

# Cocoon Blocks

Daniel Fagerström  
danielf@nada.kth.se

# Motivation

- Cocoon is great but ...
- Monolithic
  - Huge download
  - Complicated configuration
  - Steep threshold
  - Few third party applications and components
  - ”Classloading hell”

# Blocks

- A plugin architecture is needed
- Designed by Stefano and the rest of the community 3+ years ago
- Compile time blocks for a few years, but no external contracts
- Much work and discussion, but less progress until the last half year
- Essentially back compatible, a new integration level: package and reuse applications

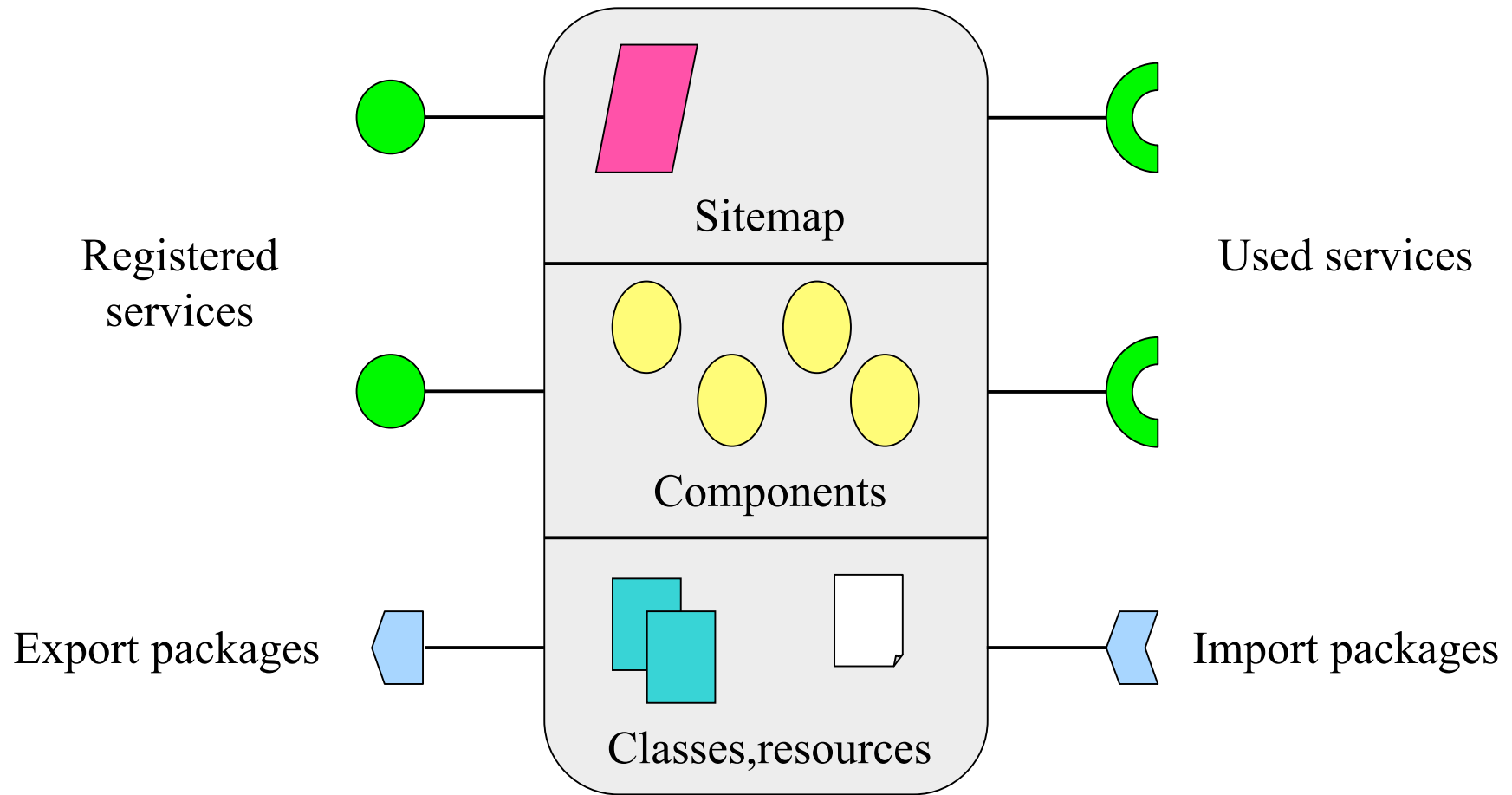
# Overview

- The big picture
- Architecture
  - Examples
- Current state and next steps

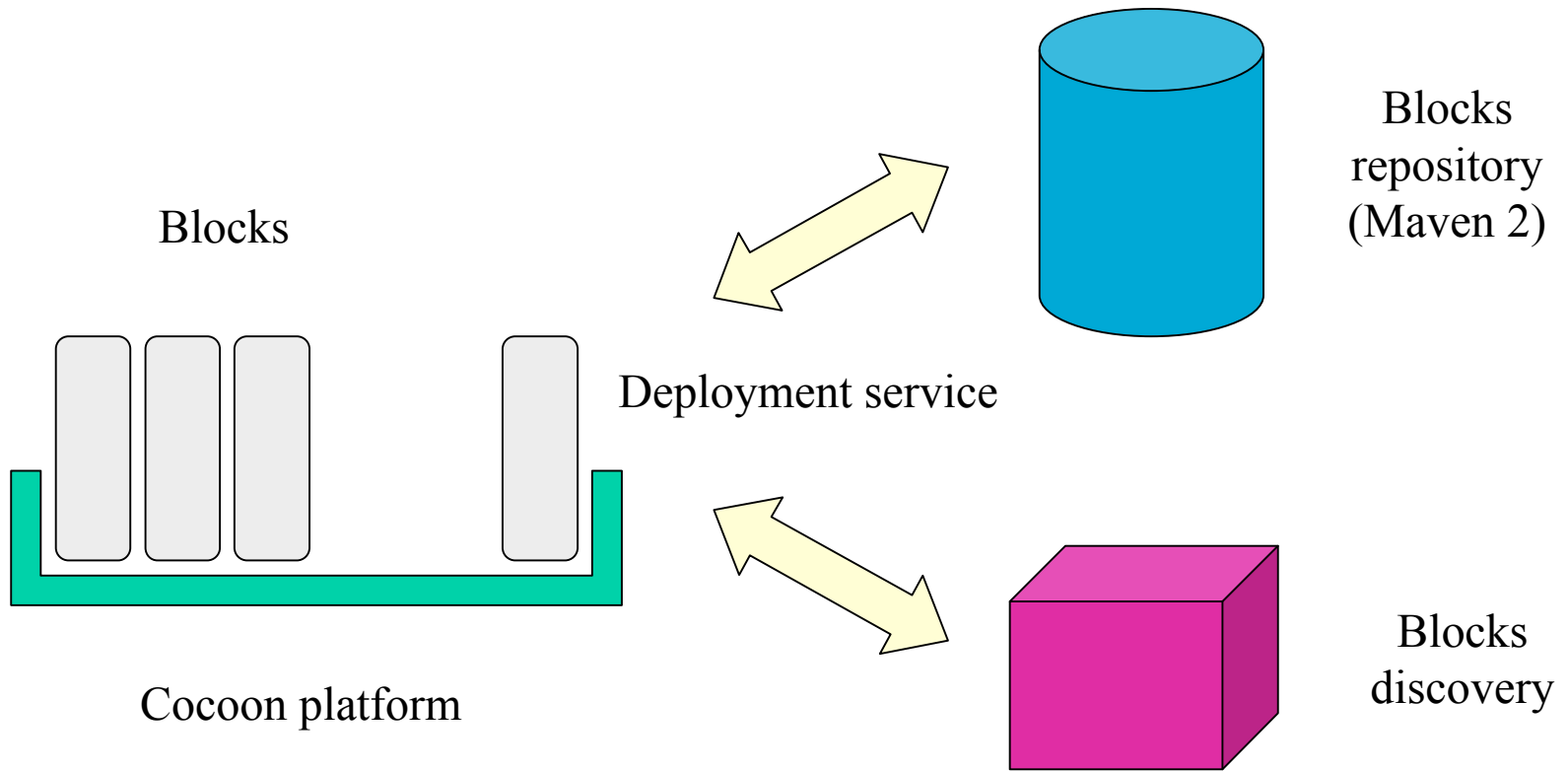
# What is a block?

- A packaged application (or part) containing:
  - Libraries and resources
  - Components
  - Sitemap functionality
- Configurable at deploy time
- Might depend on other blocks
- Isolated internals

# What is a Block?



# Deployment architecture



# Block Architecture

- Built upon OSGi (same as Eclipse 3+)
- A block is an OSGi bundle
  - Class loader isolation
  - Packaging format
  - Services
  - Security
  - Hot deployment possible

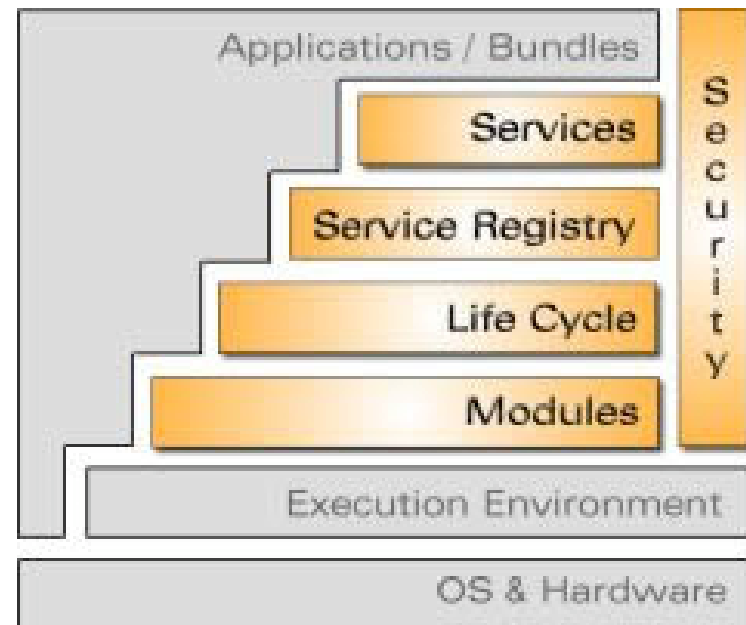
OSGi

# OSGi

- Standardized, component oriented, computing environment for networked services
- Since 1998
- 80+ members
- R3, soon R4
- 10+ implementations
- Smart phones, home automation, BMW 5, Eclipse

# OSGi Overview

- Secure execution environment
- Bundles (applications)
- Life cycle management
- Service architecture
- Standard services

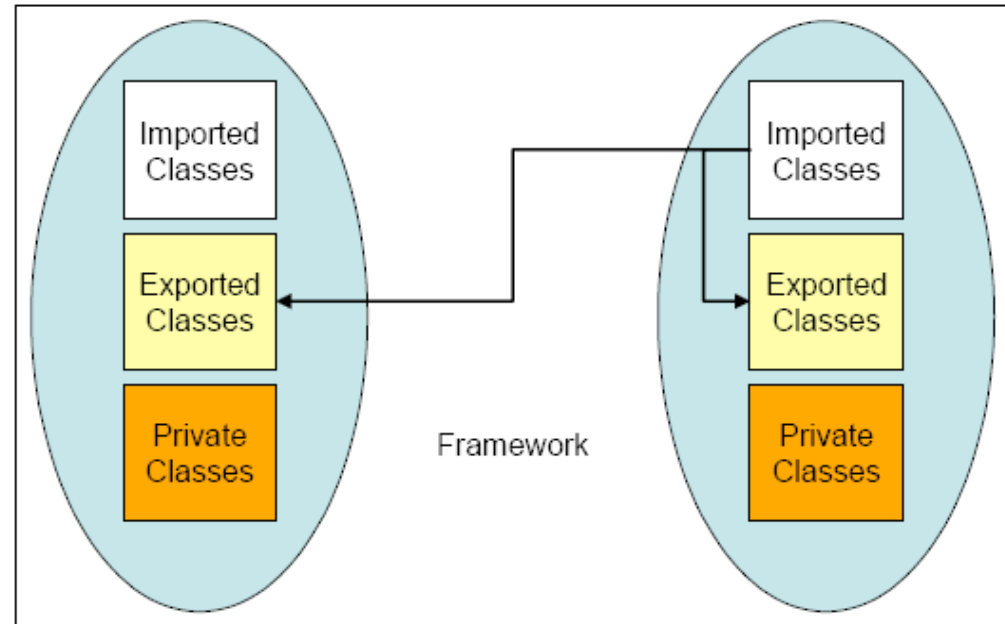


# Bundles

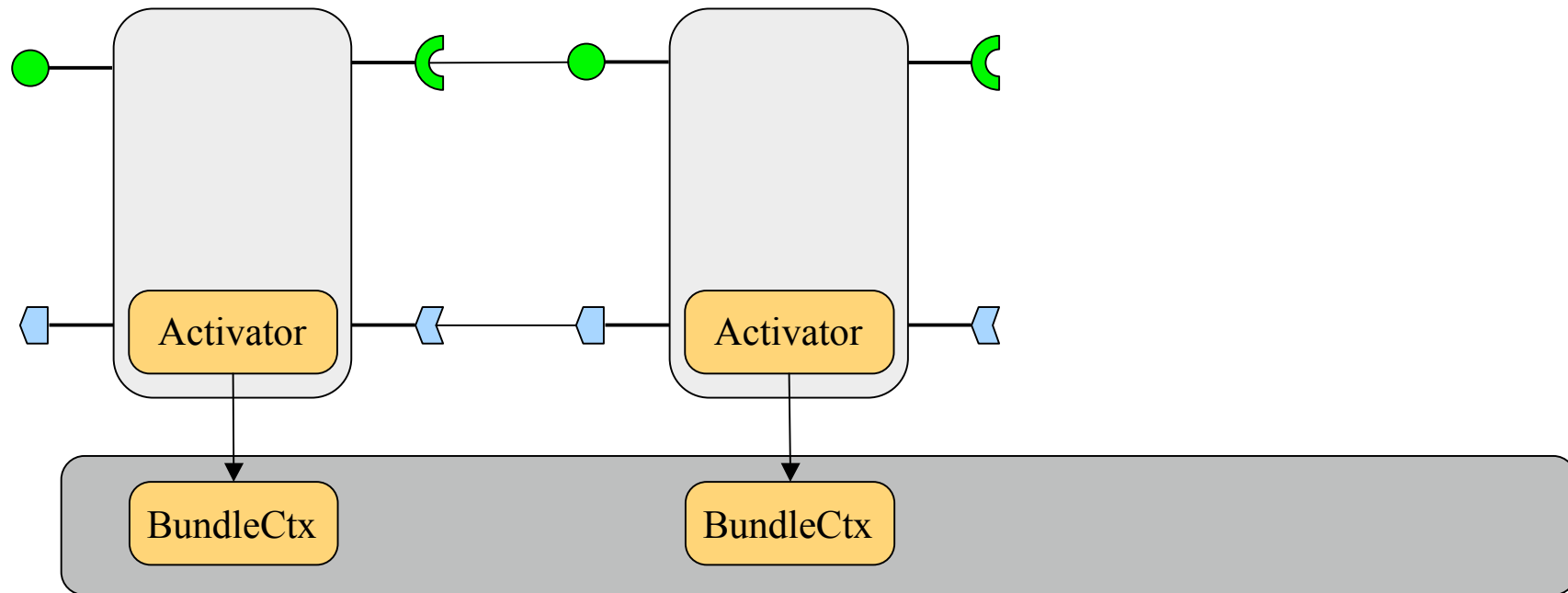
- Applications
- JAR containing
  - Compiled code
  - Resources
  - JARs that the code depend on
  - Meta information – manifest.mf

# Class sharing

- Import-Package
- DynamicImport-Package
- Export-Package
- Bundle-Classpath



# Activator



OSGi Framework



# Manifest.mf

```
Manifest-Version: 1.0
Bundle-Name: cocoon_servlet
Bundle-Version: 1.0.0
Bundle-Description: Cocoon servlet bundle
Bundle-Vendor: Apache
Bundle-DocURL: http://cocoon.apache.org
Bundle-ContactAddress: http://cocoon.apache.org
Bundle-Activator:
    org.apache.cocoon.service.servlet.impl.Activator
Bundle-Category: servlet
Import-Package:
    javax.servlet, javax.servlet.http,
    org.apache.cocoon.servlet,
    org.osgi.framework, org.osgi.service.http,
    org.osgi.service.log, org.osgi.util.tracker
```

# Blocks

# Block structure

```
myblock/  
  META-INF/  
    MANIFEST.MF      # bundle manifest  
  BLOCK-INF/  
    block.xml        # block configuration  
    myblock.xconf    # exported components  
    classes/  
    lib/  
    src/  
    sitemap.xmap     # block sitemap  
  resources/  
  ...
```

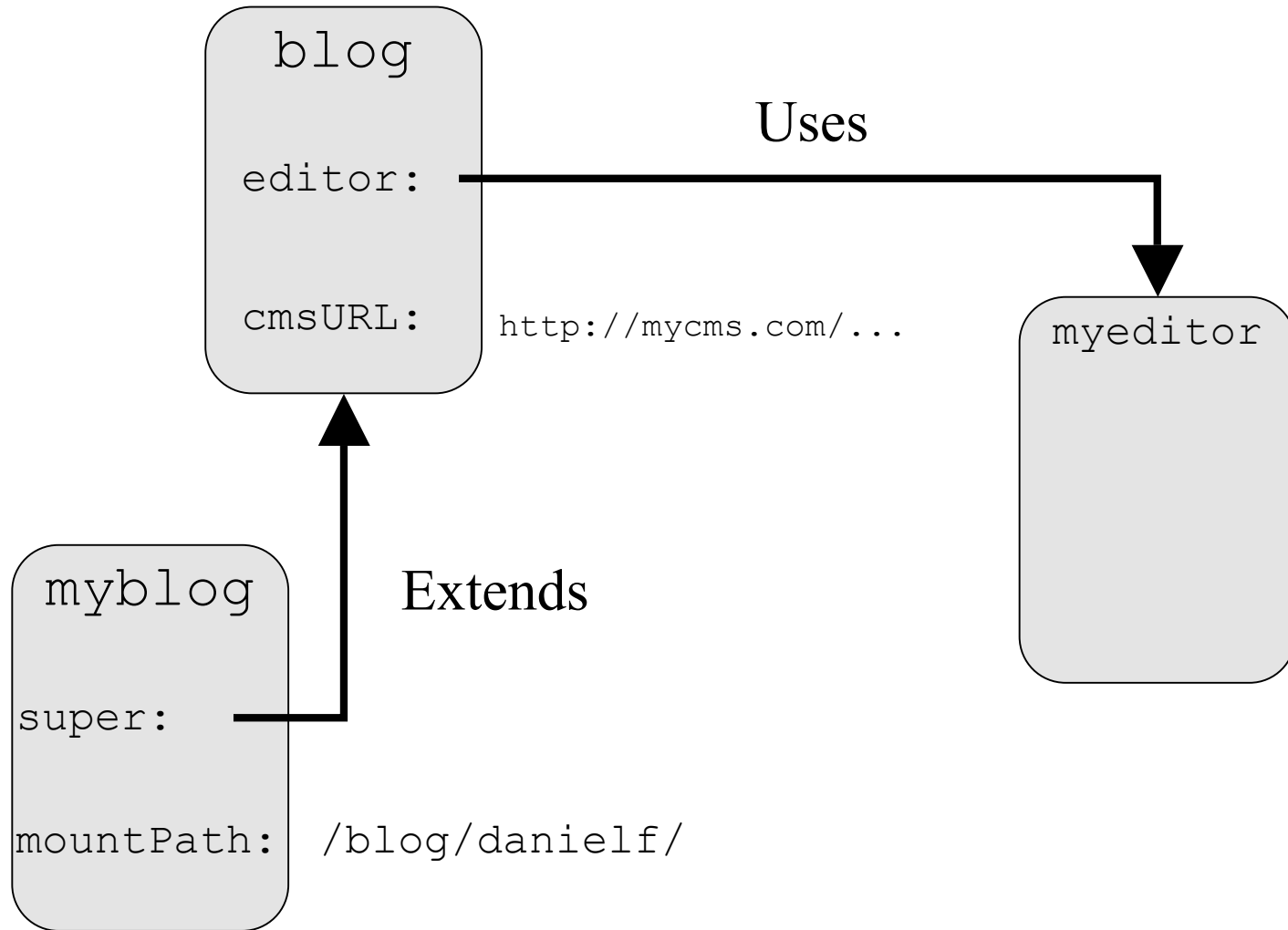
# Block configuration

```
<block xmlns="http://apache.org/cocoon/blocks/cob/1.0"
      id="http://cocoon.apache.org/blocks/blog-123">
  <name>blog</name>
  <sitemap src="sitemap.xmap"/>
  <components>
    <include src="blog.xconf"/>
  </components>

  <properties>
    <property name="cmsURL">
      <default>http://mycms.com/test</default>
    </property>
  </properties>

  <requirements>
    <requires
      interface="http://cocoon.apache.org/blocks/editor/1.0"
      name="editor"/>
  </requirements>
</block>
```

# Wiring



# wiring.xml

```
<wiring xmlns="http://apache.org/cocoon/blocks/wiring/1.0">

  <block id="blog-123" location="file:/blocks/blog-1.4.jar">
    <connections>
      <connection name="editor" block="editor-234"/>
    </connections>
    <properties>
      <property name="cmsURL"
        value="http://mycms.com/danielf"/>
    </properties>
  </block>

  <block id="editor-234" location="file:/blocks/editor-1.4.jar"/>

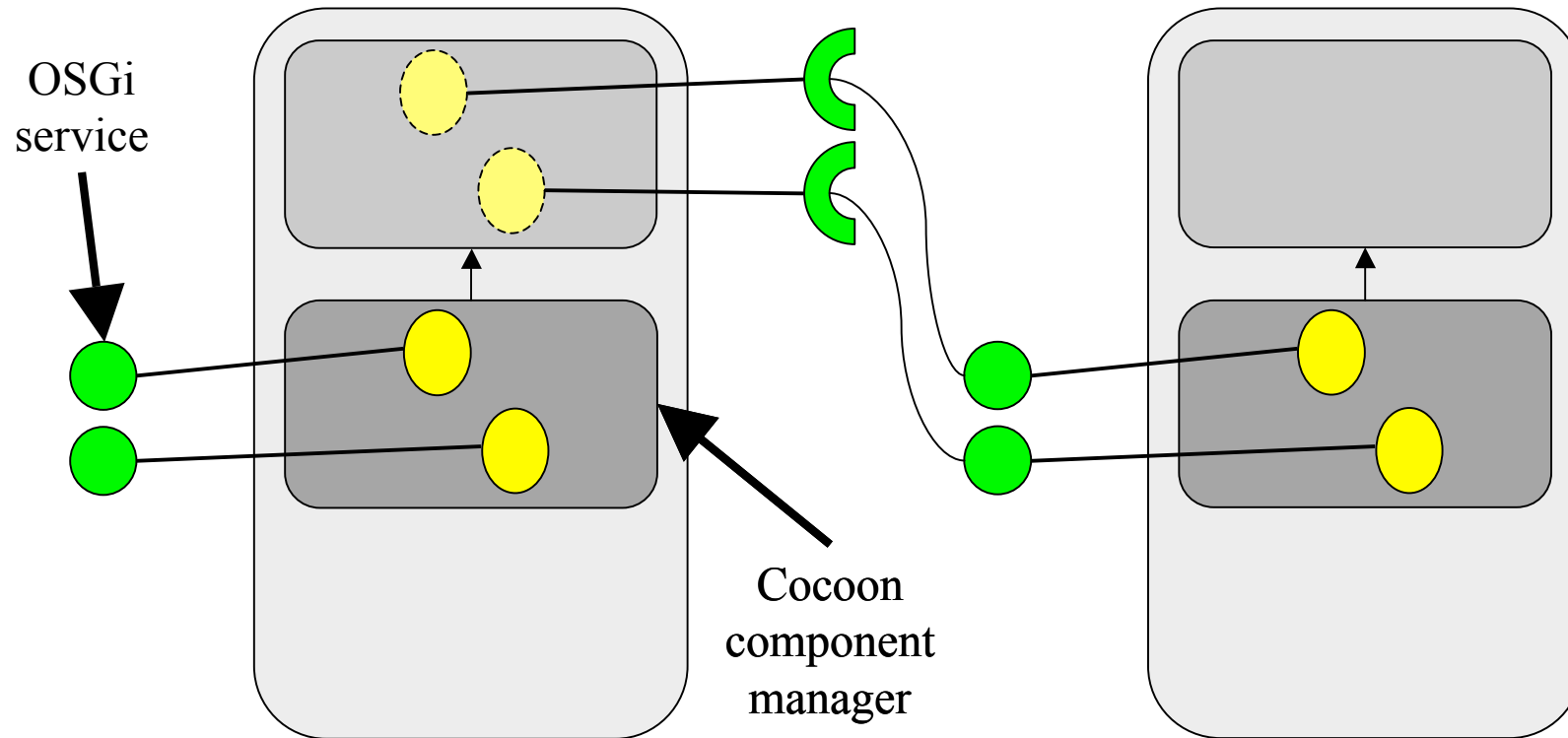
  <block id="myblog-345" location="file:/blocks/myblog/">
    <mount path="/blog/danielf"/>
    <connections>
      <connection name="super" block="blog-123"/>
    </connections>
  </block>

</wiring>
```

# Components in blocks

- Components only available from the own block and connected blocks

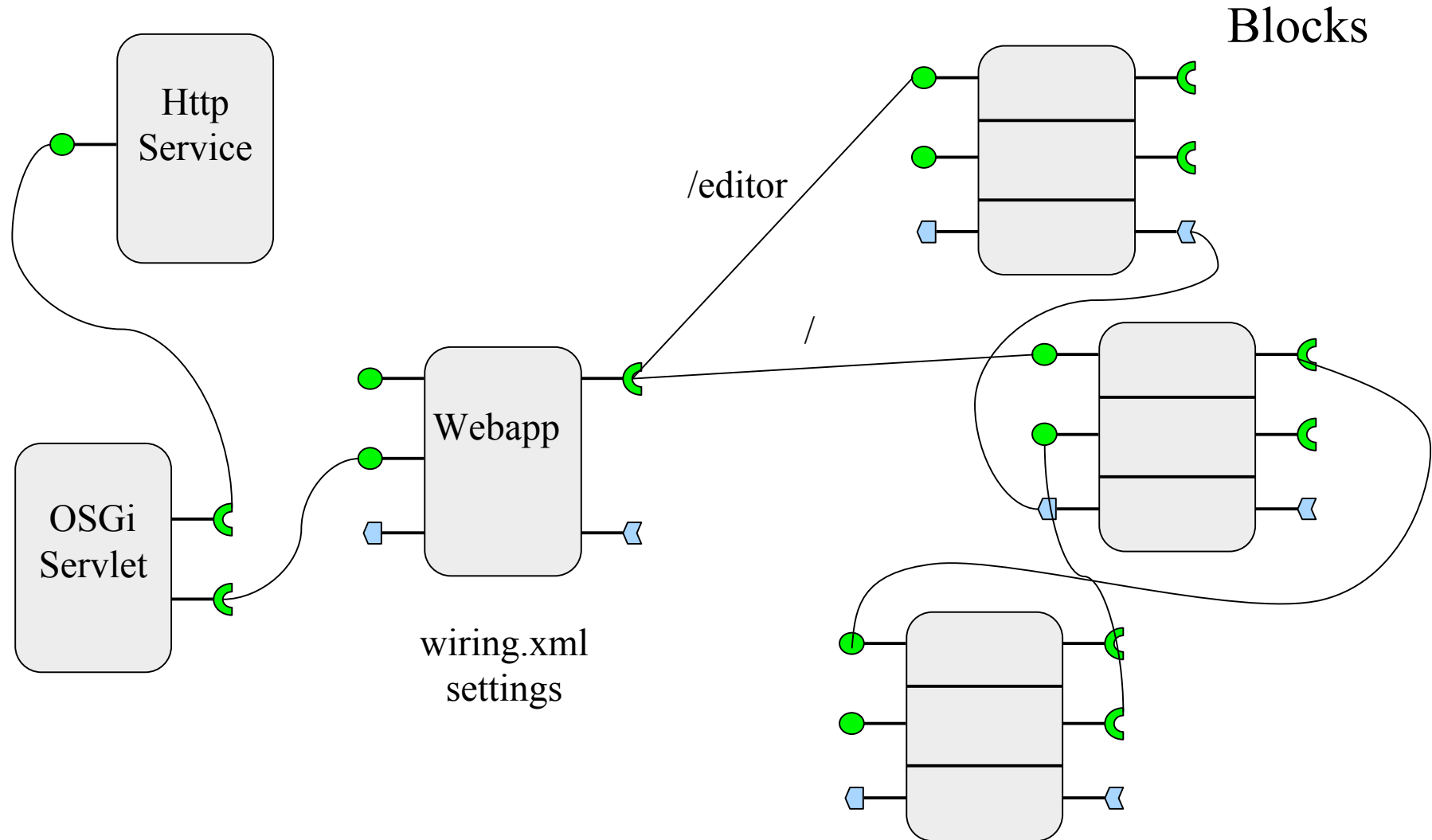
# Component bridge



# Sitemaps in blocks

- As usual
- Adds
  - Call sitemaps in connected blocks
  - Use block deploy time attributes
  - Extend blocks (with polymorphism)

# Block architecture



# Block protocol

**block:/foo.xml**

- root sitemap in current block

**block:./bar.xml**

- current sitemap in current block

**block:editor:/foo.xml**

- root sitemap in editor block

**block:super:/foo.xml**

- root sitemap in extended block

# Block properties, paths

**{block-property: cmsURL}**

- Block property in sitemap (input module)

**{ cmsURL }**

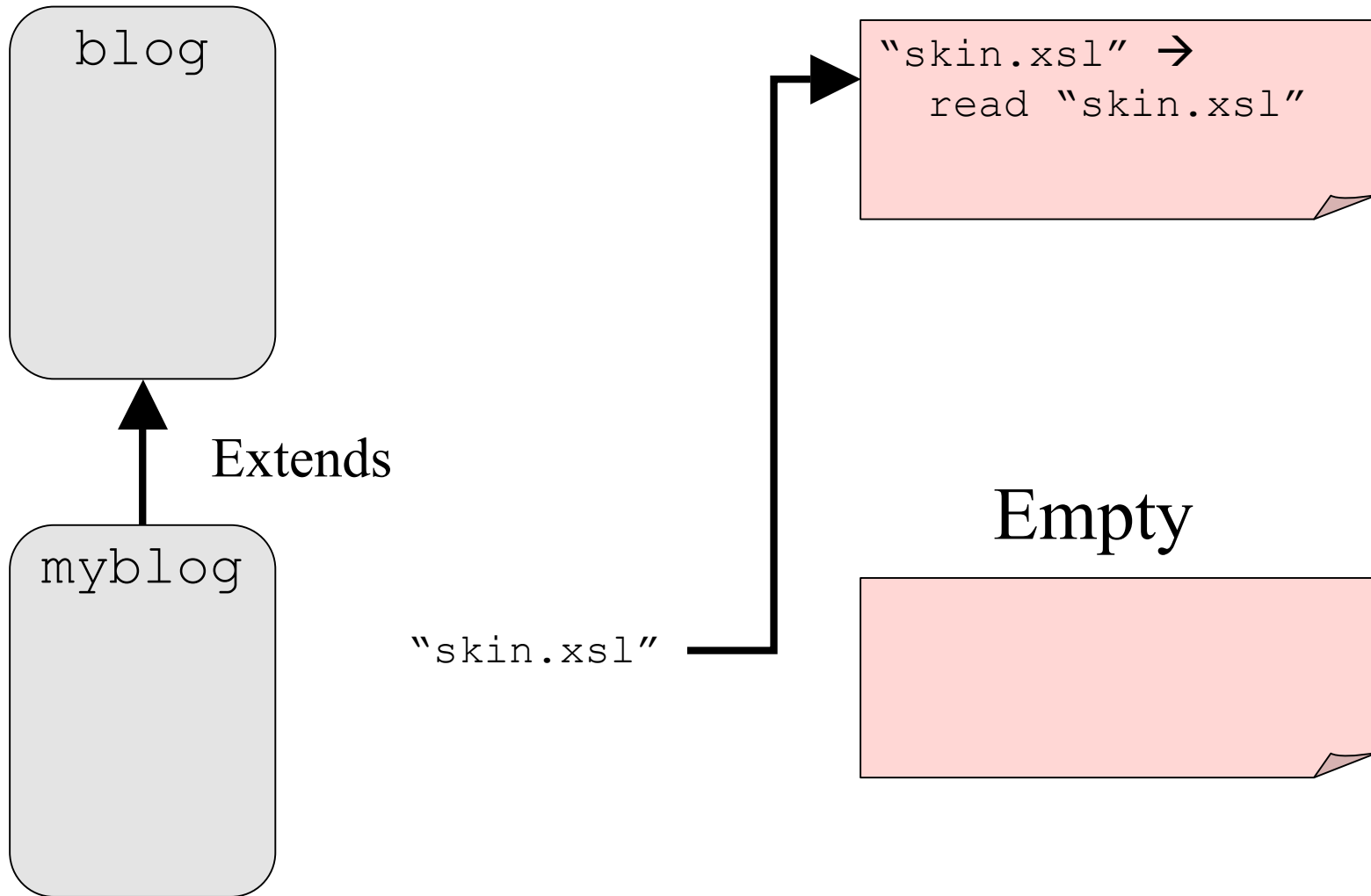
- Block property in component configuration

**{block-path:myblog:/start}**

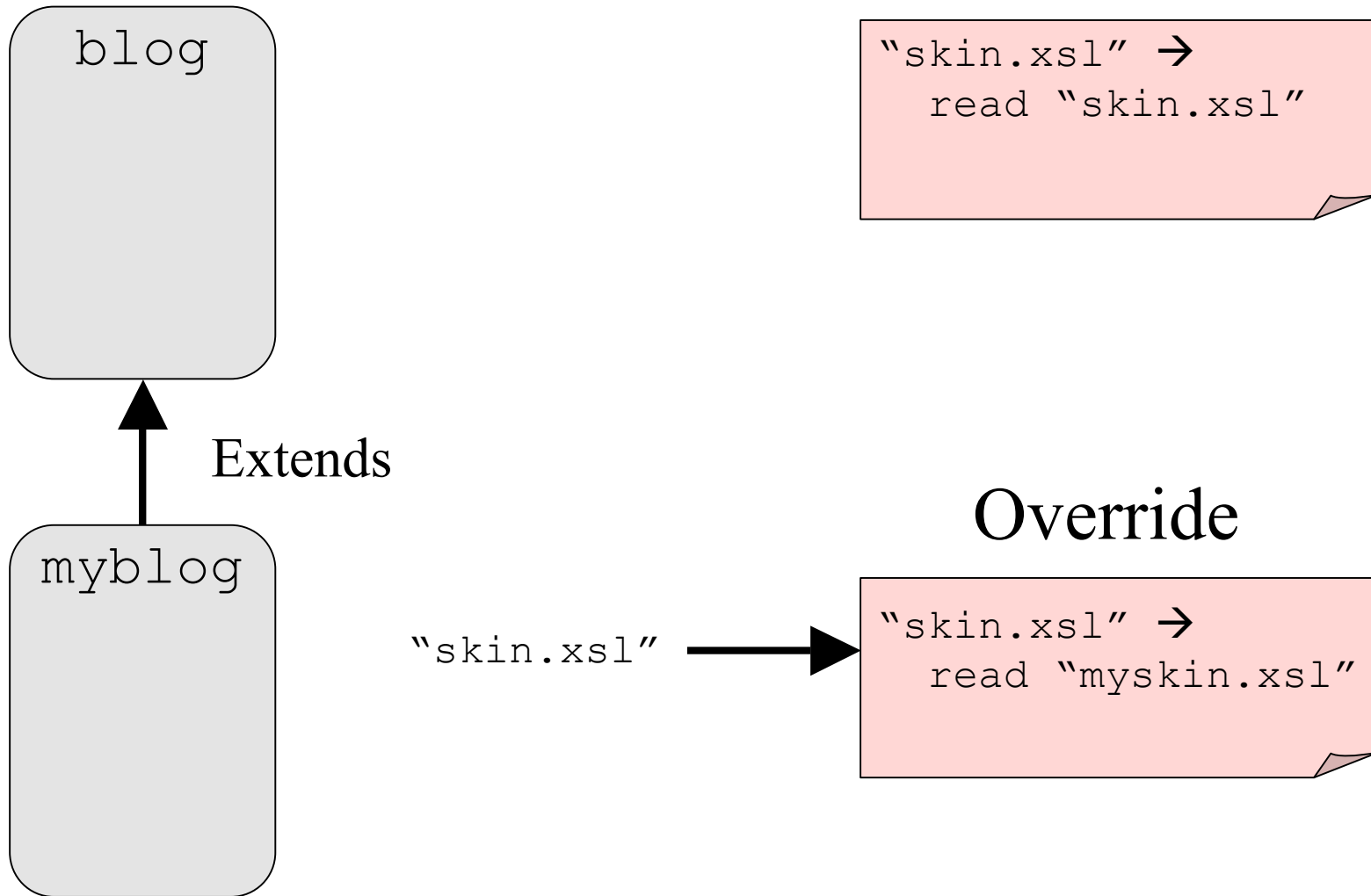
**--> /blog/danielf/start**

- “Absolutizes” block protocol URIs to mounted URIs, used in link transformer

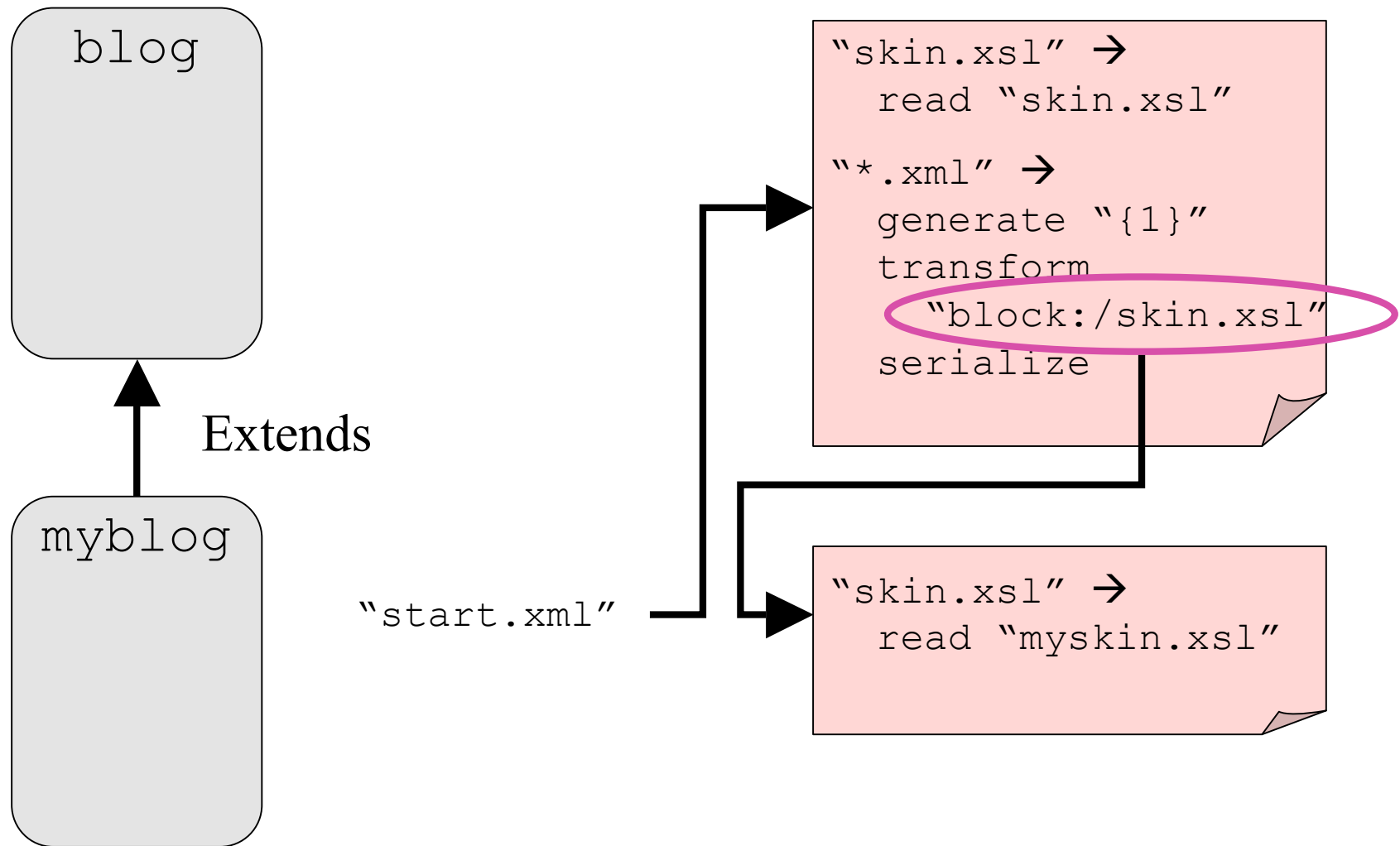
# Sitemap polymorphism



# Sitemap polymorphism



# Sitemap polymorphism



# Scenario

- Download blog block
- Deploy with parameters (or use default)
  - Test
- Create empty extension (skeleton generator)
  - Test
- Override some default or example rule
  - Test
- ...

# Summary

Blocks gives us:

- Binary application packages
  - Classes & resources
  - Components
  - Sitemap functionality
- Parameterizable applications
- Reusability by extension
- Dependency handling between applications

# Current state

- Cocoon runs under OSGi
- Sitemap blocks works
- Component bridge implemented
- Above parts are not yet integrated
- Maven 2 build on its way

# Next steps

- 2.2
  - Binary distribution of all blocks in M2 repository
  - Blocks architecture in experimental version, without OSGi
- 3.0
  - OSGi based
  - class loader isolation
  - partial hot plugablility