Autonomic Computing in Action

Ric Telford
Director, Strategy and Technology
Autonomic Computing
April 14, 2004
Agenda

- Putting the pieces together
- Autonomic Computing system examples
  - Building a self-healing system
  - A self-configuring system
- Autonomic Computing research
  - Self-configuring devices
  - Self-optimizing web servers
  - Utility functions
  - Autonomic Computing Storage
  - Ethnographic Studies
When autonomic managers are introduced, the AMs must register with the fabric. This knowledge must be passed to the fabric, AND the touchpoints must register themselves with the fabric. Now the system is ready to run! CBEs can flow, MAPE loops can make decisions, resources can be managed.

Initially, there is the fabric. The services in the fabric must register themselves with the Fabric Registries. When managed resources are introduced, knowledge about each managed resource is created. This knowledge must either be passed to the AMs, OR it must be made available via k-services.

Symptom: something that indicates the existence of something else (from webster.com)

AMs and TPs must talk to fabric registries to find the appropriate manager/touchpoint. CBEs are created by MRs, CBEs are passed to the fabric, which uses the deliver-notification interaction style to deliver a symptom and passes that to ‘E’. ‘E’ executes on the plan, using the fabric and the perform operation interaction style.
Disparate pieces and parts

Tools focused on individual products

No common interfaces among tools

No synergies in building or correlating log entries

Common set of tools

Common interfaces among tools

Common Base Event submitted to OASIS
Autonomic computing self-healing systems

Applications

Database

Application Server

Servers

Storage devices

Networks

Adapters

Feedback

Autonomic manager

Knowledge

Data

Policies

Symptom

Policy engine

Policy engine

Adapters

Autonomic manager

Policies

Feedback
In a basic problem bypass and resolution service flow, most tasks and information exchanges are performed manually.

<table>
<thead>
<tr>
<th>Customer</th>
<th>Customer-Detected Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Service</td>
<td>Incident Management</td>
</tr>
<tr>
<td>Problem Manager</td>
<td>Identify Problem</td>
</tr>
<tr>
<td>Problem Analyst</td>
<td>Bypass Problem</td>
</tr>
<tr>
<td>Problem Queue Manager</td>
<td>Report Problem Status and Trends</td>
</tr>
<tr>
<td>Automated Tools</td>
<td>Analyze Problem</td>
</tr>
<tr>
<td></td>
<td>Manage Problem Resolution</td>
</tr>
<tr>
<td></td>
<td>Resolved Problem</td>
</tr>
</tbody>
</table>
In order for AM to operate:
- Knowledge must be created about the symptoms and treatable causes.
- Managed Resource properties of the MR, like what sensors and effectors are implemented, must be created.
- Knowledge must be created in an abstract manner, based on customer policies and goals.
- All this knowledge must be passed into the AM.

Knowledge Creation

Symptom 1
Component: WAS
Situation: DEPENDENCY NOT MET
Correlation: MATCH
Treatable Cause: Path configured wrong
Symptom: Symptom 1
Action: CONFIG PATH
In a more autonomic workflow, the infrastructure can detect and bypass many problems and the tools help automate the information exchange between tasks.
A self-configuring system: Think Dynamics
Focus on Autonomic Computing Research

- Autonomic Storage
- Autonomic PCs
- Autonomic Middleware
- Autonomic Database Technology
- Autonomic Computing Core Research
- Autonomic Capabilities for eServer

Policy technologies
System prototyping
AC building block technologies
Autonomic Computing Research PCE - Personal Software Configuration Engine

- Automate SW maintenance & migration on personal devices
  - “Upgrade all my applications”
  - “Make my new laptop work like the old one”
  - “Migrate most valuable Palm apps to my PC”
An important scenario: Workload surge

- Systems can go from steady state ...

- to overloaded without warning

Few minutes later...
Autonomic Computing: Dynamic Surge Protection

Monitor

Analyze (Forecaster)

Plan (Perf Modeler)

Execute (Controller)

Knowledge

Sensors

Effectors

Element

Web Application Servers

Database

Driver (simulates Internet in/out)

Surge Button

© 2004 IBM Corporation
1. Steady State

#Active Servers

#Requested Servers

Actual BOPS

Predicted BOPS

Response Time
2. Monitor, Detect Surge

- #Active Servers
- #Requested Servers
- Actual BOPS
- Predicted BOPS
- Response Time
3. Forecast, Provision Servers

- #Active Servers
- #Requested Servers
- Actual BOPS
- Predicted BOPS
- Response Time
4. Monitor, Remove Servers

- #Active Servers
- #Requested Servers
- Actual BOPS
- Predicted BOPS
- Response Time
Utility Functions and Autonomic Computing

- Utility functions can guide autonomic decision making
  - Self-optimization: natural way to express optimization criteria
    - Declarative: preferable to implicitly hard-coded in special purpose algorithms
  - Derivable from business objectives (e.g. optimize total profits)
    - Can translate to computing metrics at different levels
  - Exploring applications in Workload Management, other areas
IBM IceCube Server

- Lego-like Collection of ‘Intelligent Bricks’
- Fail-in-place policy: bad bricks are left in place
- 7 x smaller than equivalent standard systems
- Fast, power-hungry components (CPU etc) ok
- Includes resource allocation software
- First Application: Petabyte-class Storage Server
  - intended to be managed by one person

Prototype Brick:
- (12) 2.5” disks
- 8-port Switch
- Linux on fast CPU

Full IceCube System
blue: Storage Bricks
yellow: Compute Bricks
3D mesh @ 10 Gb/s per link

© 2004 IBM Corporation
Ethnographic Field Studies of Middleware Admins

- What tasks do Autonomic Systems need to simplify for Sys Administrators?

- Study and Analysis methods used to study Sys Administrators at work

- Early finding: Collaboration compensating for system complexity and simple tools
  “Let’s do it together”
What do those admins do? (Video Analysis)

- ~90% time collaborating
- Main topics
  - collaboration
  - configuration
  - strategies

- ~60% time collaborating
- Main topics
  - commands
  - strategies
  - collaboration
  - configuration
Preliminary findings for AC Console UI designs

• **Collaboration is a vital part of admin practice**
  - Help establish trust among participants
  - Support side-by-side exploration of system events, configurations, etc.
  - Enable ready, shared access to all relevant information

• **Troubleshooting depends on information access**
  - Need clear and consistent naming conventions for system objects
  - Support easy correlation of log data and config settings
  - Make logs browsable, visualizable, query-able, annotatable
  - Place events in the context of end-to-end systems rather than of components

• **Admins build their own tools, configure their own environments**
  - Support “tools built by admins for admins”
  - Provide composable components and appropriate combination methods
  - Enable widespread sharing of admin developed tools
Summary

- Autonomic Computing architecture components and interfaces come together to form a system
- Today’s IT processes can be improved through Autonomic Computing
- IBM Research is working on some of the hard problems in Autonomic Computing, but there is still a lot of work to do!
© Copyright IBM Corporation 2004. All rights reserved.

The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.

IBM, the IBM logo, the e-business logo and other IBM products and services are trademarks or registered trademarks of the International Business Machines Corporation, in the United States, other countries or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries or both.

Microsoft, Windows, Windows NT and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries or both.

All other trademarks, company, products or service names may be trademarks, registered trademarks or service marks of others.

Disclaimer: NOTICE – BUSINESS VALUE INFORMATION IS PROVIDED TO YOU 'AS IS' WITH THE UNDERSTANDING THAT THERE ARE NO REPRESENTATIONS OR WARRANTIES OF ANY KIND EITHER EXPRESS OR IMPLIED. IBM DISCLAIMS ALL WARRANTIES INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IBM DOES NOT WARRANT OR MAKE ANY REPRESENTATIONS REGARDING THE USE, VALIDITY, ACCURACY OR RELIABILITY OF THE BUSINESS BENEFITS SHOWN. IN NO EVENT SHALL IBM BE LIABLE FOR ANY DAMAGES, INCLUDING THOSE ARISING AS A RESULT OF IBM'S NEGLIGENCE, WHETHER THOSE DAMAGES ARE DIRECT, CONSEQUENTIAL, INCIDENTAL, OR SPECIAL, flowing from your use of or inability to use the information provided here with or results even if IBM has been advised of the possibility of such damages. THE ULTIMATE RESPONSIBILITY FOR ACHIEVING THE CALCULATED RESULTS REMAINS WITH YOU.