Simple crosscutting concerns are not so simple

an AOSD2007 paper

Magiel Bruntink
Arie van Deursen
Maja D’Hondt
Tom Tourwé
Simple crosscutting concerns are not so simple
Simple crosscutting concerns are not so simple
such as tracing
Simple crosscutting concerns are not so simple in terms of the amount of variability in the implementation.
Research question

“Is the idioms-based implementation of a crosscutting concern sufficiently systematic such that it is suitable for an aspect-oriented solution”
Tracing @ ASML

* "Every" function has to trace its arguments
* Input arguments must be traced at the beginning
* Output arguments must be traced at the end
* Approx 16% tracing code in 4 components studied (83 kLoC)
int CC_scan_pending(
    int scan_id,
    CC_scan_component scan_component,
    bool *scan_pending_p) {

    trace("CC": TRACE_INT, func_name,
        " (scan_id = %d;scan_component = %s)",
        scan_id,
        CC_SCAN_COMPONENT_ENUM2STR(scan_component));

<< BASE FUNCTIONALITY >>

    trace("CC": TRACE_INT, func_name,
        " (scan_pending = %b) = %R",
        *scan_pending_p, r);

}
<table>
<thead>
<tr>
<th></th>
<th>CC1</th>
<th>CC2</th>
<th>CC3</th>
<th>CC4</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoC</td>
<td>29339</td>
<td>17848</td>
<td>31165</td>
<td>4985</td>
<td>83337</td>
</tr>
<tr>
<td>functions</td>
<td>161</td>
<td>134</td>
<td>174</td>
<td>68</td>
<td>537</td>
</tr>
<tr>
<td>parameters</td>
<td>108</td>
<td>71</td>
<td>65</td>
<td>49</td>
<td>249</td>
</tr>
<tr>
<td>tracing</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>macro's</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>component</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>function</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>
function-level variability

Group all functions that invoke tracing in the same way

```c
int CC_scan_pending(int scan_id, CC_scan_component scan_component, bool *scan_pending_p) {
    trace("CC", TRACE_INT, func_name, " > (scan_id = %d;scan_component = %s)", scan_id, CC_SCAN_COMPONENT_ENUM2STR(scan_component));

    "<< BASE FUNCTIONALITY >>"

    trace("CC", TRACE_INT, func_name, " < (scan_pending = %b) = %R", *scan_pending_p, r);
}
```
set_ill_rema_and_wl   : trace "CC1" INT func_name
get_grating_indices   : trace CC INT f_name
CC_rq_PFNRM           : trace "CC1" INT func_name
CC_request_ags_height : trace "CC1" INT func_name
Standard tracing
Inconsistency
Specific tracing variant
<table>
<thead>
<tr>
<th></th>
<th>CC1</th>
<th>CC2</th>
<th>CC3</th>
<th>CC4</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>aspects</td>
<td>6</td>
<td>4</td>
<td>19</td>
<td>2</td>
<td>29</td>
</tr>
<tr>
<td>anomalies</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>
parameter-level variability

Group functions that have a parameter of a certain kind, and that convert that parameter in the same way

```c
int CC_scan_pending(
    int scan_id,
    CC_scan_component scan_component,
    bool *scan_pending_p)
{

    trace("CC", TRACE_INT, func_name,
        ">
            (scan_id = %d; scan_component = %s)",
            scan_id,
            CC_SCAN_COMPONENT_ENUM2STR(scan_component));

    << BASE FUNCTIONALITY >>

    trace("CC", TRACE_INT, func_name,
        ">
            (scan_pending = %b) = %R",
            *scan_pending_p, r);

}
```
Extraction from code

```c
bool:       ID

bool:      not_traced

REQ_MODE:   ID

chuck_id_enum: CHUCK_ID_ENUM2STR()
```
<table>
<thead>
<tr>
<th></th>
<th>CC1</th>
<th>CC2</th>
<th>CC3</th>
<th>CC4</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>inconsistently traced types</td>
<td>11</td>
<td>6</td>
<td>39</td>
<td>8</td>
<td>64</td>
</tr>
<tr>
<td>consistently traced types</td>
<td>15</td>
<td>5</td>
<td>16</td>
<td>19</td>
<td>55</td>
</tr>
</tbody>
</table>
Implications

Migration trade-off: textual equivalence vs. solution quality

Textual equivalence: large number of exceptions to generic aspects

Solution quality: possible differences between source and target
Conclusions

- Migration is feasible
- Migration trade-off needs attention
- We have developed technology to help