Vortrag:
Markus Kröpfl
Christian Spielvogel

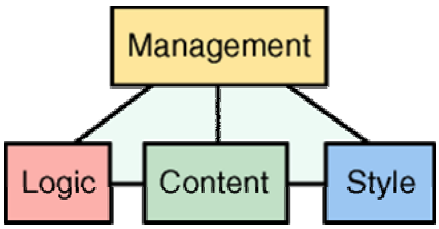
LV-Leiter:
DI Christian Timmerer

Cocoon

- What is cocoon?
  - Principally a dynamic content generation platform
  - Technically the same as ASP, JSP,PHP
  - Additional features with the help of XML
- Version 2.0, Developer
  - Stefano Mazzocchi, Start: November 1999

- Why XML – based?
  - Pure HTML – presentations insufficient
  - Content generated dynamically
  - Connection to external datasources like DB's

- Diffence to ASP – JSP:
  - Clear structuring of:
    - Data
    - Logic
    - Style



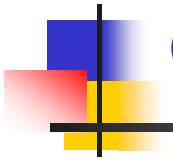
Cocoon

- Instead of ASP and JSP there are multiple output formats supported:
  - HTML
  - XML
  - Text
  - WML
  - PS
  - PDF
  - Own formats

- Various devices (browser, wml phone…)

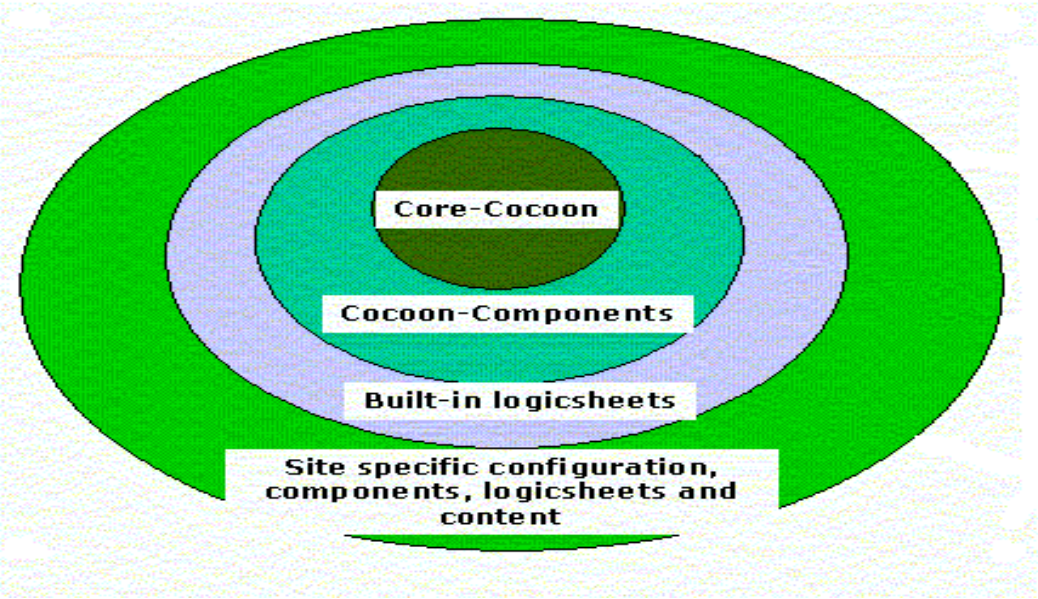
- Different browsers
  - Netscape
  - Internet Explorer
  - Opera

- Different readability
  - User readable (HTML) – machine readable (XML)



Cocoon

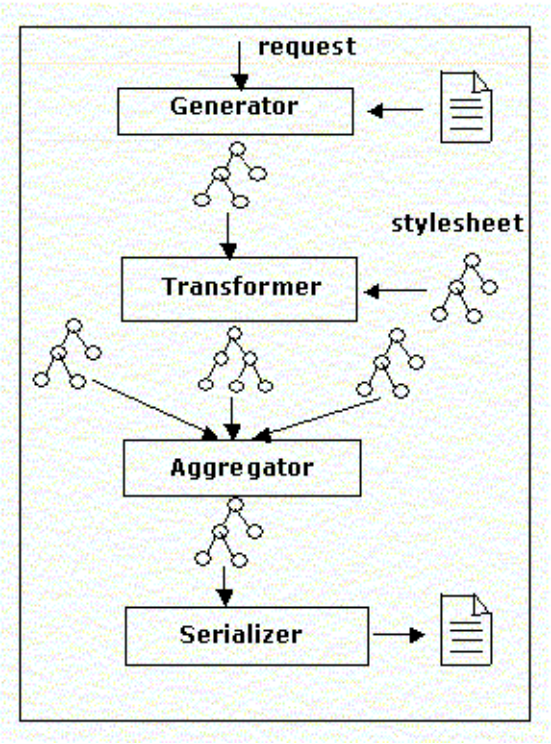
- How does it handle a request?
  - Generate XML Content from external source
  - Optionally transform it
  - Format it for output



Core - Cocoon

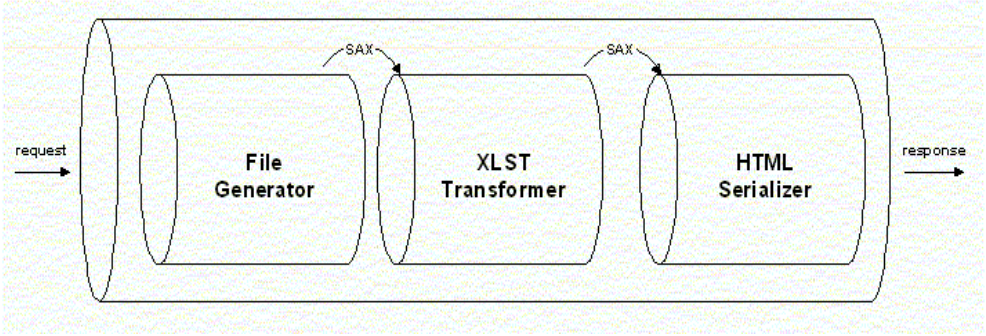
- The centre of cocoon is a servlet
- Tasks
  - Recieve requests
  - Process them
  - Generate a response

- Components
  - Generator
    - Accepts Input
    - Generates Actions
  - Transformer
    - Actions as Input
    - Layout as Output
  - Aggregator
    - Combines Output from Transformers
  - Serializers
    - Produce readable form



Cocoon - Pipelining

- Pipeline
  - Connects components among each other



- Request for „…/helloworld.html“

- 1. Step:
  - Matcher performs mapping from helloworld.html to helloworld.xml

- 2. Step:
  - Generator reads xml content from a file stored on the server

```
<?xml version="1.0"?> message text="Hello World"/>
```

Cocoon – Example HTML Request(2)

- 3. Step:
  - Transformer transforms XML actions to defined destination syntax

```
<?xml version="1.0"?>
<xsl:stylesheet
version="1.0„xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<head>
<title>A Message From Cocoon</title>
</head>
<body>
<xsl:apply-templates/>
</body>
</html>
</xsl:template>

<xsl:template match="message">
<h1><xsl:value-of select="@text"/></h1>
</xsl:template>
</xsl:stylesheet>
```

- 4. Step:
  - Serializer writes generated content to a file or sends it to the Servlet- Engine

```
<html>
<head>
<meta http-equiv="Content..Type" content="text/html;
charset=UTF..8">
<title>A Message From Cocoon</title>
</head>
<body>
<h1>Hello World</h1>
</body>
</html>
```

Cocoon – Different Generator Types

- FileGenerator (default)
  - Reads Input from XML File or URL
- HTMLGenerator
  - Reads input from HTML File from local filesystem or any URL
- DirectoryGenerator
  - Reads content from multiple files
- ImageDirectoryGenerator
  - Reads only image files

- XSLT Transformer(default)
  - Uses local stylesheet to produce desired layout
- I18n Transformer
  - Depending on the request, transformer determines destination language
- LogTransformer
  - Used for debugging
  - Prints received SAX events

Cocoon – Different Generator Types

- SQLTransformer
  - Performs and SQL query or invokes a stored procedure base on the SAX events
  - Translates queried data into XML
- FilterTransformer
  - Makes it possible to navigate within the queried recordset

- Define Pipeline Components
- XML Block for each Component
- Each Block has attribute src
  - Identifies the implementing Java Class
- Attribute name
  - Serves as reference within the pipeline

Cocoon – Sitemap(2)

- Define Pipeline Components
- 1. Part: Root element (obligatory)

```
<?xml version="1.0"?>
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
```

- 2. Part: Components

```
<map:pipelines>
<map:pipeline>
<map:match pattern="HelloWorld.html">
<map:generate src="HelloWorld.xml"/>
<map:transform src="Style.xsl"/>
<map:serialize/>
</map:match>
</map:pipeline>
</map:pipelines>
```