Autonomic Computing

Hausi A. Müller
Liam O'Brien
Mark Klein
Bill Wood

April 2006

Software Architecture Technology

Unlimited distribution subject to the copyright.

Technical Note
CMU/SEI-2006-TN-006

This work is sponsored by the U.S. Department of Defense.

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

Copyright 2006 Carnegie Mellon University.

NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this report is not intended in any way to infringe on the rights of the trademark holder.

Internal use. Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use. Requests for permission to reproduce this document or prepare derivative works of this document for external and commercial use should be addressed to the SEI Licensing Agent.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Contents

List of Figures

Executive Summary

This report examines selected aspects of autonomic computing, explores some of its strengths and weaknesses, and outlines some of the current research projects being undertaken in this area. It also makes connections between this area and current work in several initiatives at the Carnegie Mellon® Software Engineering Institute (SEI), namely the Predictable Assembly from Certifiable Components and Software Architecture Technology Initiatives. Several pieces of work being undertaken in these initiatives have connections to autonomic computing. Furthermore, the report describes the potential and impact of autonomic computing for Department of Defense (DoD) systems and outlines some of the challenges for the DoD as it moves to exploit autonomic computing technology.

Abstract

This report examines selected aspects of autonomic computing and explores some of the strengths and weaknesses of that technology. It also makes connections between autonomic computing and current work in several initiatives at the Carnegie Mellon® Software Engineering Institute. Furthermore, it describes the potential and impact of autonomic computing for Department of Defense (DoD) systems and outlines some of the challenges for the DoD as it moves to exploit autonomic computing technology.

1 Introduction

Autonomic computing (AC) is an approach to address the complexity and evolution problems in software systems. A software system that operates on its own or with a minimum of human interference according to a set of rules is called autonomic.[1] The term *autonomic* derives from the human body's autonomic nervous system, which controls key functions without conscious awareness or involvement [IBM 02].

IBM started the autonomic computing initiative in 2001 to build self-managing computing systems to overcome the rapidly growing complexity problem. After four years, IBM can point to significant success stories, such as the DB2 Configuration Advisor [Kwan 03] or the Tivoli Risk Manager [IBM 05b]. By April 2005, IBM had woven more than 475 autonomic features into more than 75 products [IBM 05a]. Moreover, IBM has been exceedingly successful in rallying the research community behind their autonomic computing initiative [IBM 02]. Several conferences and workshops emerged, including the Institute of Electrical and Electronics Engineers (IEEE) International Conference on Autonomic Computing (ICAC); the Association for Computing Machinery (ACM) Workshop on Self-Managed Systems (WOSS); the ACM Workshop on Design and Evolution of Autonomic Systems (DEAS); the Autonomic Computing Workshop (AMS); the Conference on Human Impact and Application of Autonomic Computing Systems (CHIACS); the Autonomic Applications Workshop (AAW); the Engineering of Autonomic Systems (EAS) Workshop; and the Workshop on Software Architecture for Dependable Systems (WADS).

Many other companies besides IBM have launched related initiatives including Microsoft's Dynamic Systems Initiative (DSI) [Microsoft 05]; Hewlett Packard's Adaptive Enterprise [HP 03]; Sun's N1 Grid Engine [Sun 03]; Dell's Dynamic Computing Initiative; Hitachi's Harmonious Computing [Hitachi 03]; and Electronic Data Systems' (EDS's) Agile Enterprise [EDS 05]. These initiatives have several objectives in common—reining in the software complexity problem is central to all of them.

Autonomic computing is not a new field but rather an amalgamation of selected theories and practices from several existing areas including control theory, adaptive algorithms, software agents, robotics, fault-tolerant computing, distributed and real-time systems, machine learning, human-computer interaction (HCI), artificial intelligence, and many more. The future of autonomic computing is heavily dependent on the developments and successes in several other technology arenas that provide an infrastructure for autonomic computing systems including Web and grid services, architecture platforms such as service-oriented architecture (SOA) [Papazoglou 03], Open Grid Services Architecture (OGSA) [Globus 05], and pervasive and ubiquitous computing.

[1] The Greek origin for autonomic—"αυτονομος"—literally means "self-governed."

1.1 The Complexity Problem

The increasing complexity of computing systems is overwhelming the capabilities of software developers and system administrators who design, evaluate, integrate, and manage these systems [Ganek 03]. Today, computing systems include very complex infrastructures and operate in complex heterogeneous environments. With the proliferation of handheld devices, the ever-expanding spectrum of users, and the emergence of the information economy with the advent of the Web, computing vendors have difficulty providing an infrastructure to address all the needs of users, devices, and applications.

SOAs with Web services as their core technology have solved many problems, but they have also raised numerous complexity issues. One approach to deal with the business challenges arising from these complexity problems is to make the systems more self-managed or autonomic. For a typical information system consisting of an application server, a Web server, messaging facilities, and layers of middleware and operating systems, the number of tuning parameters exceeds human comprehension and analytical capabilities. Thus, major software and system vendors endeavor to create autonomic, dynamic, or self-managing systems by developing methods, architecture models, middleware, algorithms, and policies to mitigate the complexity problem.

In a 2004 *Economist* article, Kluth investigates how other industrial sectors successfully dealt with complexity [Kluth 04]. He and others have argued that for a technology to be truly successful, its complexity has to disappear. He illustrates his arguments with many examples including the automobile and electricity markets. Only mechanics were able to operate early automobiles successfully. In the early 20th century, companies needed a position of vice president of electricity to deal with power generation and consumption issues. In both cases, the respective industries managed to reduce the need for human expertise and simplify the usage of the underlying technology. However, usage simplicity comes with an increased complexity of the overall system (e.g., what is “under the hood”). Basically for every mouse click or return we take out of the user experience, 20 things have to happen in the software behind the scenes. Given this historical perspective with this predictable path of technology evolution, maybe there is hope for the information technology sector.

1.2 The Evolution Problem

By attacking the software complexity problem through technology simplification and automation, autonomic computing also promises to solve selected software evolution problems. Instrumenting software systems with autonomic technology will allow us to monitor or verify requirements (functional or nonfunctional) over long periods of time. For example, self-managing systems will be able to monitor and control the brittleness of legacy systems, provide automatic updates to evolve installed software, adapt safety-critical systems without halting them, immunize computers against malware automatically, facilitate enterprise integration with self-managing integration mechanisms, document architectural

drift by equipping systems with architecture analysis frameworks, and keep the values of quality attributes within desired ranges.

This report is organized as follows. Section 2 presents basic concepts and foundations within autonomic computing and explores some of the current and emerging technologies. Section 3 outlines a sample of AC research projects. Section 4 analyzes some of the limitations and problems of AC. Section 5 examines the impact of AC technology to some of the work currently being undertaken at the Carnegie Mellon® Software Engineering Institute (SEI). Section 6 examines the potential for AC technology within Department of Defense (DoD) systems. Section 7 concludes the report and outlines some steps that the SEI can take to continue investigation in this area. At the end of this report is an extensive bibliography, which shows how quickly the field has grown since 2001. This bibliography contains a number of documents that would be good starting points for newcomers to the field of autonomic computing.

® Carnegie Mellon is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

2 Foundations and Concepts

2.1 The Ubiquitous Control Loop

At the heart of an autonomic system is a control system, which is a combination of components that act together to maintain actual system attribute values close to desired specifications, as shown in Figure 1. Open-loop control systems (e.g., automatic toasters and alarm clocks) are those in which the output has no effect on the input. Closed-loop control systems (e.g., thermostats or automotive cruise-control systems) are those in which the output has an effect on the input in such a way as to maintain a desired output value. An autonomic system embodies one or more closed control loops. A closed-loop system includes some way to sense changes in the managed element, so corrective action can be taken. The speed with which a simple closed-loop control system moves to correct its output is described by its damping ratio and natural frequency. Properties of a control system include spatial and temporal separability of the controller from the controlled element, evolvability of the controller, and filtering of the controlled resource.

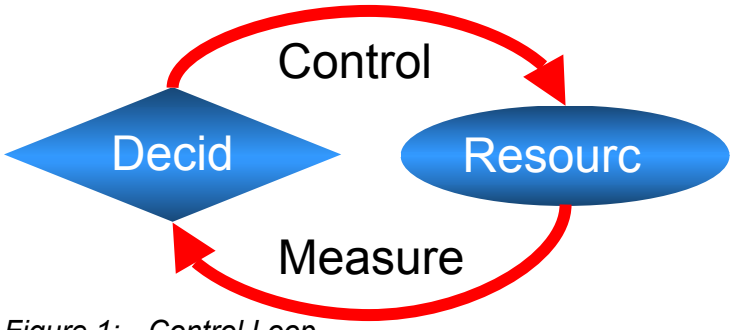

Figure 1: Control Loop

Numerous engineering products embody open-loop or closed-loop control systems. The AC community often refers to the human autonomic nervous system (ANS) with its many control loops as a prototypical example. The ANS monitors and regulates vital signs such as body temperature, heart rate, blood pressure, pupil dilation, digestion blood sugar, breathing rate, immune response, and many more involuntary, reflexive responses in our bodies. The ANS consists of two separate divisions called the parasympathetic nervous system, which regulates day-to-day internal processes and behaviors, and the sympathetic nervous system, which deals with stressful situations. Studying the ANS might be instructive for the design of autonomic software systems. For example, physically separating the control loops that deal with normal and abnormal situations might be a useful design idea for autonomic software systems.

2.2 Autonomic Elements

IBM researchers have established an architectural framework for autonomic systems [Kephart 03]. An autonomic system consists of a set of autonomic elements that contain and manage resources and deliver services to humans or other autonomic elements. An autonomic element consists of one autonomic manager and one or more managed elements. At the core of an autonomic element is a control loop that integrates the manager with the managed element. The autonomic manager consists of sensors, effectors, and a five-component analysis and planning engine as depicted in Figure 2. The monitor observes the sensors, filters the data collected from them, and then stores the distilled data in the knowledge base. The analysis engine compares the collected data against the desired sensor values also stored in the knowledge base. The planning engine devises strategies to correct the trends identified by the planning engine. The execution engine finally adjusts parameters of the managed element by means of effectors and stores the affected values in the knowledge base.

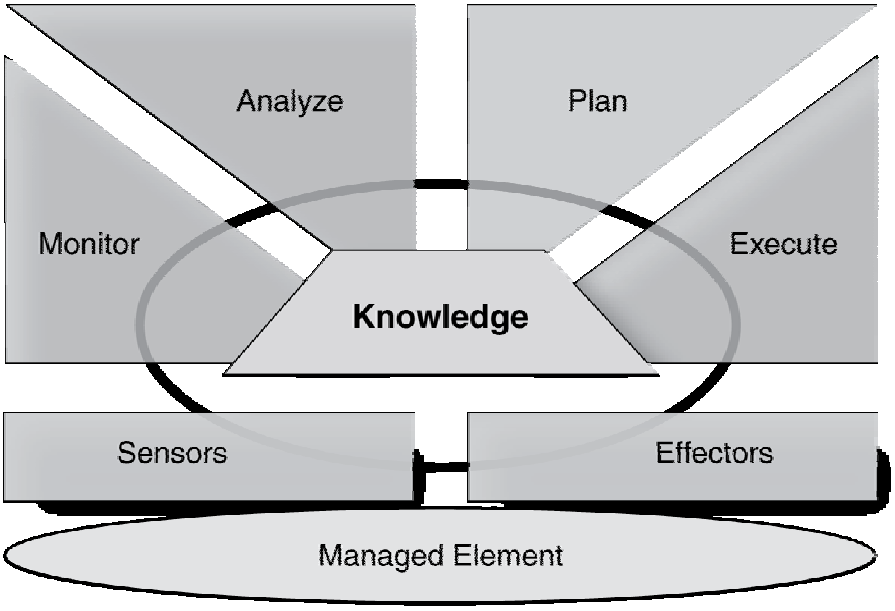

Figure 2: Autonomic Element

An autonomic element manages its own internal state and its interactions with its environment (i.e., other autonomic elements). An element's internal behavior and its relationships with other elements are driven by the goals and policies the designers have built into the system. Autonomic elements can be arranged as strict hierarchies or graphs.

Touchpoints represent the interface between the autonomic manager and the managed element. Through touchpoints, autonomic managers control a managed resource or another autonomic element. It is imperative that touchpoints are standardized, so autonomic managers can manipulate other autonomic elements in a uniform manner. That is, a single standard manageability interface, as provided by a touchpoint, can be used to manage routers, servers,

application software, middleware, a Web service, or any other autonomic element. This is one of the key values of AC: a single manageability interface, rather than the numerous sorts of manageability interfaces that exist today, to manage various types of resources [Miller 05e]. Thus, a touchpoint constitutes a level of indirection and is the key to adaptability.

A manageability interface consists of a sensor and an effector interface. The sensor interface enables an autonomic manager to retrieve information from the managed element through the touchpoint using two interaction styles: (1) request-response for solicited (queried) data retrieval and (2) send-notification for unsolicited (event-driven) data retrieval. The effector interface enables an autonomic manager to manage the managed element through the touchpoint with two interaction types: (1) perform-operation to control the behavior (e.g., adjust parameters or send commands) and (2) solicit-response to enable call-back functions.

IBM has proposed interface standards for touchpoints and developed a simulator to aid the development of autonomic managers. The Touchpoint Simulator can be used to simulate different managed elements and resources and to verify standard interface compliance.

2.3 Characteristics of Autonomic Systems

An autonomic system can self-configure at runtime to meet changing operating environments, self-tune to optimize its performance, self-heal when it encounters unexpected obstacles during its operation, and—of particular current interest—protect itself from malicious attacks. Research and development teams concentrate on developing theories, methods, tools, and technology for building self-healing, self-configuring, self-optimizing, and self-protecting systems, as depicted in Figure 3. An autonomic system can self-manage anything including a single property or multiple properties.

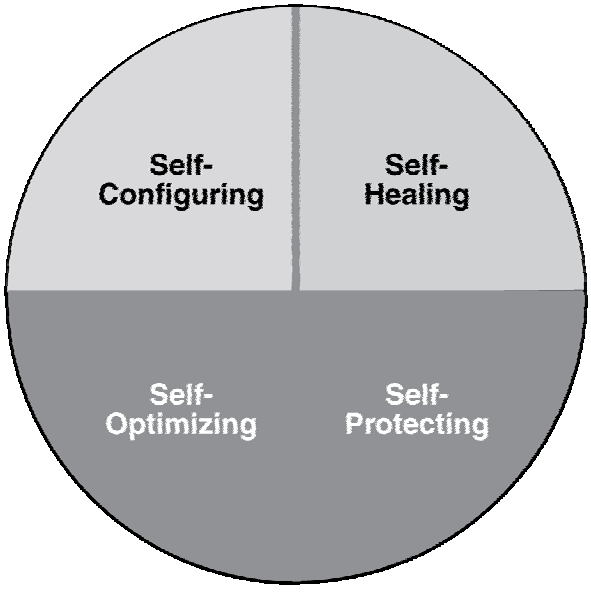

Figure 3: Autonomic Characteristics

An autonomic system has the following characteristics:

- **reflexivity:** An autonomic system must have detailed knowledge of its components, current status, capabilities, limits, boundaries, interdependencies with other systems, and available resources. Moreover, the system must be aware of its possible configurations and how they affect particular nonfunctional requirements.
- **self-configuring:** Self-configuring systems provide increased responsiveness by adapting to a dynamically changing environment. A self-configuring system must be able to configure and reconfigure itself under varying and unpredictable conditions. Varying degrees of end-user involvement should be allowed, from user-based reconfiguration to automatic reconfiguration based on monitoring and feedback loops. For example, the user may be given the option of reconfiguring the system at runtime; alternatively, adaptive algorithms could learn the best configurations to achieve mandated performance or to service any other desired functional or nonfunctional requirement. Variability can be accommodated at design time (e.g., by implementing goal graphs) or at runtime (e.g., by adjusting parameters). Systems should be designed to provide configurability at a feature level with capabilities such as separation of concerns, levels of indirection, integration mechanisms (data and control), scripting layers, plug and play, and set-up wizards. Adaptive algorithms have to detect and respond to short-term and long-term trends.
- **self-optimizing:** Self-optimizing systems provide operational efficiency by tuning resources and balancing workloads. Such a system will continually monitor and tune its resources and operations. In general, the system will continually seek to optimize its operation with respect to a set of prioritized nonfunctional requirements to meet the ever-changing needs of the application environment. Capabilities such as repartitioning, reclustering, load balancing, and rerouting must be designed into the system to provide self-optimization. Again, adaptive algorithms, along with other systems, are needed for monitoring and response.
- **self-healing:** Self-healing systems provide resiliency by discovering and preventing disruptions as well as recovering from malfunctions. Such a system will be able to recover—without loss of data or noticeable delays in processing—from routine and extraordinary events that might cause some of its parts to malfunction. Self-recovery means that the system will select, possibly with user input, an alternative configuration to the one it is currently using and will switch to that configuration with minimal loss of information or delay.
- **self-protecting:** Self-protecting systems secure information and resources by anticipating, detecting, and protecting against attacks. Such a system will be capable of protecting itself by detecting and counteracting threats through the use of pattern recognition and other techniques. This capability means that the design of the system will include an analysis of the vulnerabilities and the inclusion of protective mechanisms that might be employed when a threat is detected. The design must provide for capabilities to recognize and handle different kinds of threats in various contexts more easily, thereby reducing the burden on administrators.

- **adapting:** At the core of the complexity problem addressed by the AC initiative is the problem of evaluating complex tradeoffs to make informed decisions. Most of the characteristics listed above are founded on the ability of an autonomic system to monitor its performance and its environment and respond to changes by switching to a different behavior. At the core of this ability is a control loop. Sensors observe an activity of a controlled process, a controller component decides what has to be done, and then the controller component executes the required operations through a set of actuators. The adaptive mechanisms to be explored will be inspired by work on machine learning, multi-agent systems, and control theory.

2.4 Policies

Autonomic elements can function at different levels of abstraction. At the lowest levels, the capabilities and the interaction range of an autonomic element are limited and hard-coded. At higher levels, elements pursue more flexible goals specified with policies, and the relationships among elements are flexible and may evolve.

Recently, Kephart and Walsh proposed a unified framework for AC policies based on the well-understood notions of states and actions [Kephart 04]. In this framework, a policy will directly or indirectly cause an action to be taken that transitions the system into a new state. Kephart and Walsh distinguish three types of AC policies, which correspond to different levels of abstraction, as follows:

1. **action policies:** An action policy dictates the action that should be taken when the system is in a given current state. Typically this action takes the form of "IF (condition) THEN (action)," where the condition specifies either a specific state or a set of possible states that all satisfy the given condition. Note that the state that will be reached by taking the given action is not specified explicitly. Presumably, the author knows which state will be reached upon taking the recommended action and deems this state more desirable than states that would be reached via alternative actions. This type of policy is generally necessary to ensure that the system is exhibiting rational behavior.
2. **goal policies:** Rather than specifying exactly what to do in the current state, goal policies specify either a single desired state, or one or more criteria that characterize an entire set of desired states. Implicitly, any member of this set is equally acceptable. Rather than relying on a human to explicitly encode rational behavior, as in action policies, the system generates rational behavior itself from the goal policy. This type of policy permits greater flexibility and frees human policy makers from the "need to know" low-level details of system function, at the cost of requiring reasonably sophisticated planning or modeling algorithms.
3. **utility-function policies:** A utility-function policy is an objective function that expresses the value of each possible state. Utility-function policies generalize goal policies. Instead of performing a binary classification into desirable versus undesirable states, they ascribe a real-valued scalar desirability to each state. Because the most desired state is not specified in advance, it is computed on a recurrent basis by selecting the state that

has the highest utility from the present collection of feasible states. Utility-function policies provide more fine-grained and flexible specification of behavior than goal and action policies. In situations in which multiple goal policies would conflict (i.e., they could not be simultaneously achieved), utility-function policies allow for unambiguous, rational decision making by specifying the appropriate tradeoff. On the other hand, utility-function policies can require policy authors to specify a multidimensional set of preferences, which may be difficult to elicit; furthermore they require the use of modeling, optimization, and possibly other algorithms.

2.5 Issues of Trust

Dealing with issues of trust is critical for the successful design, implementation, and operation of AC systems. Since an autonomic system is supposed to reduce human interference or even take over certain heretofore human duties, it is imperative to make trust development a core component of its design. Even when users begin to trust the policies hard-wired into low-level autonomic elements, it is a big step to gain their trust in higher level autonomic elements that use these low-level elements as part of their policies.

Autonomic elements are instrumented to provide feedback to users beyond what they provide as their service. Deciding what kind of feedback to provide and how to instrument the autonomic elements is a difficult problem. The trust feedback required by users will evolve with the evolution of the autonomic system. However, the AC field can draw experience from the automation and HCI communities to tackle these problems.

Autonomic systems can become more trustable by actively communicating with their users. Improved interaction will also allow these systems to be more autonomous over time, exhibiting increased initiative without losing the users' trust. Higher trustability— and usability—should, in turn, lead to improved adoptability.

2.6 Evolution Rather Than Revolution

Most existing systems cannot be redesigned and redeveloped from scratch to engineer autonomic capabilities into them. Rather, self-management capabilities have to be added gradually and incrementally—one component (i.e., architecture, subsystem, or service) at a time.

With the proliferation of autonomic components, users will impose increasingly more demands with respect to functional and nonfunctional requirements for autonomicity. Thus, the process of equipping software systems with autonomic technology will be evolutionary rather than revolutionary. Moreover, the evolution of autonomic systems will happen at two levels—(1) the introduction of autonomic components into existing systems and (2) the change of requirements with the proliferation and integration of autonomic system elements.

IBM has defined five levels of maturity as depicted in Figure 4 to characterize the gradual injection of autonomicity into software systems.

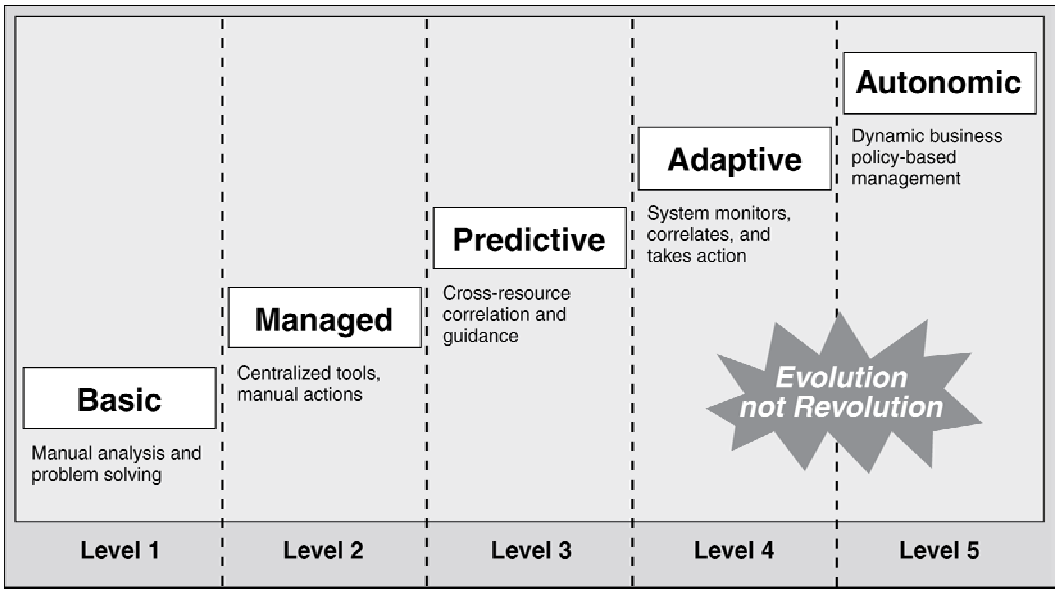

Figure 4: Increasing Autonomic Functionality

3 Selected Research Projects

There is a large range of AC projects in academia and industry. Because many research areas contribute to the AC field, it is difficult to identify "pure" autonomic research projects. In particular, many highly relevant projects dealing with agents, grid computing (e.g., OceanStore), ubiquitous computing, Web services, or quality of service (e.g., Q-Fabric) are not discussed in this section.

3.1 Unity Project and Autonomic Computing Toolkit

Unity is a research project, carried out at IBM's Thomas J. Watson Research Center, that explores some of the behaviors and relationships that will allow complex computing systems to self-manage—that is, to be self-configuring, self-optimizing, self-protecting, and self-healing [Chess 04]. The four principal aspects examined with the Unity prototype environment are (1) the overall architecture of the system, (2) the role of utility functions in decision making within the system, (3) the way the system uses goal-driven self-assembly to configure itself, and (4) the design patterns that enable self-healing within the system.

The IBM Autonomic Computing Toolkit is a collection of technologies, tools, scenarios, and documentation that is designed for users who want to learn, adapt, and develop autonomic behavior in their products and systems. These tools, technologies, and scenarios can be grouped into three main categories: (1) problem determination, (2) solution installation and deployment, and (3) integrated solutions [IBM 05c].

3.2 Information Economies Project

One of the most ambitious AC projects is the information economies project conducted at IBM's Thomas J. Watson Research Center under the leadership of Kephart [Kephart 05a]. The project envisions a free-market information economy on the Internet where software agents buy and sell a variety of information goods and services. The economically motivated autonomic elements are expected to find and process information and disseminate it to humans and, increasingly, to other autonomic elements. The milieu of economic autonomic elements will naturally evolve, the elements will morph from facilitators into decision makers, and their degree of autonomy and responsibility will continually increase. This project envisions millions of autonomic elements buying and selling in a large-scale cooperative or competitive information economy. This is in stark contrast to most autonomic system designs that typically incorporate few autonomic elements.

3.3 KX Project (Columbia University)

The goal of the KX Project, led by Kaiser, is to inject AC technology into legacy software systems without any need to understand or modify the code of the existing system. The project designed a meta-architecture implemented as an active middleware infrastructure to add autonomic services explicitly via an attached feedback loop that provides continual monitoring and, as needed, reconfiguration or repair. The lightweight design and separation of concerns enable easy adoption of individual components, as well as the full infrastructure, for use with a large variety of systems [Kaiser 03, Griffith 05].

3.4 Rainbow Project (Carnegie Mellon University)

The Rainbow Project, led by Garlan, investigates the use of software architectural models at runtime as the basis for reflection and dynamic adaptation [Cheng 04b, Garlan 04]. The project aims to provide capabilities that will reduce the need for user intervention in adapting systems to achieve quality goals, improve the dependability of changes, and support a new breed of systems that can perform reliable self-modification in response to dynamic changes in the environment. The goal of the DiscoTect subproject is to produce architectural views by observing a running system [Yan 04a].

3.5 ROC Project (University of California Berkeley/Stanford)

The Recovery-Oriented Computing (ROC) Project, led by Patterson, investigates novel techniques for building highly dependable Internet services. ROC emphasized recovery from failures rather than failure avoidance [Patterson 02]. This philosophy is motivated by the observation that even the most robust systems still occasionally encounter failures due to human operator error, transient or permanent hardware failure, and software anomalies resulting from software aging.

3.6 DEAS Project (Universities of Victoria and Toronto, Canada)

The Design and Evolution of Autonomic Application Software (DEAS) Project, led by Müller and Mylopoulos, uses requirements goal models as a foundation for designing autonomic applications. Tasks such as self-configuration, self-optimization, self-healing, and self-protection are often accomplished by switching at runtime to a different system behavior. The goal of the project is to push this kind of system variability from runtime back into design time [Lapouchnian 05]. A second objective of the project is to instrument autonomic architectures with analysis frameworks for autonomic-specific and generic quality attributes using attribute-based architectural styles (ABASs) to ease the evolution of autonomic application software.

3.7 Autonomia Project (University of Arizona)

The goal of the Autonomia Project, led by Hariri, is to develop an infrastructure and tools that provide dynamically programmable control and management services to support the development of autonomic applications [Hariri 03].

3.8 AutoMate Project (Rutgers University)

The overall objective of the AutoMate Project is to investigate key technologies to enable the development of autonomic grid applications that are context aware and capable of self-configuring, self-composing, self-optimizing, and self-adapting [Parashar 03a, Agarwal 03]. Specifically, it will investigate the definition of autonomic components, the development of autonomic applications as dynamic compositions of autonomic components, and the design of key enhancements to existing grid middleware and runtime services to support these applications.

3.9 AMUSE Project (University of Glasgow and Imperial College, UK)

The Autonomic Management of Ubiquitous Systems for E-Health (AMUSE) Project focuses on the architecture and development of autonomous management capabilities for ubiquitous computing environments in general and electronic-health environments in particular [Sventek 05]. AMUSE defines the concept of a self-managed cell (SMC). The project aims to create and develop an SMC and create methods for inter-cell composition, intra-cell federation, and layering while developing various demonstrators and evaluators.

3.10 Astrolabe Project (Cornell University)

Astrolabe is a new system to automate self-configuration and self-monitoring and to control adaptation. Astrolabe operates by creating a virtual system-wide hierarchical database, which evolves as the underlying information changes [Birman 03, Valetto 05]. Astrolabe is secure and robust under a wide range of failure and attack scenarios, and it imposes low loads even under stress.

4 Analysis and Benefits of Current AC Work

4.1 AC Framework

The AC framework outlined in Section 2 provides methods, algorithms, architectures, technology, and tools to standardize, automate, and simplify myriad system administration tasks. Just a few years ago, the installation—or uninstallation—of an application on a desktop computer required the expertise of an experienced system administrator. Today, most users can install applications using standard install shields with just a handful of mouse clicks. By building self-managing systems using the AC framework, developers can accomplish similar simplifications for many other system administration tasks (e.g., installing, configuring, monitoring, tuning, optimizing, recovering, protecting, and extending).

4.2 Quality Attributes and Architecture Evaluation

The architectural blueprint introduced in Section 2 constitutes a solid foundation for building AC systems. But so far, this blueprint has not come with a software analysis and reasoning framework to facilitate architecture evaluation for self-managing applications. The DEAS Project, mentioned above, proposes to develop such a framework based on ABASs [Klein 99]. When the system evolves, engineers can use this analysis framework to revisit, analyze, and verify certain system properties.

Quality attributes for autonomic architectures should include not only traditional quality criteria such as variability, modifiability, reliability, availability, and security but also autonomicity-specific criteria such as support for dynamic adaptation, dynamic upgrade, detecting anomalous system behavior, how to keep the user informed, sampling rate adjustments in sensors, simulation of expected or predicted behavior, determining the difference between expected and actual behavior, and accountability (i.e., how can users gain trust by monitoring the underlying autonomic system).

Traditionally, for most quality attributes, applying stimuli and observing responses for architectural analysis is basically a thought exercise performed during design and system evolution. However, the DEAS principal investigators envision that many of the autonomicity-specific quality attributes can be analyzed by directly stimulating events and observing responses on the running application, which is already equipped with sensors/monitors and executors/effectors as an autonomic element.

Codifying the relationship between architecture and quality attributes not only enhances the current architecture design but also allows developers to reuse the architecture analysis for other applications. The codification will make the design tradeoffs, which often exist only in

the chief architect's mind, explicit and aid in analyzing the impact of an architecture reconfiguration to meet certain quality attributes during long-term evolution. The fundamental idea is to equip the architecture of autonomic applications with predictability by attaching, at design time, an analysis framework to the architecture of a software system to validate and reassess quality attributes regularly and automatically over long periods of time.

This codification will also aid in the development of standards and curricula materials, which are discussed in more detail in subsequent sections.

4.3 Standards

Many successful solutions in the information technology industry are based on standards. The Internet and World Wide Web are two obvious examples, both of which are built on a host of protocols and content formats standardized by the Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C), respectively.

Before AC technology can be widely adopted, many aspects of its technical foundation have to be standardized. IBM is actively involved in standardizing protocols and interfaces among all interfaces within an autonomic element as well as among elements, as depicted in Figure 5 [Miller 05b].

In March 2005, the Organization for the Advancement of Structured Information Standards (OASIS) standards body approved the Web Services Distributed Management (WSDM) standard, which is potentially a key standard for AC technology. The development of standards for AC and Web services is highly competitive and politically charged.

The Autonomic Computing Forum (ACF) is a European organization that is open and independent. Its mandate is to generate and promote AC technology [Popescu-Zeletin 04].

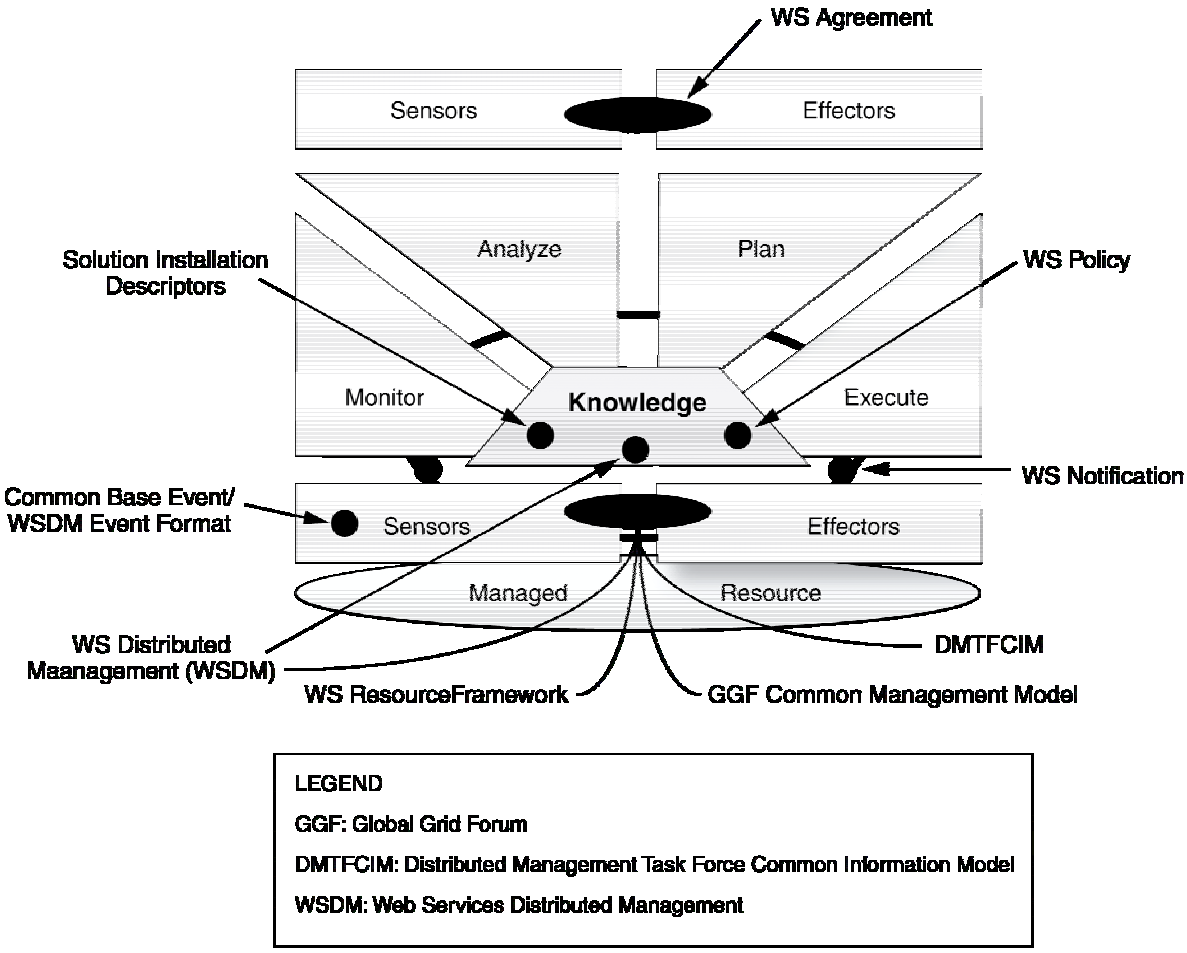

Figure 5: Interface Standards Within an Autonomic Element [Miller 05b]

4.4 Curriculum Development

Control systems are typically featured prominently in electrical and mechanical engineering curricula. Historically, computer science curricula do not require control theory courses. Recently developed software engineering curricula, however, do require control theory [UVIC 03].

Current software architecture courses cover control loops only peripherally. The architecture of autonomic elements is not usually discussed. Note that event-based architectures, which are typically discussed in a computer science curriculum, are different from the architectures for autonomic systems. Courses on self-managed systems should be introduced into all engineering and computing curricula along the lines of real-time and embedded systems courses.

How to build systems from the ground up as self-managed computing systems will likely be a core topic in software engineering and computer science curricula.

5 Connections to the SEI Initiatives

5.1 Basics of SAT and PACC

An important principle of the Software Architecture Technology (SAT) and Predictable Assembly from Certifiable Components (PACC) Initiatives is that quality attributes (such as performance and reliability) exert a dominant influence over the "shape" of the design. For example, performance requirements will determine resource allocation strategies and, hence, the allocation of functionality to units of concurrency, which is an important architectural design decision. In turn, the resource allocation strategy determines the type or types of models that are appropriate for analysis. The extent to which the models can be applied to the architecture is the extent to which the behavior of the architecture can be predicted. This relationship between quality attribute models and the architecture is one reason why quality attributes exert influence over architectural design.

PACC formalizes the relationship through the notion of interpretation. Interpretation is a formal mapping from an architecture (or assembly of components in PACC vernacular) to a quality attribute model. Evaluating the quality attribute model yields quality attribute predictions.

Software architecture technology provides an informal description of the relationship between an architecture and its associated quality attribute models through architectural tactics. "An architectural tactic is a means of satisfying a quality attribute response measure by manipulating some aspect of a quality attribute model through architectural design decisions" [Bachmann 03]. In other words, an architectural tactic uses key parameters of quality attribute models to identify changes in the architecture that will have a positive impact on the achievement of a specific quality attribute requirement.

5.2 SAT and PACC Initiatives in Relation to AC

These two notions of predictive models and controlled architectural change are central to AC, which uses models to implement its control loops. These models must account for the structure of the managed element (ME), how the ME reacts to inputs (relative to policy objectives), and how to adjust the ME to achieve policy objectives. The use of models suggests several areas of potential synergy between the SAT and PACC Initiatives and the various AC initiatives.

- **SAT Initiative:** The goal of the ArchE Project in the SAT Initiative is to create a prototype architectural design assistant. ArchE uses a search strategy that involves automatically creating quality attribute models via interpretation to evaluate the current

design and that uses tactics to generate the next level of the search tree. This search strategy is ArchE's analogue to the autonomic manager control loop, albeit at design time rather than runtime. However, the issues are identical:

- predicting quality attribute behavior, given an architectural design
- determining design alternatives to meet quality attribute objectives
- making tradeoffs between competing quality attribute objectives (possibly through the use of utility functions)

Self-adaptive architectures are a relatively new area of focus for the SAT Initiative that is very synergistic with AC. The SEI has been working on the development of the DiscoTect tool for generating a view of the architecture of a system at runtime [Yan 04a, Yan 04b]. A further aspect of this work is predictably changing architectural patterns at runtime. This work is strongly aligned with the self-adaptation within AC.

- **PACC Initiative:** Some of the key goals of the PACC Initiative are to develop automated interpretation, evaluation, and validation procedures for existing and emerging quality attribute theories—in particular, for the quality attribute's performance (for example, various scheduling theories and real-time queuing theory), safety (for example, the use of model checking to check safety assertions), and security.

  Component certification is another area of focus for PACC that might be relevant to AC (that is, verifying *a priori* that MEs satisfy designated properties).

6 The Potential of AC Technology for DoD Systems

The DoD builds warfighting platforms (ships, aircraft, tanks, command-and-control centers), which contain many systems and are integrated "systems of systems" (SoSs), with a computing infrastructure to support the SoS. In previous acquisitions of such warfighting platforms, each system had its own computing platforms, which were connected point to point or with low-bandwidth local area networks (LANs). Hence, the software-to-hardware allocation was usually static. In some warfighting platforms, a few fixed static allocations were approved, and transitions from one configuration to another were allowed, although this type of platform was rare.

Although applications can execute in any platform, in the case of a ship, the applications must be distributed throughout the ship such that they operate effectively for the multiple mission threads. When the ship's mission is changed, some new applications may have to be installed automatically, and some applications may be removed.

The new development paradigm is to use common computing platforms and to build the software such that "any software component can run anywhere." In many ways, a ship is the largest scale platform built, so the remainder of the discussion concentrates on ships. The problem of allocating software components to hardware components and dealing with multiple simultaneous missions, changing missions, and large-scale computing and communications equipment failure is generally referred to as dynamic resource management (DRM), which overlaps considerably with AC. The basic software elements to be addressed in this domain are listed below:

- Mission (or warfare) threads of applications are sequences of applications to be executed to conduct a mission. Examples include anti-aircraft warfare, anti-submarine warfare, anti-mine warfare, surface warfare, logistics, and ship control. There are often well-defined qualities of service (QoSs) associated with mission threads, often starting from "threat detection" and ending with "engagement of the threat by a weapon." These QoSs often budget latency time between the threat detection and weapon firing, and they allocate time within the thread to each application, to "recovery from a single failure," and to "computing infrastructure." The computing infrastructure time lag usually includes the time to communicate between computing platforms.
- Each mission-thread application may consist of many software components connected in various ways. There is usually a mixture of styles (Publish-Subscribe, Client-Server, Hub-Spoke, Pipe-and-Filter) used to interoperate these components. There are often stringent timing requirements, which can be derived from the QoSs, in executing some paths through these components. There are thousands of software components.

- Different mission threads may include the same software components. In some cases, each mission thread may use its own copy of the component; in other cases, multiple mission threads may share the same component.

The computing infrastructure on a ship consists of hundreds of computing platforms that are organized into zones, each with independent electric supplies, air conditioning, and fire-suppression systems. There may also be multiple control centers, each with many workstations that control the ship's missions and engagements and coordinate its activities with other ships or that operate the ship. In addition, security zones and safety zones usually overlay these other zones.

Some warfighting platforms can conduct a number of mission threads concurrently, and the mission assignments can change dynamically due to warfighting environment changes or large-scale failures of computing resources, such as electric supply failures, air conditioning failures, or battle damage failures. Yet other warfighting platforms can be assigned different missions at different times.

Some of the challenges in this dynamic configuration management that AC can help with are described below.

- Given the current status and capability of the current computing and communication infrastructure resources, how can the software be allocated to hardware "from scratch" to satisfy the QoS requirements for the assigned mission threads under well-defined loading conditions? The QoS requirements may include threshold loading on communication lines; lag times between threat detection and weapons engagement; recovery times from single computing-platform failures; recovery times from software failures; the ability to ensure that a large-scale failure will disable only some mission threads, while others are not affected; and the ability to maintain information at the correct security level during operation. In practice, determining the QoS requirements must be done (offline) for a number of mixes of different missions and environments, each static allocation must be documented, and each configuration must be loaded and tested for conformance to the desired QoS. The results of this analysis can then be used to allocate the software to the hardware.
- Given that the ship is operational and conducting some missions and the current allocation of software to the current computing infrastructure is known, when a large-scale failure occurs, how can it be recognized from a sequence of individual failures, and how can the boundaries of the remaining viable hardware and communications components be recognized? How can software components be reallocated to the remaining working hardware components to provide degraded operation to satisfy a prioritized list of mission capabilities?

The availability of trained ship's personnel to conduct missions from workstations in control centers must also be considered in the above situations.

7 Conclusions

The time is right for the emergence of self-managed or autonomic systems. Over the past decade, we have come to expect that "plug-and-play" for Universal Serial Bus (USB) devices, such as memory sticks and cameras, simply works—even for technophobic users. Today, users demand and crave simplicity in computing solutions.

With the advent of Web and grid service architectures, we begin to expect that an average user can provide Web services with high resiliency and high availability. The goal of building a system that is used by millions of people each day and administered by a half-time person, as articulated by Jim Gray of Microsoft Research, seems attainable with the notion of automatic updates. Thus, autonomic computing seems to be more than just a new middleware technology; in fact, it may be a solid solution for reining in the complexity problem.

Historically, most software systems were not designed as self-managing systems. Retrofitting existing systems with self-management capabilities is a difficult problem. Even if autonomic computing technology is readily available and taught in computer science and engineering curricula, it will take another decade for the proliferation of autonomicity in existing systems.

There are myriad opportunities for applying autonomic computing technology for DoD software. Because the field of autonomic computing draws from many research areas, experts with a variety of backgrounds are required to make this technology palatable for the DoD.

The current work being undertaken in the PACC and SAT Initiatives could be more closely aligned with the autonomic computing field. The work in predictive models, controlled architecture change, and self-adapting systems has a strong synergy with the work in autonomic computing. The SEI is ideally positioned to develop autonomic computing technology further and ensure the development and operation of self-managed systems with predictable and improved cost, schedule, and quality.

Bibliography

Although this autonomic computing bibliography is incomplete, the documents listed give a good indication of how quickly the field has grown since 2001. The documents tagged with an asterisk are good starting points for newcomers to the field of autonomic computing.

URLs are valid as of the publication date of this document.

[Abbondanzio 03] Abbondanzio, A.; Aridor, Y.; Biran, O.; Fong, L. L.; Goldszmidt, G. S.; Harper, R. E.; Krishnakumar, S. M.; Pruett, G.; &Yassur, B.A. "Management of Application Complexes in Multitier Clustered Systems," *IBM Systems Journal 42,* 1 (2003): 189-195. http://www.research.ibm.com/journal/sj/421/forum.html.

[Adi 03] Adi, Asaf; Biger, Ayelet; Botzer, David; Etzion, Opher; & Sommer, Ziva. "Context Awareness in Amit," 160-167. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003*). Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Agarwal 03] Agarwal, M.; Bhat, V.; Liu, H.; Matossian, V.; Putty, V.; Schmidt, C.; Zhang, G.; Zhen, L.; Parashar, M.; Rutgers, Khargharia, B.; & Hariri, S. "AutoMate: Enabling Autonomic Applications on the Grid," 48-59. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Aggarwal 04] Aggarwal, Gagan; Datar, Mayur; Mishra, Nina; & Motwani, Rajeev. "On Identifying Stable Ways to Configure Systems," 148-153. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Aiber 04] Aiber, Sarel; Gilat, Dagan; Landau, Ariel; Razinkov, Natalia; Sela, Aviad; & Wasserkrug, Segev. "Autonomic Self-Optimization According to Business Objectives," 206-213. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Ananthanarayanan 05] Ananthanarayanan, Rema; Mohania, Mukesh; & Gupta, Ajay. "Management of Conflicting Obligations in Self-Protecting Policy-Based Systems," 274-285. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Anthony 04] Anthony, Richard John. "Emergence: A Paradigm for Robust and Scalable Distributed Applications," 132-139. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Appavoo 03] Appavoo, J.; Hui, K.; Soules, C. A. N.; Wisniewski, R. W.; Da Silva, D. M.; Krieger, O.; Auslander, M. A.; Edelsohn, D. J.; Gamsa, B.; Ganger, G. R.; McKenney, P.; Ostrowski, M.; Rosenburg, B.; Stumm, M.; & Xenidis, J. "Enabling Autonomic Behavior in Systems Software with Hot Swapping." *IBM Systems Journal 42,* 1 (2003): 60-76. http://www.research.ibm.com/journal/sj/421/appavoo.html.

[Ari 04] Ari, Ismail; Gottwals, Melanie; & Henze, Dick. "SANBoost: Automated SAN-Level Caching in Storage Area Network," 164-171. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Arshad 05] Arshad, N.; Heimbigner, D.; & Wolf, A. "Dealing with Failures During Failure Recovery of Distributed Systems," 9-14. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Bachmann 03] Bachmann, Felix; Bass, Len; & Klein, Mark. *Deriving Architectural Tactics: A Step Toward Methodical Architectural Design* (CMU/SEI-2003-TR-004, ADA413644). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003. http://www.sei.cmu.edu /publications/documents/03.reports/03tr004.html.

[Balasubramaniam 05] Balasubramaniam, D.; Morrison, R.; Kirby, G.; Mickan, K.; Warboys, B.; Robertson, I.; Snowdon, B.; Greenwood, M.; & Seet, W. "A Software Architecture Approach for Structuring Autonomic Systems," 59-65. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005).* St. Louis, Missouri, May 21, 2005. New York, NY: ACM Press, 2005.

[Bantz 03] Bantz, D. F.; Bisdikian, C.; Challener, D.; Karidis, J. P.; Mastrianni, S.; Mohindra, A.; Shea, D. G.; & Vanover, M. "Autonomic Personal Computing." *IBM Systems Journal 42,* 1 (2003): 165-176. http://www.research.ibm.com/journal/sj/421/bantz.html.

[Barrett 04] Barrett, Rob; Maglio, Paul P.; Kandogan, Eser; & Bailey, John. "Usable Autonomic Computing Systems: The Administrator's Perspective," 18-26. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Bartlet 05] Bartlet, C.; Fischer, T.; Niebuhr, D.; Rausch, A.; Seidl, F.; & Trapp, M. "Dynamic Integration of Heterogeneous Mobile Devices," 92-98. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005).* St. Louis, Missouri, May 21, 2005. New York, NY: ACM Press, 2005.

[Bennani 04] Bennani, Mohamed N. & Menasce, Daniel A. "Assessing the Robustness of Self-Managing Computer Systems Under Highly Variable Workloads," 62-69. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Birman 03] Birman, Kenneth P.; van Renesse, Robbert; Kaufman, James; & Vogels, Werner. "Navigating in the Storm: Using Astrolabe for Distributed Self-Configuration, Monitoring, and Adaptation," 4-13. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Biyani 05] Biyani, K. & Kulkarni, S. "Building Component Families to Support Adaptation," 125-131. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Bodíc 05] Bodíc, Peter; Friedman, Greg; Biewald, Lukas; Levine, Helen; Candea, George; Patel, Kayur; Tolle, Gilman; Hui, Jon; Fox, Armando; Jordan, Michael I.; & Patterson, David. "Combining Visualization and Statistical Analysis to Improve Operator Confidence and Efficiency for Failure Detection and Localization," 89-100. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Bohra 04] Bohra, Aniruddha; Neamtiu, Iulian; Gallard, Pascal; Sultan, Florin; & Iftode, Liviu. "Remote Repair of Operating System State Using Backdoors," 256-263. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Bonino 04] Bonino, Dario; Bosca, Alessio; & Corno, Fulvio. "An Agent-Based Autonomic Semantic Platform," 189-196. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Bradshaw 04] Bradshaw, Jeffrey. "Toward Semantically-Rich Policy Representation Issues and Applications." *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com/asr/chiacs/presentations/bradshaw.pdf.

[Brennan 04] Brennan, Susan. "Adaptive Spoken Dialogue with Human and Computer Partners: Implications for Autonomic Systems." *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com/asr/chiacs/bios.shtml?brennan.

[Brodie 05] Brodie, Mark; Ma, Sheng; Lohman, Guy; Mignet, Laurent; Modani, Natwar; Wilding, Mark; Champlin, Jon; & Sohn, Peter. "Quickly Finding Known Software Problems via Automated Symptom Matching," 101-110. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Candea 03] Candea, George; Kiciman, Emre; Zhang, Steve; Keyani, Pedram; & Fox, Armando. "JAGR: An Autonomous Self-Recovering Application Server," 168-177. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003).* Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Canfora 05] Canfora, G.; Corte, P.; De Nigro, A.; Desideri, D.; Di Penta, M.; Esposito, R.; Falanga, A.; Renna, G.; Scognamiglio, R.; Torelli, F.; & Villani. M. "The C-Cube Framework: Developing Autonomic Applications Through Web Services," 106-111. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005).* St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Chakravarti 04] Chakravarti, Arjav J.; Baumgartner, Gerald; & Lauria, Mario. "The Organic Grid: Self-Organizing Computation on a Peer-to-Peer Network," 96-103. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Chang 04] Chang, Yu-Han; Ho, Tracey; & Kaelbling, Leslie Pack. "Mobilized Ad-Hoc Networks: A Reinforcement Learning Approach," 240-247. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Chen 04] Chen, Mike; Zheng, Alice X.; Jordan, Michael I.; & Brewer, Eric. "Failure Diagnosis Using Decision Trees," 36-43. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Chen 05] Chen, Ying. "Information Valuation for Information Lifecycle Management," 135-146. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Cheng 01] Cheng, Shang-Wen & Garlan, David. "Mapping Architectural Concepts to UML-RT," 172-179. *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2001)*. Las Vegas, NV, June 25-28, 2001. Las Vegas, NV: Computer Science Research, Education, and Applications Press, 2001.

[Cheng 02a] Cheng, Shang-Wen; Garlan, David; Schmerl, Bradley; Sousa, João Pedro; Spitznagel, Bridget; & Steenkiste, Peter. "Using Architectural Style as a Basis for Self-Repair," 45-59. *Software Architecture: System Design, Development, and Maintenance. Proceedings of the Third Working IEEE/IFIP Conference on Software Architecture.* Montreal, Quebec, Canada, August 25-30, 2002. Norwell, Ma: Kluwer Academic Publishers, 2002.

[Cheng 02b] Cheng, Shang-Wen; Garlan, David; Schmerl, Bradley; Sousa, João Pedro; Spitznagel, Bridget; Steenkiste, Peter; & Hu, Ningning. "Software Architecture-Based Adaptation for Pervasive Systems," 67-82. *Proceedings of the International Conference on Architecture of Computing Systems (ARCS 2002): Trends in Network and Pervasive Computing*. Karlsruhe, Germany, April 8-11, 2002. Berlin, Germany: Springer-Verlag, 2002 (Lecture Notes in Computer Science, Volume 2299).

[Cheng 02c] Cheng, Shang-Wen; Garlan, David; Schmerl, Bradley; Steenkiste, Peter; & Hu, Ningning. "Software Architecture-Based Adaptation for Grid Computing," 389-398. *Proceedings of the 11th IEEE Conference on High Performance Distributed Computing (HPDC 2002)*. Edinburgh, Scotland, July 23-26, 2002. Piscataway, NJ: IEEE Computer Society, 2002.

[Cheng 04a] Cheng, Shang-Wen; Huang, An-Cheng; Garlan, David; Schmerl, Bradley; & Steenkiste, Peter. "An Architecture for Coordinating Multiple Self-Management Systems," 243-252. *Proceedings of the Fourth Working IEEE/IFIP Conference on Software Architectures*. Oslo, Norway, June 11-14, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Cheng 04b] Cheng, Shang-Wen; Huang, An-Cheng; Garlan, David; Schmerl, Bradley; & Steenkiste, Peter. "Rainbow: Architecture-Based Self Adaptation with Reusable Infrastructure." *IEEE Computer 37,* 10 (October 2004): 46-54.

[Chess 03] Chess, D. M.; Palmer, C. C.; & White, S. R. "Security in an Autonomic Computing Environment." *IBM Systems Journal 42*, 1 (2003): 107-118. http://www.research.ibm.com/journal/sj/421/chess.html.

[Chess 04] Chess, David M.; Segal, Alla; Whalley, Ian; & White, Steve R. "Unity: Experiences with a Prototype Autonomic Computing System," 140-147. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Chess 05] Chess, David M.; Pacifici, Giovanni; Spreitzer, Mike; Steinder, Malgorzata; & Tantawi, Asser. "Experience with Collaborating Managers: Node Group Manager and Provisioning Manager," 39-50. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Cornwell 04] Cornwell, Mark; Just, James; Kagal, Lalana; Finin, Tim; & Huhns, Mike. "Autonomy in Open Distributed Systems." *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com/asr/chiacs/presentations/finin.pdf.

[De Wolf 05] De Wolf, Tom; Samaey, Giovanni; Holvoet, Tom; & Roose, Dirk. "Decentralised Autonomic Computing: Analysing Self-Organising Emergent Behaviour Using Advanced Numerical Methods," 52-63. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Deen 03] Deen, Glenn; Lehman, Toby; & Kaufman, James. "The Almaden OptimalGrid Project," 14-21. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Denaro 05] Denaro, G.; Pezze, M.; & Tosi, D. "Adaptive Integration of Third-Party Web Services," 112-117. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Diaconescu 04] Diaconescu, Ada; Mos, Adrian; & Murphy, John. "Automatic Performance Management in Component Based Software Systems," 214-221. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Diao 03] Diao, Y.; Hellerstein, J. L.; Parekh, S.; & Bigus, J. P. "Managing Web Server Performance with AutoTune Agents." *IBM Systems Journal 42*, 1 (2003): 136-149. http://www.research.ibm.com/journal/sj/421/diao.html.

[Dunagan 04] Dunagan, John; Roussev, Roussi; Daniels, Brad; Johnson, Aaron; Verbowski, Chad; &Wang, Yi-Min. "Towards a Self-Managing Software Patching Process Using Black-Box Persistent-State Manifests," 106-113. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[EDS 05] Electronic Data Systems Corporation (EDS). *Agile Enterprise.* http://www.eds.com/services /agileenterprise/overview.aspx (2005).

[Femal 05] Femal, Mark E. & Freeh, Vincent W. "Boosting Data Center Performance Through Non-Uniform Power Allocation," 250-261. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Finden-Browne 04] Finden-Browne, Chris. "An Approach to Estimating the Productivity Impact of Autonomic Computing Improvement Initiatives." *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com /asr/chiacs/presentations/finden-browne.pdf.

[Fleming 05] Fleming, S.; Cheng, B.; Stirewalt, K.; & McKinley, P. "An Approach to Implementing Dynamic Adaptation in C++," 118-124. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Fukuda 03] Fukuda, Munehiro; Tanaka, Yuichiro; Suzuki, Naoya; Bic, Lubomir F.; & Kobayashi, Shinya. "A Mobile-Agent-Based PC Grid," 142-150. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

***[Ganek 03]** Ganek, A. & Corbi, T. "The Dawning of the Autonomic Computing Era." *IBM Systems Journal 42*, 1 (2003): 5-18. http://www.research.ibm.com/journal/sj/421/ganek.pdf.

[Ganek 04] Ganek, Alan. "Managing the Human-Computer Partnership with Autonomic Computing." *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com/asr/chiacs /presentations/ganek.pdf.

[Garlan 01] Garlan, David; Schmerl, Bradley; & Chang, Jichuan. "Using Gauges for Architecture-Based Monitoring and Adaptation." *Proceedings of the Working Conference on Complex and Dynamic Systems Architecture*. Brisbane, Australia, December 12-14, 2001. Brisbane, Australia: Distributed Systems Technology Centre (DSTC), 2001. http://www.cs.cmu.edu/afs/cs/project/able/ftp /cdsa01/cdsa.pdf.

[Garlan 02a] Garlan, David & Schmerl, Bradley. "Model-Based Adaptation for Self-Healing Systems," 27-32. *Proceedings of the ACM SIGSOFT Workshop on Self-Healing Systems (WOSS 2002)*. Charleston, SC, November 18-19, 2002. New York, NY: Association for Computing Machinery, 2002.

[Garlan 02b] Garlan, David; Kompanek, Andrew J.; & Cheng, Shang-Wen. "Reconciling the Needs of Architectural Description with Object-Modeling Notations." *Science of Computer Programming 44*, 1 (July 2002): 23-49. http://www.cs.cmu.edu/~able/publications/uml01/.

[Garlan 03] Garlan, David; Cheng, Shang-Wen; & Schmerl, Bradley. "Increasing System Dependability Through Architecture-Based Self-Repair," 61-89. *Architecting Dependable Systems*. (Edited by R. de Lemos, C. Gacek, & A. Romanovsky). Berlin, Germany: Springer-Verlag, 2003 (Lecture Notes in Computer Science, Volume 2677).

[Garlan 04] Garlan, David & Schmerl, Bradley. "Using Architectural Models at Runtime: Research Challenges," 200-205. *Proceedings of the First European Workshop on Software Architecture (EWSA 2004)*. St. Andrews, Scotland, May 21-22, 2004. Berlin, Germany: Springer-Verlag, 2004.

[Gelenbe 04] Gelenbe, Erol; Gellman, Michael; Lent, Ricardo; Liu, Peixiang; & Su, Pu. "Autonomous Smart Routing for Network QoS," 232-239. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

***[Gibbs 02]** Gibbs, W. Wayt. "Autonomic Computing." *Scientific American* (May 2002). http://www.sciam.com/article.cfm?articleID=000B0152-8C15-1CDA-B4A8809EC588EEDF.

[Globus 05] The Globus Alliance. *Towards The Open Grid Services Architecture.* http://www.globus.org/ogsa/ (2005).

[Goteti 03] Goteti, Srikanth & Subhlok, Jaspal. "Communication Pattern Based Node Selection for Shared Networks," 69-76. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Greenwood 04] Greenwood, P. & Blair, L. "Using Dynamic Aspect-Oriented Programming to Implement an Autonomic System," 76-88. *Proceedings of the 2004 Dynamic Aspects Workshop (DAW 2004)*. Lancaster, England, March 23, 2004. Mountainview, CA: Research Institute for Advanced Computer Science (RIACS), 2004.

[Griffith 05] Griffith, R. & Kaiser, G. "Manipulating Managed Execution Runtimes to Support Self-Healing," 2-8. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005*). St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Gurguis 05] Gurguis, S. & Zeid, A. "Towards Autonomic Web Services: Applying Self-Healing to Web Services," 22-26. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Haas 03] Haas, R.; Droz, P.; & Stiller, B. "Autonomic Service Deployment in Networks." *IBM Systems Journal 42*, 1 (2003): 150-164. http://www.research.ibm.com /journal/sj/421/haas.html.

[Hariri 03] Hariri, Salim. *AUTONOMIA: An Autonomic Computing Environment*. http://www.ece.arizona.edu /~hpdc/projects/AUTONOMIA/ (2003).

[He 03] He, Yu & Raghavendr, Cauligi S. "A Programmable Routing Framework for Autonomic Sensor Networks," 60-68. Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003). Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Heinis 05] Heinis, Thomas; Pautasso, Cesare; & Alonso, Gustavo. "Design and Evaluation of an Autonomic Workflow Engine," 27-38. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

***[Herger 03]** Herger, L.; Iwano, K.; Pattnaik, P.; Davis, A.; & Ritsko, J., eds. "Special Issue on Autonomic Computing." *IBM Systems Journal 42*, 1 (2003). http://www.research.ibm.com/journal/sj42-1.html.

[Hidrobo 03] Hidrobo, Francisco & Cortes, Toni. “Towards a Zero-Knowledge Model for Disk Drives,” 122-130. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Hitachi 03] Hitachi Ltd. *Creating the Value of Harmonious Computing*. http://www.harmonious.jp/products/harmonious/center/gl /main/information/concept/index3.html (2003).

[HP 03] Hewlett-Packard Development Company. *Adaptive Enterprise*. http://www.hp.com/products1/promos /adaptive_enterprise/us/adaptive_enterprise.html (2003).

***[IBM 02]** IBM Research. *Autonomic Computing.* http://www.research.ibm.com/autonomic/ (2002).

***[IBM 03]** IBM Corporation. *An Architectural Blueprint for Autonomic Computing.* http://www-03.ibm.com/autonomic/pdfs /ACwpFinal.pdf (April 2003).

[IBM 05a] IBM Corporation. *IBM Paves the Way for Mainstream Adoption of Autonomic Computing* (Press Release). http://www-03.ibm.com/autonomic/press_041905.shtml (April 2005).

[IBM 05b] IBM Corporation. *IBM Tivoli Risk Manager*. http://www-306.ibm.com/software/tivoli/products/risk-mgr/ (2005).

***[IBM 05c]** IBM Corporation. *Autonomic Computing Toolkit.* http://www-128.ibm.com/developerworks/autonomic /overview.html **(**2005).

[Jann 03] Jann, J.; Browning, L. M.; & Burugula, R. S. “Dynamic Reconfiguration: Basic Building Blocks for Autonomic Computing on IBM pSeries Servers.” *IBM Systems Journal 42*, 1 (2003): 29-37. http://www.research.ibm.com /journal/sj/421/jann.html.

[Jarvis 03] Jarvis, Stephen A.; Spooner, Daniel P.; Lim, Helene N.; Keung, Choi; Dyson, Justin R. D.; Zhao, Lei; & Nudd, Graham R. "Performance-Based Middleware Services for Grid Computing," 151-159. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003).* Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Jiang 05] Jiang, Guofei; Chen, Haifeng; Ungureanu, Cristian; & Yoshihira, Kenji. "Multi-Resolution Abnormal Trace Detection Using Varied-Length N-grams and Automata," 111-122. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005).* Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Julie 04] Julie, M. & Markus, C. H. "Evaluation Issues in Autonomic Computing," 597-608. *Proceedings of the Grid and Cooperative Computing - GCC 2004 Workshops*. Wuhan, China, October 21-24, 2004. Berlin, Germany: Springer-Verlag, 2004 (Lecture Notes in Computer Science, Volume 3252).

[Kaiser 03] Kaiser, Gail; Parekh, Janak; Gross, Philip; & Valetto, Giuseppe. "Kinesthetics eXtreme: An External Infrastructure for Monitoring Distributed Legacy Systems," 22-31. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Kaminski 05] Kaminski, P.; Agrawal, P.; Kienle, H.; & Müller, H. "<username>, I Need You! Initiative and Interaction in Autonomic Systems," 73-76. *Proceedings of the Design and Evolution of Autonomic Application Software (DEAS 2005*). St. Louis, MO, May 21, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Kandasamy 04] Kandasamy, Nagarajan; Abdelwahed, Sherif; & Hayes, John P. "Self-Optimization in Computer Systems via On-Line Control: Application to Power Management," 54-61. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Kandogan 03] Kandogan, E. & Maglio, P. "Why Don't You Trust Me Anymore? Or the Role of Trust in Troubleshooting Activities of System Administrators." *Conference on Human Factors in Computing Systems (CHI 2003).* Fort Lauderdale, FL, April 7, 2003. http://darkseid.cs.berkeley.edu/~mikeychen/chi2003/papers.shtml.

[Kandogan 04] Kandogan, Eser. "Policy-Based System Management: Collaboration and Cooperation." *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com/asr/chiacs/presentations/kandogan.pdf.

[Karlsson 05] Karlsson, Magnus & Covell, Michele. "Dynamic Black-Box Performance Model Estimation for Self-Tuning Regulators," 172-182. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

***[Kephart 03]** Kephart, J. O. & Chess, D. M. "The Vision of Autonomic Computing." *IEEE Computer 36*, 1 (January 2003): 41-50.

***[Kephart 04]** Kephart, Jeffrey O. & Walsh, William E. "An Artificial Intelligence Perspective on Autonomic Computing Policies," 3-12. *Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (Policy 2004)*. Yorktown Heights, NY, June 7-9, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Kephart 05a] Kephart, J. O. *Information Economies*. http://www.research.ibm.com/infoecon/ (2005).

[Kephart 05b] Kephart, J. O. “Research Challenges of Autonomic Computing,” 15-22. *Proceedings of the International Conference on Software Engineering (ICSE)*. St. Louis, MO, May 15-21, 2005. New York, NY: Association for Computing Machinery, 2005.

[Kim 05] Kim, M.; Jeong, J.; & Park, S. “From Product Lines to Self-Managed Systems: An Architecture-Based Runtime Reconfiguration Framework,” 66-72. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Kistler-Glendon 04] Kistler-Glendon, Keith. “Beginning the Autonomic Journey: A Review & Lessons Learned from an Autonomic Computing Readiness Assessment @ A Major US Telecomm.” *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com /asr/chiacs/presentations/kistler-glendon.pdf.

[Klein 99] Klein, Mark & Kazman, Rick. *Attribute-Based Architectural Styles* (CMU/SEI-99-TR-022, ADA371802). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1999. http://www.sei.cmu.edu/publications /documents/99.reports/99tr022/99tr022abstract.html.

***[Kluth 04]** Kluth, A. “Survey: Information Technology. Make It Simple.” *The Economist.* http://www.economist.com/surveys /showsurvey.cfm?issue=20041030 (Oct. 28, 2004).

[Konstantinou 03] Konstantinou, Alexander V. & Yemini, Yechiam. “Programming Systems for Autonomy,” 186-195. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Kumar 05] Kumar, Vibhore; Cooper, Brian F.; & Schwan, Karsten. "Distributed Stream Management Using Utility-Driven Self-Adaptive Middleware," 3-14. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Kurmas 04] Kurmas, Zachary & Keeton, Kimberly. "Using the Distiller to Direct the Development of Self-Configuration Software," 172-179. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Kwan 03] Kwan, Eva; Lightstone, Sam; Schiefer, Berni; Storm, Adam; & Wu, Leanne. "Automatic Database Configuration for DB2 Universal Database: Compressing Years of Performance Expertise into Seconds of Execution," 620-629. *Proceedings of Database Systems for Business, Technology, and the Web Conference (BTW 2003)*. Leipzig, Germany, February 26-28, 2003. Leipzig, Germany: Gesellschaft für Informatik, 2003. http://doesen0.informatik.uni-leipzig.de/proceedings/proceedings.en.shtml.

[Kycyman 04] Kycyman, Emre & Wang, Yi-Min. "Discovering Correctness Constraints for Self-Management of System Configuration," 28-35. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Lanfranchi 03] Lanfranchi, G.; Della Peruta, P.; Perrone, A.; & Calvanese, D. "Toward a New Landscape of Systems Management in an Autonomic Computing Environment." *IBM Systems Journal 42*, 1 (2003): 119-128. http://www.research.ibm.com/journal/sj/421/lanfranchi.html.

[Lapouchnian 05] Lapouchnian, A.; Liaskos, S.; Mylopoulos, J.; & Yu, Y. "Towards Requirements-Driven Autonomic Systems Design," 45-51. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005*). St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Lewis 04] Lewis, Dave; O'Donnell, Tony; Feeney, Kevin; Brady, Aoife; & Wade, Vincent. "Managing User-Centric Adaptive Services for Pervasive Computing," 248-255. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Ling 03] Ling, Benjamin C. & Fox, Armando. "A Self-Tuning, Self-Protecting, Self-Healing Session State Management Layer," 131-139. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003*). Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Litoiu 05] Litoiu, M.; Woodside, M.; & Zheng, T. "Hierarchical Model-Based Autonomic Control of Software Systems," 27-33. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Liu 04] Liu, Hua; Parashar, Manish; & Hariri, Salim. "A Component-Based Programming Model for Autonomic Applications," 10-17. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Lotlika 05] Lotlika, Rohit R. M.; Vatsavai, Ranga R.; Mohania, Mukesh; & Chakravarthy, Sharma. "Policy Schedule Advisor for Performance Management," 183-192. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Maglio 05] Maglio, Paul P.; Campbell, Christopher S.; & Kandogan, Eser. "Learning Automation Policies for Pervasive Computing Environments," 193-203. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Mahabhashyam 04] Mahabhashyam, Sai Rajesh & Gautam, Natarajan M. "Dynamic Resource Allocation of Shared Data Centers Supporting Multiclass Requests," 222-229. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Markl 03] Markl, V.; Lohman, G. M.; & Raman, V. "LEO: An Autonomic Query Optimizer for DB2." *IBM Systems Journal 42*, 1 (2003): 98-106. http://www.research.ibm.com/journal/sj/421/markl.html.

***[McKinley 04]** McKinley, Philip K.; Sadjadi, Seyed Masoud; Kasten, Eric P.; & Cheng, Betty C. "Composing Adaptive Software." *IEEE Computer 37*, 7 (July 2004): 56-64.

[Merideth 05] Merideth, M. & Narasimhan, P. "Retrofitting Networked Applications to Add Autonomic Reconfiguration," 38-44. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Mesnier 04] Mesnier, Michael; Thereska, Eno; Ganger, Gregory R.; Ellard, Daniel; & Seltzer, Margo. "File Classification in Self-* Storage Systems," 44-51. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

***[Microsoft 05]** Microsoft Corporation. *Dynamic Systems Initiative*. http://www.microsoft.com/dsi/ (2005).

[Mikik-Racik 02] Mikik-Racik, M.; Mehta, N.; & Medvidovic, N. "Architectural Style Requirements for Self-Healing Systems," 49-54. *Proceedings of the First Workshop on Self-Healing Systems.* November 18-19, 2002. New York, NY: Association for Computing Machinery, 2002.

[Mikik-Racik 04] Mikik-Racik, Marija & Medvidovic, Nenad. "Support for Disconnected Operation via Architectural Self-Reconfiguration," 114-121. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Miller 05a] Miller, Brent. *The Autonomic Computing Edge: Autonomic Computing Heats Up in Japan.* http://www-128.ibm.com /developerworks/library/ac-edge/ (February 2005).

[Miller 05b] Miller, Brent. *The Autonomic Computing Edge: The "Standard" Way of Autonomic Computing*. http://www-128.ibm.com/developerworks/autonomic/library /ac-edge2/ (March 2005).

[Miller 05c] Miller, Brent. *The Autonomic Computing Edge: Autonomic Computing Heats Up in Academia*. http://www-128.ibm.com/developerworks/autonomic /library/ac-edge3/ (May 2005).

[Miller 05d] Miller, Brent. *The Autonomic Computing Edge: Can You CHOP Up Autonomic Computing?* http://www-128.ibm.com/developerworks /autonomic/library/ac-edge4/ (June 2005).

[Miller 05e] Miller, Brent. *The Autonomic Computing Edge: Keeping in Touch with Touchpoints*. http://www-128.ibm.com /developerworks/autonomic/library/ac-edge5/ (August 2005).

[Miller 05f] Miller, Brent. *The Autonomic Computing Edge: The Role of Knowledge in Autonomic Systems*. http://www-128.ibm.com/developerworks/autonomic/library /ac-edge6/ (September 2005).

[Minsky 03] Minsky, Naftaly H. "On Conditions for Self-Healing in Distributed Software Systems," 86-92. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Mischke 03] Mischke, Jan & Stiller, Burkhard. "Semantic Software Engineering Approaches for Automatic Service Lookup and Integration," 112-121. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Munawar 05] Munawar, M. & Ward, P. "Better Performance or Better Manageability?" 34-37. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

***[Murch 04]** Murch, R. *Autonomic Computing.* Upper Saddle River, NJ: Prentice Hall, 2004 (ISBN 0-13-144025X).

[Narain 03] Narain, Sanjai; Cheng, Thanh; Coan, Brian; Kau, Vikram; Parmeswaran, Kirthika; & and Stephens, William. "Building Autonomic Systems via Configuration," 77-85. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Norman 03] Norman, D. A.; Ortony, A.; & Russell, D. M. "Affect and Machine Design: Lessons for the Development of Autonomous Machines." *IBM Systems Journal 42*, 1 (2003): 38-44. http://www.research.ibm.com/journal/sj /421/norman.html.

[Norris 04] Norris, James; Coleman, Keith; Fox, Armando; & Candea, George. "OnCall: Defeating Spikes with a Free-Market Application Cluster," 198-205. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Ogel 03] Ogel, F.; Patarin, S.; Piumarta, I.; & Folliot, B. "C/SPAN: A Self-Adapting Web Proxy Cache," 178-185. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Paar 03] Paar, Alexander & Tichy, Walter F. "Semantic Software Engineering Approaches for Automatic Service Lookup and Integration," 103-111. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Papazoglou 03] Papazoglou, M. P. & Georgakopoulos, D. “Service-Oriented Computing: Introduction.” *Communications of the ACM 46*, 10 (October 2003): 24-28.

[Parashar 03a] Parashar, Manish. *Project AutoMate*. http://automate.rutgers.edu/ (2003).

[Parashar 03b] Parashar, M. & Hariri, S. “Autonomic Computing Tutorial. s.” *ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 03)*. Tunisia, July 14-18, 2003. http://automate.rutgers.edu/tutorials/AICCSA2003.htm.

[Parashar 05] Parashar, M. & Hariri, S. “Autonomic Computing: An Overview,” 257. *Unconventional Programming Paradigms (UPP 2004): Revised Selected and Invited Papers* (Edited by: J.-P. Banâtre et al.). Le Mont Saint-Michel, France, September 15-17, 2004. Berlin, Germany: Springer-Verlag, 2005 (Lecture Notes in Computer Science, Vol. 3566).

[Patterson 02] Patterson, David. *The Berkeley/Stanford Recovery-Oriented Computing (ROC) Project*. http://roc.cs.berkeley.edu/ (March 2002).

[Poellabauer 03] Poellabauer, Christian; Schwan, Karsten; Agarwala, Sandip; Gavrilovska, Ada; & Eisenhauer, Greg. “Service Morphing: Integrated System- and Application-Level Service,” 93-102. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Popescu-Zeletin 04] Popescu-Zeletin, Radu. *Autonomic Computing Forum (ACF)*. http://www.autonomic-communication-forum.org/im /presentations/01-ACF-IM-Welcome_RPZ.pdf (2004).

[Ramakrishna 03] Ramakrishna, V.; Robinson, Max; Eustince, Kevin; & Reiher, Peter. “An Active Self-Optimizing Multiplayer Gaming Architecture,” 32-41. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003*). Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Ranganathan 04] Ranganathan, Anand & Campbell, Roy H. "Autonomic Pervasive Computing Based on Planning," 80-87. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Roblee 05] Roblee, Christopher; Berk, Vincent; & Cybenko, George. "Implementing Large-Scale Autonomic Server Monitoring Using Process Query Systems," 123-133. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Russell 03a] Russell, L. W.; Morgan, S. P.; & Chron, E. G. "Clockwork: A New Movement in Autonomic Systems." *IBM Systems Journal 42*, 1 (2003): 77-84. http://www.research.ibm.com/journal/sj/421/russell.html.

[Russell 03b] Russell, D. M.; Maglio, P. P.; Dordick, R.; & Neti, C. "Dealing with Ghosts: Managing the User Experience of Autonomic Computing." *IBM Systems Journal 42*, 1 (2003): 177-188. http://www.research.ibm.com/journal/sj /421/maglio.html.

[Sadjadi 04] Sadjadi, S. M. & McKinley, P. K. "Transparent Self-Optimization in Existing CORBA Applications," 88-95. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Sadjadi 05a] Sadjadi, S. Masoud & McKinley, Philip K. "Using Transparent Shaping and Web Services to Support Self-Management of Composite Systems," 76-87. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005*). Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Sadjadi 05b] Sadjadi, M.; McKinley, P. K.; & Cheng, B. "Transparent Shaping of Existing Software to Support Pervasive and Autonomic Computing," 99-105. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Salehie 05] Salehie, M. & Tahvildari, L. "Autonomic Computing: Emerging Trends and Open Problems," 82-88. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Schmerl 02] Schmerl, Bradley & Garlan, David. "Exploiting Architectural Design Knowledge to Support Self-Repairing Systems," 241-248. *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering*. Ischia, Italy, July 15-19, 2002. New York, NY: ACM Press, 2002.

[Sheridan 04] Sheridan, Thomas B. "Human Supervisory Control of Dynamic Systems." *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com/asr /chiacs/presentations/sheridan.pdf.

[Shi 05] Shi, Weidong; Lee, Hsien-Hsin S.; Gu, Guofei; Falk, Laura; Mudge, Trevor N.; & Ghosh, Mrinmoy. "An Intrusion-Tolerant and Self-Recoverable Network Service System Using A Security Enhanced Chip Multiprocessor," 263-273. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Shin 05] Shin, Michael. "Connector-Based Self-Healing Mechanism for Components of a System," 15-21. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Simmons 05] Simmons, B. & Lutfiyya, H. "Policies, Grids and Autonomic Computing," 77-81. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Smith 05] Smith, D.; Carney, D.; & Morris, E. "Interoperability Issues Affecting Autonomic Computing," 89-91. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Spitznagel 01] Spitznagel, Bridget & Garlan, David. "A Compositional Approach for Constructing Connectors," 148-157. *Proceedings of the Working IEEE/IFIP Conference on Software Architecture (WICSA 2001)*. Amsterdam, The Netherlands, August 28-31, 2001. Los Alamitos, CA: IEEE Computer Society, 2001.

[Spitznagel 03] Spitznagel, Bridget & Garlan, David. "A Compositional Formalization of Connector Wrappers," 374-384. *Proceedings of the ACM/IEEE 25th International Conference on Software Engineering (ICSE 2003)*. Portland, Oregon, May 3-10, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Srivastava 04] Srivastava, Biplav; Bigus, Joseph P.; & Schlosnagle, Donald A. "Bringing Planning to Autonomic Applications with ABLE," 154-161. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Strickland 05] Strickland, Jonathan W.; Freeh, Vincent W.; Ma, Xiaosong; & Vazhkudai, Sudharshan S. "Governor: Autonomic Throttling for Aggressive Idle Resource Scavenging," 64-75. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Sun 03] Sun Microsystems. *Sun N1 Grid Engine 6*. http://www.sun.com/software/gridware/index.xml (2003).

[Sventek 05] Sventek, Joe; Sloman, Morris; Dulay, Naranker; & Lupu, Emil. *Autonomic Management of Ubiquitous Systems for E-Health (AMUSE*). http://www.dcs.gla.ac.uk /amuse/presentations.htm (2005).

[Tosi 04] Tosi, D. *Research Perspectives in Self-Healing Systems* (Technical report number LTA: 2004: 06). Milano, Italy: Laboratorio di Test e Analist del Software (LTA), 2004. http://www.lta.disco.unimib.it/doc/ei/pdf/lta.2004.06.pdf.

[Trumler 03] Trumler, Wolfgang; Bagci, Faruk; & Petzold, Jan. "Smart Doorplates - Toward an Autonomic Computing System," 42-47. *Proceedings of the Autonomic Computing Workshop Fifth Annual International Workshop on Active Middleware Services (AMS 2003)*. Seattle, WA, June 25, 2003. Los Alamitos, CA: IEEE Computer Society, 2003.

[Urgaonkar 05] Urgaonkar, Bhuvan; Shenoy, Prashant; Chandra, Abhishek; & Goyal, Pawan. "Dynamic Provisioning of Multi-Tier Internet Applications," 217-228. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[UVIC 03] University of Victoria. *Curriculum of the Bachelor of Software Engineering (BSENG) Program*. http://www.bseng.uvic.ca (2003).

[Valetto 05] Valetto, Giuseppe; Kaiser, Gail; & Phung, Dan. "A Uniform Programming Abstraction for Effecting Autonomic Adaptations onto Software Systems," 286-297. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005*). Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[van Renesse 03] van Renesse, Robert; Birman, Ken; & Vogels, Werner. "Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining." *ACM Transactions on Computer Systems 21*, 2 (May 2003): 164-206.

[Verma 03] Verma, D. C.; Sahu, S.; Calo, S.; Shaikh, A.; Chang, I.; & Acharya, A. "SRIRAM: A Scalable Resilient Autonomic Mesh." *IBM Systems Journal 42*, 1 (2003): 19-28. http://www.research.ibm.com/journal/sj/421/verma.html.

[Wagner 04] Wagner, Earl. "Understanding and Debugging System Configuration." *Conference on the Human Impact and Application of Autonomic Computing Systems (CHIACS2)*. Yorktown Heights, NY (IBM T. J. Watson Research Center), April 21, 2004. San Jose, CA: IBM Almaden Research Center, 2004. http://www.almaden.ibm.com /asr/chiacs/presentations/wagner.pdf.

[Walsh 04] Walsh, William E.; Tesauro, Gerald; Kephart, Jeffrey O.; & Das, Rajarshi. "Utility Functions in Autonomic Systems," 70-77. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Want 03] Want, R.; Pering, T.; & Tennenhouse, D. "Comparing Autonomic and Proactive Computing." *IBM Systems Journal 42*, 1 (2003): 129-135. http://www.research.ibm.com/journal/sj/421/want.html.

[Weyns 05] Weyns, D.; Schelfhout, K.; & Holvoet, T. "Architectural Design of a Distributed Application with Autonomic Quality Requirements," 52-58. *Proceedings of Design and Evolution of Autonomic Application Software (DEAS 2005)*. St. Louis, MO, May 21, 2005. New York, NY: ACM Press, 2005.

[Whisnant 03] Whisnant, K.; Kalbarczyk, Z. T.; & Iyer, R. K. "A System Model for Dynamically Reconfigurable Software." *IBM Systems Journal 42*, 1 (2003): 45-59. http://www.research.ibm.com/journal/sj/421/whisnant.html.

[White 04] White, Steve R.; Hanson, James E.; Whalley, Ian; Chess, David M.; & Kephart, Jeffrey O. "An Architectural Approach to Autonomic Computing," 2-9. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[White 05] White, Tony; Calvert, Dan; & Litkey, Jay. "Design of an Autonomic Element for Server Management," 147-158. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Wildstrom 05] Wildstrom, Jonathan; Stone, Peter; Witchel, Emmett; Mooney, Raymond J.; & Dahlin, Mike. "Towards Self-Configuring Hardware for Distributed Computer Systems," 241-249. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Xu 05] Xu, Jing; Adabala, Sumalatha; &Fortes, José A. B. "Towards Autonomic Virtual Applications in the In-VIGO System," 15-26. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Yan 04a] Yan, Hong; Garlan, David; Schmerl, Bradley; Aldrich, Jonathan; & Kazman, Rick. "DiscoTect: A System for Discovering Architectures from Running Systems," 470-479. *Proceedings of the ACM/IEEE 26th International Conference on Software Engineering (ICSE 2004)*. Edinburgh, Scotland, May 23-28, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Yan 04b] Yan, H.; Aldrich, J.; Garlan, D.; Kazman, R.; & Schmerl, B. *Discovering Architectures from Running Systems: Lessons Learned* (CMU/SEI-2004-TR-016, ADA441834). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications /documents/04.reports/04tr016.html.

[Yellin 03] Yellin, D. M. "Clockwork: A New Movement in Autonomic Systems." *IBM Systems Journal 42*, 1 (2003): 85-97. http://www.research.ibm.com/journal/sj/421/russell.html.

[Zhang 04] Zhang, Zheng; Lin, Shiding; Lian, Qiao; & Jin, Chao. "RepStore: A Self-Managing and Self-Tuning Storage Backend with Smart Bricks," 122-129. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

[Ziebart 05] Ziebart, Brian D.; Roth, Dan; Campbell, Roy H.; & Dey, Anind K. "Automated and Adaptive Threshold Setting: Enabling Technology for Autonomy and Self-Management," 204-215. *Proceedings of the IEEE Second International Conference on Autonomic Computing (ICAC 2005)*. Seattle, WA, June 13-16, 2005. Los Alamitos, CA: IEEE Computer Society, 2005.

[Zilio 04] Zilio, Daniel C.; Zuzarte, Calisto; Lightstone, Sam; Ma, Wenbin; Lohman, Guy M.; Cochrane, Roberta J.; Pirahesh, Hamid; Colby, Latha; Gryz, Jarek; Alton, Eric; Liang, Dongming; & Valentin, Gary. "Recommending Materialized Views and Indexes with IBM DB2 Design Advisor," 180-188. *Proceedings of the IEEE First International Conference on Autonomic Computing (ICAC 2004)*. New York, NY, May 17-18, 2004. Los Alamitos, CA: IEEE Computer Society, 2004.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	April 2006	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Autonomic Computing	FA8721-05-C-0003

6. AUTHOR(S)

Hausi A. Müller, Liam O'Brien, Mark Klein, Bill Wood

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2006-TN-006

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

This report examines selected aspects of autonomic computing and explores some of the strengths and weaknesses of that technology. It also makes connections between autonomic computing and current work in several initiatives at the Software Engineering Institute. Furthermore, it describes the potential and impact of autonomic computing for Department of Defense (DoD) systems and outlines some of the challenges for the DoD as it moves to exploit autonomic computing technology.

14. SUBJECT TERMS	15. NUMBER OF PAGES
autonomic computing, Department of Defense, software complexity	60

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL