Abstract
What is System Performance? Why should a software engineer have knowledge of the other parts of the system, such as the Hardware, the Operating System and the Middleware? The applications that he/she writes are self-contained, so how can other parts have any influence? This introduction sketches the problem and shows that at least a high level understanding of the system is very useful in order to get optimal performance.
content of this presentation

Example of problem

Problem statements
Image Retrieval Performance

Sample application code:

```java
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

alternative application code:

```xml
<screen 3*3>
    <row 1>
        <col 1><image 1,1></col 1>
        <col 2><image 1,2></col 2>
        <col 3><image 1,3></col 3>
    </row 1>
    <row 2>
        <col 1><image 1,1></col 1>
        <col 2><image 1,2></col 2>
        <col 3><image 1,3></col 3>
    </row 2>
    <row 3>
        <col 1><image 1,1></col 1>
        <col 2><image 1,2></col 2>
        <col 3><image 1,3></col 3>
    </row 3>
</screen 3*3>
```

application need:

at event 3*3 show 3*3 images
instantaneous

or
What If....

Sample application code:

```plaintext
for x = 1 to 3 {
  for y = 1 to 3 {
    retrieve_image(x,y)
  }
}
```

UI process

store

screen
More Process Communication

What If....

Sample application code:

```java
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

UI process

screen server

database

screen

9 * retrieve

update

version: 0.5
March 6, 2013
PINTROwhatIf2
Meta Information Realization Overhead

What If....

Sample application code:

```java
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

Attribute = 1 COM object
100 attributes / image
9 images = 900 COM objects
1 COM object = 80µs
9 images = 72 ms
I/O overhead

What If....

- I/O on line basis \((512^2\text{ image})\)

- . . .

Sample application code:

```c
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

\[9 \times 512 \times t_{I/O}\]

\[t_{I/O} \approx 1\text{ms}\]
Non Functional Requirements Require System View

Sample application code:

```plaintext
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

can be:
- fast, but very local
- slow, but very generic
- slow, but very robust
- fast and robust
- ...

The emerging properties (behavior, performance) cannot be seen from the code itself!

Underlying platform and neighbouring functions determine emerging properties mostly.
The performance and behavior of a function depend on realizations of used layers, functions in the same context, and the usage context.
## Challenge

<table>
<thead>
<tr>
<th></th>
<th>F &amp; S</th>
<th>F &amp; S</th>
<th>F &amp; S</th>
<th>F &amp; S</th>
<th>F &amp; S</th>
<th>F &amp; S</th>
<th>F &amp; S</th>
</tr>
</thead>
<tbody>
<tr>
<td>MW</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Functions & Services

### Middleware

### Operating systems

### Hardware

Performance = Function (F&S, other F&S, MW, OS, HW)
MW, OS, HW >> 100 Manyear : very complex

Challenge: How to understand MW, OS, HW with only a few parameters
Summary of Introduction to Problem

Resulting System Characteristics cannot be deduced from local code.

Underlying platform, neighboring applications and user context:
- have a big impact on system characteristics
- are big and complex

Models require decomposition, relations and representations to analyse.
The ASP™ course is partially derived from the EXARCH course developed at Philips CTT by Ton Kostelijk and Gerrit Muller.

Extensions and additional slides have been developed at ESI by Teun Hendriks, Roland Mathijssen and Gerrit Muller.