Detecting Mode Inconsistencies in Component-Based Embedded Software

DSN 2007
Workshop on Architecting Dependable Systems

27-06-2007
Edinburgh, Scotland - UK

Hasan Sözer
Outline

• The problem
  ■ Mode inconsistency
• The solution approach for error detection
• An implementation of the approach
• Discussion
  ■ Implementation issues
  ■ Integrating diagnosis and recovery
• Conclusion
The Mode Inconsistency Problem

• An example:
  ■ system lock-up in TV

• User perspective:
  ■ TV screen is blanked out or frozen
  ■ No response to user actions
    ♦ Teletext page cannot be changed
The Mode Inconsistency Problem

• An example:
  ■ system lock-up in TV

• System perspective:
  ■ Each sub-system seems to work fine
  ■ Synchronization is lost
The Mode Inconsistency Problem

Teletext Page Handler

Display Manager
The Mode Inconsistency Problem

User is watching TV

User is browsing Teletext
The Mode Inconsistency Problem

No page to display, blank the screen

show teletext
The Mode Inconsistency Problem

- system lock-up in TV: Observations
  - system is **unaware**
  - Existing error detection mechanisms are too low level
    - Stack overflow, memory corruption, …
Error Detection: Solution Approach

- Model the working modes of each component
- Model the working modes of the system
- Define links among three levels of abstraction:
  - Specification of the system working modes
  - Specification of the working modes of components
  - Run-time behavior of components (implementation)
Error Detection: Solution Approach

Working Modes of the System

Working Modes of Component \( C_0 \)

Working Modes of Component \( C_{n-1} \)

Implementation of Component \( C_0 \)

Implementation of Component \( C_{n-1} \)

derivation of mutual consistency condition for \( C_0 \ldots C_{n-1} \)

observation of component modes at run-time
Error Detection: Solution Approach

- Implementation of the approach
  - How to model the working modes?
  - How to map the models to each other?
  - How to use the mappings to detect errors?
An Implementation of the Approach

- Each working mode $s$ of a component $C$ is represented by a bit vector.
- Each bit corresponds to a working mode $g$ of the system.
- Each bit is set to:
  - 0 if $C$ cannot be in working mode $s$ while system is in working mode $g$.
  - 1 otherwise.
- There is an error if current working modes of components are not consistent:
  - Bit vectors have no common bit with value 1.
Working Example: Error Detection

<table>
<thead>
<tr>
<th>Txt. Page Handler</th>
<th>map</th>
<th>Display Mgr.</th>
<th>map</th>
</tr>
</thead>
<tbody>
<tr>
<td>off</td>
<td>10</td>
<td>default screen</td>
<td>10</td>
</tr>
<tr>
<td>txt</td>
<td>01</td>
<td>on screen display</td>
<td>10</td>
</tr>
<tr>
<td>subtitle</td>
<td>11</td>
<td>txt left-half screen</td>
<td>01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>txt full screen</td>
<td>01</td>
</tr>
</tbody>
</table>

$10 \land 01 = 0 \rightarrow \text{error}$
Properties of the Detection Mechanism

• The model of the system working modes is independent of its internal decomposition
• Components do not have to know about the working modes of the other components
• The mechanism can be incrementally introduced
• Low-cost
  ■ effort to map the working modes:
    ♦ Developers have to set the bits to 0 or 1 based on consistency
  ■ space complexity:
    ♦ (number of components) x (number of system working modes)
  ■ time complexity:
    ♦ Error checking can be done with a single AND operation over bit-vector representations of component working modes
Implementation Issues

• When to check for consistency?
  ▪ If there is no transaction concept in the system

• Current implementation: Periodic Error checking at a separate, lowest priority, preemptive task
  ▪ Checking at a stable system state
  ▪ Does not interfere with the other system functions
    ■ External observer
  ▪ Does not degrade the system performance
Integration of Recovery & Diagnosis

• Recovery
  ■ Backward recovery
    ◦ Check-pointing consistent states and roll-back
  ■ Cheap/Simple solution: restart the system
  ■ Local-recovery (e.g. microreboot)
    ◦ Does the system support?
    ◦ How to localize the error? (Diagnosis)

• Diagnosis
  ■ Voting Based
    ◦ Is there a consensus on the global system state?
    ◦ Who does not agree?
Conclusion

• **Mode Inconsistency**
  - Can lead to serious failures
  - Such errors should be detected for fault tolerance
  - Existing error detection mechanisms are too low-level and fall short to detect such errors

• **A simple, lightweight solution for error detection**
  - Specifying the consistency of component working modes with respect to system working modes
  - Monitoring component working modes at run-time
  - Checking the mutual consistency of monitored working modes that are represented by bit-vectors
Questions

- Working Modes of the System
  - Working Modes of Component C₀
  - Working Modes of Component Cₙ₋₁
  - Implementation of Component C₀
  - Implementation of Component Cₙ₋₁