Software Performance Antipatterns challenges: how to get rid of worms before contaminating the apple?

Catia Trubiani
Dipartimento di Ingegneria e Scienze dell’Informazione e Matematica (DISIM), University of L’Aquila, Italy
http://www.di.univaq.it/catia.trubiani

A common thought

Life was much easier when APPLE and BLACKBERRY were just fruits

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
What happens today

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Technology and its drawbacks

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
A proactive approach...

... get rid of worms before contaminating the apple!

Software (Performance) Antipatterns

Two main challenges:
1- Identify when a software system is (performance) sick
2- Fix the issues to get (performance) improvements

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

(Performance) Antipatterns

» Negative features of a software system
  » Conceptually similar to design patterns: recurring solutions to common design problems
  » The definition includes common mistakes (i.e. bad practices) in software development as well as their solutions

» What to avoid and how to solve (performance) problems!


"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Outline of this talk

» Problem statement

» Reasoning on Performance Antipatterns

» Performance Antipatterns in Modeling Languages
  - Unified Modeling Language (UML)
  - Domain Specific Language (PCM)
  - Architecture Description Language (AEmilia)

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

PROBLEM STATEMENT...

AND DIFFERENT WAYS TO APPROACH IT...
"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
**Problem Reasoning on Perf. Antipat.**

**Performance results**

---

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

---

**Software architectural models vs Performance results**

The interpretation of performance results is not a trivial task!

---

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
### Software Performance Feedback: state-of-the-art

<table>
<thead>
<tr>
<th>Approach</th>
<th>(Annotated) Software Architectural Model</th>
<th>Performance Model</th>
<th>Framework</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Antipattern-based</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Williams et al. 2002</td>
<td>Software Execution Model</td>
<td>System Execution Model</td>
<td>SPE-ED</td>
</tr>
<tr>
<td>Parsons et al. 2008</td>
<td>JEE systems</td>
<td>Reconstructed runtime model</td>
<td>PAD</td>
</tr>
<tr>
<td><strong>Rule-based</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Barber et al. 2002</td>
<td>Domain Reference Arch.</td>
<td>Simulation Model</td>
<td>RARE / ARCADE</td>
</tr>
<tr>
<td>Xu 2010</td>
<td>UML</td>
<td>Layered QN</td>
<td>Perf. Booster</td>
</tr>
<tr>
<td><strong>Design Space Exploration</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simple Criteria</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zheng et al. 2003</td>
<td>UML</td>
<td>Simulation Model</td>
<td>-</td>
</tr>
<tr>
<td>Ipek et al. 2008</td>
<td>Artificial Neural Network</td>
<td>Simulation Model</td>
<td>-</td>
</tr>
<tr>
<td><strong>Meta-Heuristics</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Canfora et al. 2005</td>
<td>Workflow Model</td>
<td>Workflow QoS Model</td>
<td>-</td>
</tr>
<tr>
<td>Martens et al. 2010</td>
<td>PCM</td>
<td>Simulation Model</td>
<td>PerOpteryx</td>
</tr>
</tbody>
</table>

*SPA challenges: how to get rid of worms before contaminating the apple?*, Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

### Antipattern-based approaches
- They make use of antipatterns knowledge to cope with performance issues


*SPA challenges: how to get rid of worms before contaminating the apple?*, Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Rule-based approaches

- They encapsulate general knowledge on how to improve system performance into executable rules.


Design space exploration - simple criteria

- They explore the design space by examining alternatives that can cope with performance flaws.


Design space exploration – Metaheuristics
- They make use of evolutionary algorithms that look for design alternatives aimed at improving the system performance


"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Software performance process: introducing **automation**

- **Modeling**
  - (Annotated) Software Architectural Model
  - Model2Model Transformation

- **Analysis**
  - Performance Model
  - Model Solution

- **Refactoring**
  - Performance Results
  - Detection & Solution of Performance Antipatterns

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

**A bird’s-eye look to the problem**

1. Representing Antipatterns
2. Detecting Antipatterns
3. Solving Antipatterns

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Representing Antipatterns: What are the software architectural model elements we need for representing antipatterns?

"SPA challenges: how to get rid of worms before contaminating the apple?",
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Detecting Antipatterns: How to explore the architectural models to recognize antipattern occurrences?

"SPA challenges: how to get rid of worms before contaminating the apple?",
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Solving Antipatterns:
What are the refactoring actions that lead the architectural model to remove performance flaws?

"SPA challenges: how to get rid of worms before contaminating the apple?",
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Software Performance Antipatterns Classification

<table>
<thead>
<tr>
<th>Antipattern</th>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concurrent Processing</td>
<td>Processing cannot make use of available processors.</td>
<td>Restructure software or change scheduling algorithms to enable concurrent execution.</td>
</tr>
<tr>
<td>&quot;Pipe and Filter&quot;</td>
<td>The slowest filter in a &quot;pipe and filter&quot; architecture causes the system to have unacceptable throughput.</td>
<td>Break large filters into more stages and combine very small ones to reduce overhead.</td>
</tr>
<tr>
<td>Extensive Processing</td>
<td>Extensive processing impedes overall response time.</td>
<td>Move extensive processing so that it doesn’t impede high traffic or more important work.</td>
</tr>
<tr>
<td>The Ramp</td>
<td>Occurs when processing time increases as the system is used.</td>
<td>Select algorithms or data structures based on maximum size or use algorithms that adapt to the size.</td>
</tr>
</tbody>
</table>


"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Single-value vs Multiple-values

» Single-value: performance indices are evaluated in a certain interval, i.e. the mean, max or min values.

![Single-value](image)

» Multiple-values: performance indices are evaluated along the time, i.e. the values trend (or evolution).

![Multiple-values](image)

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
SPA: why are they complex?


<table>
<thead>
<tr>
<th>Antipattern</th>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blob</td>
<td>Occurs when a single class or component either 1) performs most of the work of an application or 2) holds all of the applications data. Either manifestation results in excessive message traffic that can degrade performance.</td>
<td>Refactor the design to distribute intelligence uniformly over the applications top-level classes, and to keep related data and behavior together.</td>
</tr>
</tbody>
</table>

What does it mean “most” of the work”?
What does it mean “excessive” message traffic?

“SPA challenges: how to get rid of worms before contaminating the apple?”, Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

How to make more “formal” (i.e. machine-processable) the specifications of antipatterns???

“SPA challenges: how to get rid of worms before contaminating the apple?”, Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Graphical representation of the "Blob" Antipattern

**PROBLEM:** "occurs when a single class or component either 1) performs most of the work of an application or 2) holds all of the applications data. Either manifestation results in excessive message traffic that can degrade performance"

**SOLUTION:** "Refactor the design to distribute intelligence uniformly over the applications top-level classes, and to keep related data and behavior together"

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Idea: building a vocabulary to include all model elements involved in the antipatterns specification

STATIC VIEW: software resources, relationships among them, ... to model static aspects

DYNAMIC VIEW: interactions such as messages between sw resources, ... to model dynamic aspects

DEPLOYMENT VIEW: hardware resources, ... to model deployment aspects

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

PAML: Perf. Antipatterns Modeling Language

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
The “Model Elements Specification” sub-MM: READ

Basic Idea: a performance antipattern can be formalized as a logical predicate $L_{\text{antipatName}}$

A logical predicate is made of (Static, Dynamic, Deployment) basic predicates, $B_P_i$

$$L_{\text{antipatName}} = B_{P_1} (\wedge, \lor) \ldots (\wedge, \lor) B_{P_n}$$
Auxiliary elements in the formalization process

Functions

> \( F_{functionName} \) elaborates information of the model

(e.g. \( F_{numMsgs} \) is a function counting the number of messages sent by an instance of the class/component model element)

Thresholds

> \( Th_{thresholdName} \) is a value used to establish the acceptable range of values for system features

(e.g. \( Th_{maxMsgs} \) is a threshold value representing the upper bound for an acceptable number of messages exchanged among two software instances. It can be estimated, for example, as the average number of all messages sent by all software entities, plus the corresponding variance)

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27–29 November, 2013
Thresholds in performance antipatterns specification/representation

<table>
<thead>
<tr>
<th>Antipattern</th>
<th>Design</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>Blob</td>
<td>$T_{\text{maxConnects}}$</td>
<td>Maximum bound for the number of connections a component is involved</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{maxMsgs}}$</td>
<td>Maximum bound for the number of messages sent by a component in a network link</td>
</tr>
</tbody>
</table>

Design: thresholds referring to design characteristics (e.g., high/low number of connections, number of exchanged messages).

Performance: thresholds referring to performance indices (e.g., high/low device utilization, queue length).

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Each antipattern can be expressed by means of first-order logics.

But this is only OUR interpretation of their textual description.

Key-Question:

Once we have (somehow) “represented” antipatterns, how can we detect them in a software model?

"SPA challenges: how to get rid of worms before contaminating the apple?",
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

How to establish fair thresholds for antipatterns???

"SPA challenges: how to get rid of worms before contaminating the apple?",
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Define heuristics to calculate antipatterns thresholds

E.g., $T_{\text{maxConnects}}$ can be estimated as the average number of connections per component, by considering the entire set of software components in the software system, plus the corresponding variance.

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
A preliminary complex step: setting thresholds

<table>
<thead>
<tr>
<th>Antipattern</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blob</td>
<td>$T_{\text{maxConn}}$</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{maxMsgs}}$</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{maxHist}}$</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{maxNetUtil}}$</td>
<td>0.85</td>
</tr>
<tr>
<td>CPS</td>
<td>$T_{\text{maxQueue}}$</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{cpuMaxUtil}}$</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{diskMaxUtil}}$</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{cpuMinUtil}}$</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{diskMinUtil}}$</td>
<td>0.4</td>
</tr>
<tr>
<td>EST</td>
<td>$T_{\text{remMsgs}}$</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{remInst}}$</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>$T_{\text{minNetUtil}}$</td>
<td>0.3</td>
</tr>
</tbody>
</table>

“SPA challenges: how to get rid of worms before contaminating the apple?”,
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Performance Antipattern occurrences in ECS:

<table>
<thead>
<tr>
<th>Antipattern</th>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blob</td>
<td>libraryController performs most of the work, it generates excessive message traffic towards bookLibrary</td>
<td>Refactor the design to keep related data and behavior together, i.e. delegate some work from libraryController to bookLibrary</td>
</tr>
<tr>
<td>Concurrent Processing Systems</td>
<td>Processing cannot make use of the processor webServerNode</td>
<td>Restructure software or changing scheduling algorithms between processors libraryNode and webServerNode</td>
</tr>
<tr>
<td>Empty Semi Trucks</td>
<td>An excessive number of requests are sent by the userController to perform the task of registering users</td>
<td>Refactor the design combining items into messages to make better use of available bandwidth</td>
</tr>
</tbody>
</table>

“SPA challenges: how to get rid of worms before contaminating the apple?”,
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
» An example: detecting the EST antipattern instance

The (userController, RegisterUsers) instance satisfies the Empty Semi Trucks predicate, hence it must be pointed out to the designer for a deeper analysis.

“SPA challenges: how to get rid of worms before contaminating the apple?”, Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

And another Key-Question

Once we have (somehow) “detected” antipatterns, which refactoring actions must be taken to remove (some of) them?

First rough approach

First-Order-Logic representation of antipatterns can help: refactoring actions can be (automatically) obtained from negating predicates!

“SPA challenges: how to get rid of worms before contaminating the apple?”, Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
An example: solving the EST antipattern instance

*SPA challenges: how to get rid of worms before contaminating the apple?*, Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Performance Analysis of some refactored models

*SPA challenges: how to get rid of worms before contaminating the apple?*, Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Some more thoughts on refactoring

**Role-Based Modelling Language** (RBML) defines the available constructs (roles and associations among them) the number of elements playing that role characterizes a (part of a) system in terms of roles that are played by the elements of the system.

There exists a «\( T \) Role» for each type \( T \) in the modelling language used to specify the system.

It exists a «\( T \) Role» for each type \( T \) in the modelling language used to specify the system.


"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

---

Empty Semi Trucks (EST) - SOURCE ROLE MODEL

Elements from the antipattern detection phase

Contextual elements: additional model elements that can support the removal of the bad practice

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Performance Antipatterns in Modeling Languages

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Antipattern-based process

"SPA challenges: how to get rid of worms before contaminating the apple?”,
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
**MDE support: metamodel and model transformations**

- **Generic Modeling Language (i.e. UML + Marte profile)**
- **Domain Specific Language (i.e. Palladio Component Model)**
- **Architectural Description Language (i.e. ÀEmilia)**

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

---

**Antipatterns in concrete modeling languages**

- **Blob**
- **SoftwareEntity**
- **ProcessNode**
- **UML Component**
- **PCM Basic Component**
- **PCM Resource Container**
- **ARCHE_ELEM_TYPE**

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Is UML+Marte expressive enough to specify antipatterns?

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Is PCM expressive enough to specify antipatterns?

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Is Aemilia expressive enough to specify antipatterns?

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Expressiveness of the considered modeling languages

<table>
<thead>
<tr>
<th>Antipattern</th>
<th>UML + Marte profile Detectable</th>
<th>Solvable</th>
<th>Palladio Component Model Detectable</th>
<th>Solvable</th>
<th>AEmilia Detectable</th>
<th>Solvable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blob</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>≈</td>
<td>≈</td>
</tr>
<tr>
<td>Unbalanced Processing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concurrent Processing Systems</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>✖</td>
<td>✖</td>
</tr>
<tr>
<td>Pipe and Filter Architectures</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Extensive Processing</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Tower of Babel</td>
<td>✔</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>The Ramp</td>
<td>✔</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Traffic Jam</td>
<td>✔</td>
<td>✔</td>
<td>✖</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>More is Less</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
</tr>
</tbody>
</table>

✔ = fully detectable/solvable  ≈ = partially detectable/solvable  ✖ = not detectable/solvable

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Open Issues - Basics

- Gap textual/formal representations
- Threshold setting

- Uncertainty/incompleteness in:
  > Architectural/Performance models, antipatterns specifications, input parameters (e.g., workload, op. profile)

- Validation on larger model repositories (industrial case studies)

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Open Issues - Advanced

- Evaluating the framework on:
  > Precision (ratio of actual antipatterns found)
  > Recall (ratio of antipatterns found overall)

- Language-specific antipatterns (other languages to experiment?)

- Antipatterns at the code level
- Combining approaches (antipatterns with metaheuristics)
- Tool construction and integration

"SPA challenges: how to get rid of worms before contaminating the apple?", Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013
Discussion

- Interesting points?
- Open issues?
- Threats to validity?
- Links to other research directions?
- ...

That’s all folks!

“SPA challenges: how to get rid of worms before contaminating the apple?”,
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013

Thank you!

Credits for this invited talk go to...
Vittorio Cortellessa,
Antinisa Di Marco,
Davide Arcelli,
Martina De Sanctis,
Romina Eramo,
Alfonso Pierantonio,
Anne Koziolek,
Ralf Reussner

“SPA challenges: how to get rid of worms before contaminating the apple?”,
Invited talk @ Joint Kieker/Palladio Days, Karlsruhe, Germany, 27-29 November, 2013