Are Your Sites Down? Requirements-Driven Self-Tuning for the Survivability of Web Systems

Bihuan Chen\textsuperscript{1}, Xin Peng\textsuperscript{1}, Yijun Yu\textsuperscript{2}, Wenyun Zhao\textsuperscript{1}

\textsuperscript{1}School of Computer Science, Fudan University, China
\textsuperscript{2}Centre for Research in Computing, The Open University, UK

pengxin@fudan.edu.cn
www.se.fudan.edu.cn/pengxin
Web Systems: Outages => Outrages

Amazon.com (June 29, 2010)
- an outage for more than 3 hours
- searching services and shopping carts didn’t work
- $51,400 loss per minute
- shares closed down by 7.8%

Google Apps services (just last week)
- The affected users are unable to access Google Docs List...

12/6/11

http://www.se.fudan.edu.cn
Survivability of Web Systems

• Survivability rather than absolute reliability
  – absolute reliability is often expensive, or even impossible

• Survivability [Knight et al. @ 2004] capability of ensuring crucial services under severe or adverse conditions, with
  – acceptable quality degradation
  – or even sacrifice of some desirable services

Our Work: A requirements-driven self-tuning method for survivability of Web-based Systems
Agenda

• Self-Tuning for Survivability
• Our Method
• Experimental Study
• Discussion
• Conclusion
Achieve Survivability by Self-Tuning

• Survivability: to ensure crucial services under severe or adverse conditions, with
  - acceptable quality degradation: quality requirements tradeoff at runtime
  - or even sacrifice of some desired services: functional requirements tradeoff at runtime

General Idea: self-tuning for survivability
- Follow the general MAPE control loop for self-adaptive systems
- Define measurable objective function for the survivability assurance as feedback
- Requirements-driven: goal models as the knowledge base to support tradeoff decisions
Target of survivability assurance:
To maximize the capability of the system to create value for the e-commerce company running the system.
Runtime Tradeoff Scenarios in OSS

- **Scenario 1**
  - Environment: the system load is heavy, which could reduce the number of successful order processing or even crash down the system
  - **Quality tradeoff**: “response time” over “usability” -> goal reconfiguration: "Plain Form" instead of "Rich Form" for viewing product details

- **Scenario 2**
  - Environment: after Scenario 1, the system is still overloaded with poor user-friendliness and slow response
  - **Functional tradeoff**: unbind some non-crucial service (e.g., making reviews)

- **Scenario 3**
  - Environment: the system load is light
  - **Quality tradeoff**: “maximum usability” over “minimum response time” or **Functional tradeoff**: rebind certain service (e.g., making reviews)
Research Problems

How to interpret survivability assurance in a measurable way to provide runtime feedback?

A value-based interpretation: define a runtime earned value measurement from the business perspective

What kind of control mechanisms are used?

Feedback Control Loop

How to implement runtime tradeoff decisions?

Quality tradeoffs [Peng et al. @ RE 2010]

preference-based goal reasoning to dynamically configure the OR-decomposed goals (variation points)

Functional tradeoffs

value-based goal reasoning to dynamically unbind/bind the desired goals
Value-based Interpretation of Survivability Assurance

• **Value-Based Software Engineering (VBSE)**
  - a discipline that considers economic aspects (business value considerations) within the whole software development life cycle. [Barry Boehm]

• **Survivability assurance implies the protection of the value delivered by a system** [Knight et al. @ 2004]

• **A value interpretation should be defined to**
  - enable the runtime computation of system earned value
  - support value-based requirements (both quality and functional) tradeoff decisions
Quantify the Earned Value at Runtime

- For Web-based systems, earned value usually can be measured by successful transactions of different types
  - Baseline transactions: reflect the main value propositions of the system, e.g. paid product ordering transactions in OSS
  - Other transactions: consider the value contribution relative to the baseline transactions

- Example: earned value measurement in OSS
  - Factors: (1) sales (2) product details (3) reviews (4) consultations (5) recommended products

<table>
<thead>
<tr>
<th>Time Slot</th>
<th>Transactions</th>
<th>Total ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(1) ($)</td>
<td>(2) (#)</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>150</td>
<td>5</td>
</tr>
</tbody>
</table>

- At time slot 1: $100 * 5% + 8 * $0.08 + 7 * $0.032 + 9 * $0.04 + 3 * $0.048 = $6.368
Annotate Value Contribution in Goal Models

• Relative Value Contribution to Parent Goals
  - a value between 0.0 and 1.0 indicating to what extent the fulfillment of a sub-goal will contribute to the achievement of the value of their parent goals.

• Can be considered as an economic problem of consumer behavior analysis
  - ask the question: If the hard goal is unbound, what is the probability of not achieving its parent goal when all other sibling sub-goals have been achieved?
Our MAPE-based Feedback Control Structure

- **Plan**
  - [earned value decreases]
  - Preference Ranks of Soft Goals

- **Feedback Controller**
  - [earned value increases]
  - Earned Value, Quality Measurements

- **Analyze**
  - Earned Value Indicator
  - Runtime Data

- **Execute**
  - Quality Goal Reasoner
    - [no new configuration found]
  - Functional Goal Reasoner
    - A Leaf-Level Hard Goal (un/bind/bind)
  - Goal Model Configurator
    - Mappings
  - Architecture Configurator
    - Variation Points Configurations

- **Monitor**
  - Sensor/Log
  - Running Web System

- **http://www.se.fudan.edu.cn**
Running System

PID Controller

Preference-driven Goal Reasoner

Preference Ranks of Softgoals

goal configurations

Architecture Configurator

Architecture Reconfiguration

Running System

Process under Control

Value Indicator

runtime data

Feedback: Earned Value

[Peng et al. @ RE 2010]

http://www.se.fudan.edu.cn
Value-based Functional Goal Reasoner

• Differentiate crucial goals and desired goals
  - Relative root value: the product of the relative parent values in the path from the hard goal to the root goal
  - Crucial and desired goals: for a leaf-level hard goal
    ✓ If its relative root value is 1.0 => crucial goal
    ✓ Otherwise
      □ if its parent goal’s relative root value is 1.0 and its parent goal has only one sub-goal => crucial goal
      □ else => desired goal
  - Example
    ✓ Goal2 is crucial, Sub1 and Sub2 are desired
    ✓ If Sub2 is unbound
      □ Sub1 becomes crucial

• Function goal reasoner
  - Find a desired goal with minimum relative root value when unbind
  - Find the recently unbound desired goal when rebind

12/6/11
http://www.se.fudan.edu.cn
Experimental Study: Setting

• **Subject System**
  - Online shopping system developed in Java + JSP and deployed on Tomcat 5.5

• **Concurrent access stimulation: JMeter 2.3.4**

• **Experiment Running**
  - Continuous runs of about 77 minutes with different loads
  - Earned value and qualities (response time, usability, cost, availability)
    ✓ Analyzing Web server log files per minute

http://www.se.fudan.edu.cn
Experimental Study: Results

- **Timed delay**
  - Uncertainty in whether a radical action has taken effect in the target system [Cheng et al. @ 2008]

<table>
<thead>
<tr>
<th>Methods</th>
<th>Mean ($)</th>
<th>Standard Deviation</th>
<th>(Un)Bind Frequency (#)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>33.16</td>
<td>5.02</td>
<td>/</td>
</tr>
<tr>
<td>Quality Tradeoff</td>
<td>35.24</td>
<td>3.90</td>
<td>/</td>
</tr>
<tr>
<td>Quality and Functional Tradeoff-0</td>
<td>35.61</td>
<td>4.73</td>
<td>29</td>
</tr>
<tr>
<td>Quality and Functional Tradeoff-1</td>
<td>36.88</td>
<td>3.02</td>
<td>19</td>
</tr>
<tr>
<td>Quality and Functional Tradeoff-2</td>
<td>36.25</td>
<td>4.81</td>
<td>20</td>
</tr>
</tbody>
</table>

Quality and Function Tradeoff-?: quality and function tradeoff with a window of timed delay of ? time units
Discussion

• Cost (resource) Vs. Benefit (value)
  - Unbind the desired goal with low relative root value and large resources
  - Bind the desired goal with high relative root value and small resources

• Diagnosing for quality degradation
  - Not consider the root cause of quality degradation when reasoning
    ✓ Make incorrect decisions, i.e., sacrifice the qualities or functionalities that have little to do with the degradation
  - Integrate diagnosis for quality degradation with goal reasoning

• Long-term value feedback
  - Some value feedback takes a long time, e.g.: Quality degradation -> reputation down -> customer losing -> value decrease
  - Possible solution: involve prediction in earned value computation by capturing some early indication of long-term influences
Conclusion

• Survivability rather than absolute reliability for Web-base systems

• Requirements-driven self-tuning for Survivability
  - MAPE control loop with goal models as knowledge base
  - runtime earned value from the business perspective as feedback
  - involve both quality tradeoff and functional tradeoff
Thanks!

Q&A
Three Possible Planning Paths

- In the Value Fluctuation Tolerance Range?
  - Monitored earned business value of the last time period: val
  - Expected earned business value (the average value in the past): expVal
  - \( \frac{(val - expVal)}{expVal} \)

- Path 1
  - Earned business value decreases by \( \alpha \) (i.e., survivability assurance is unachieved)
  - Quality goal reasoner finds new variation points configurations

- Path 2
  - Earned business value decreases by \( \alpha \) (i.e., survivability assurance is unachieved), and quality goal reasoner finds no new configurations
  - Functional goal reasoner finds a leaf-level hard goal to unbind to ensure other goals' satisfaction

- Path 3
  - Earned business value increases by \( \beta \) (i.e., survivability assurance is achieved)
  - Functional goal reasoner finds the recently sacrificed leaf-level hard goal to rebind to improve the functional integrity
Goal Model Configurator

• Receiving variation points configurations
  - Unbind the old goal, and bind the new goal
• Receiving a desired goal to unbind/bind (inverse of unbind)
  - Unbind the desired goal, and
  - Case 1: the desired goal's relative parent value is 1.0
    ✓ Unbind the desired goal's sibling goals
    ✓ Recursively unbind the desired goal's parent goal
  - Case 2: the desired goal has no sibling goals
    ✓ Recursively unbind the desired goal's parent goal
Experimental Study (cont.)

Self-tuning process with 70 concurrent users and one-minute timed delay

Examples
At time 13: profit: $40.41, response time: 4605ms, usability: 10.0, cost: $34.50, and availability: 0.70
At time 14: profit: $25.91, response time: 8985ms, usability: 10.0, cost: $19.5, and availability: 0.87

Drive the reconfiguration from Fuzzy Search to Accurate Search, from Rich Form to Plain Form

At time 8: profit decreased, no new VPs configuration was found, unbind Recommended Products
At time 10: profit increased, bind the recently unbound desired goal Recommended Products

12/6/11

http://www.se.fudan.edu.cn
Performance

• Randomly generate goal models with size from 25 to 150

• Quality goal reasoner: less than 3 seconds

• Functional goal reasoner: less than 1 second