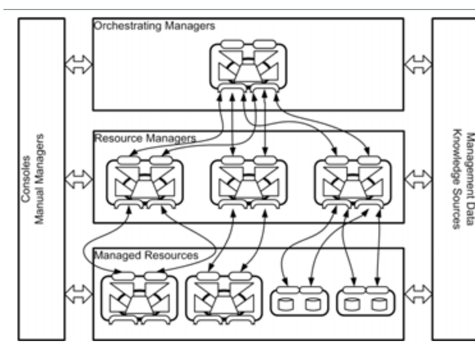
16

Autonomic Feedback Loop



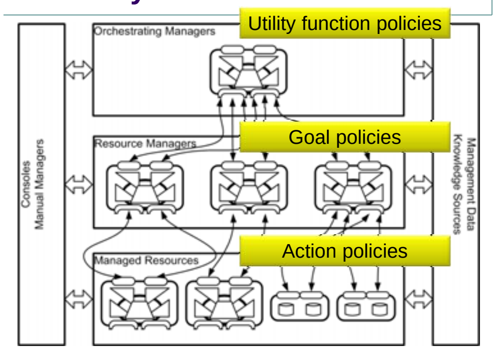
Dobson, S. et al.: A Survey of Autonomic Communications. ACM Transactions on Autonomous and Adaptive Systems (TAAS) 1(2):223-259 (2006)

Hierarchy of Autonomic Elements



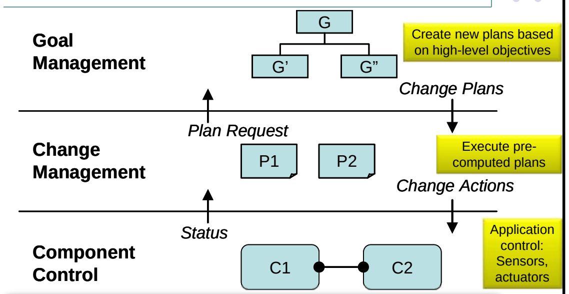
18

Hierarchy of Autonomic Elements



19

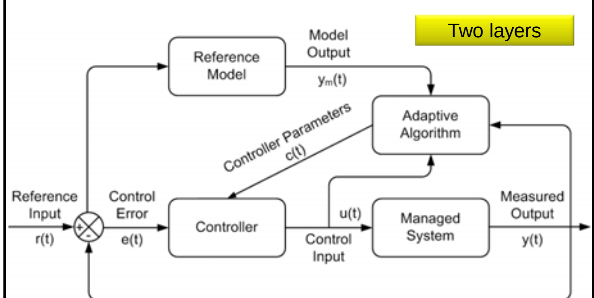
Robotics Inspired Three-Layer Architecture Model



Kramer, Magee: Self-Managed Systems—An Architectural Challenge, Future of Software Engineering (FoSE 2007), ICSE 2007.

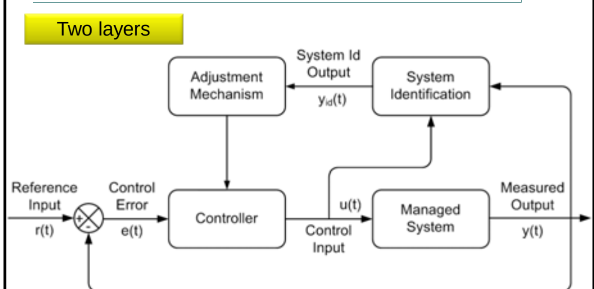
20

Adaptive Control—MRAC



21

Adaptive Control—MIAC



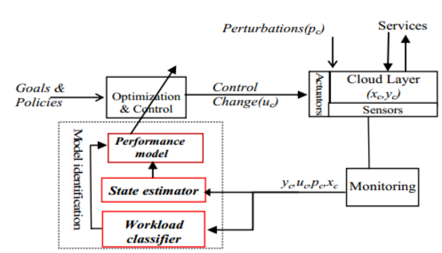
22

Model Predictive Control (MPC)

- Two-level controllers like controllers for adaptive control
- Model predictive controllers rely on dynamic models of the managed system
- Most often linear empirical models obtained by system identification
- The main advantage of MPC is the fact that it allows the current timeslot to be optimized, while taking future time slots into account
- Optimize a finite time-horizon, but only realize the current timeslot
- MPC has the ability to anticipate future events and can take control actions accordingly
- Generic PID controllers do not have predictive abilities

23

Model Predictive Control (MPC)



24

Characteristics of Three-Tier Hierarchical Intelligent Control Systems

- The three-tier architecture is prevalent
  - service-oriented software systems
  - automation systems
  - decision-support systems
  - many other types of adaptive and self-managing systems
- Three layers
  - separate concerns (e.g., three-tier web architecture where the presentation and data tiers are separated by an application or business logic tier)
  - Impose a hierarchy along a dimension where such a dimension represents an extra-functional requirement or quality criterion as outlined
    - performance, internal state, goals, policies, plan sophistication, “intelligence”, or quality of service
  - The scales depend on the actual requirement or criterion of the dimension
    - from specific goals to general goals
    - from high precision to low precision
    - from fast performance to slow performance
    - from stateless to memory of the past and predictions of the future
    - from hard-wired policies to utility-function policies (i.e., trade-off analysis)
  - Rationale for three tiers is usually not explicitly stated, but frequently a natural fit

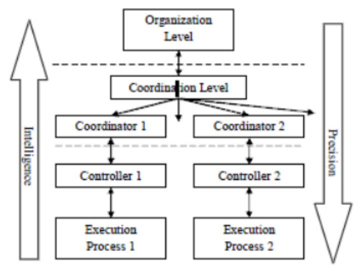
25

Hierarchical Intelligent Control

- AI and robotics communities generated several closely related three-layer reference control architectures:
  - R. A. Brooks: A Robust Layered Control System for a Mobile Robot, *IEEE Journal on Robotics and Automation* RA-2(1), March 1986.
  - R.J. Firby: *Adaptive Execution in Dynamic Domains,* PhD Thesis, TR YALEU/CSD/RR#672, Yale University, 1989.
  - E. Gat: *Reliable Goal-directed Reactive Control for Real-world Autonomous Mobile Robots,* Ph.D. Thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 1991.
  - E. Gat: Three-layer Architectures, Artificial Intelligence and Mobile Robots, MIT/AAAI Press, 1997.
  - T. Shibata & T. Fukuda: Hierarchical Intelligent Control for Robotic Motion, *IEEE Trans. On Neural Networks* 5(5): 823-832, 1994.

26

Hierarchical Intelligent Control System (HICS) Architecture



T. Shibata & T. Fukuda: Hierarchical Intelligent Control for Robotic Motion, *IEEE Trans. On Neural Networks* 5(5): 823-832, 1994

27

HICS Architecture

- Hierarchical Intelligent Control System (HICS)
- HICS is probably the most general reference architecture emerging from AI and robotics
- Three HICS layers
  - Execution
  - Coordination
  - Organization Level
- The complexity of reasoning (i.e., intelligence) increases from the execution to the organization level
- The flexibility of policies decreases from organization to execution (i.e., the precision of increases).

28

Dimensions of Three-Layer Control System Reference Architectures

Environment uncertainty	Human involvement	Algorithm state	Algorithm specification	Policy flexibility	Goal specificity	Real-time performance	Feedback latency
Significant uncertainty about the environment	Orchestrated in part by humans	Algorithms with state for past memory and future predictions	Deliberative services	Utility-function policies	High level goals and extensive planning	No real-time constraints	Feedback loops with long latency
Medium uncertainty about the environment	Fully autonomic but its policies can be adjusted by humans	Algorithms with state reflecting memory of the past	Task procedures	Goal policies	React and respond to situations using pre-computed plans	Selected real-time constraints	Feedback loops with medium latency
No or minimal uncertainty about the environment	Fully autonomic	Stateless algorithms	Control laws	Action policies	Event and component management	Hard real-time constraints	Feedback loops react quickly

29

Dimensions of Three-Layer Control System Reference Architectures

ATLANTIS Gat 1991	HICS Shibata & Fukuda 1994	3T Bonasso, Firby, Gat 1997	IBM ACRA 2006	Kramer & Magee 2007	Adaptive SOA 2008
Deliberator	Organization	Planning	Orchestrating managers	Goal management	User management
Sequencer	Coordination	Sequencing	Resource managers	Change management	Workflow management
Controller	Execution	Skill	Managed Resources	Component control	Service management

30