OpenOffice.org as a platform for developers

Mathias.Bauer@sun.com

Agenda

- Speaker introductions
- Features for development
- OpenOffice.org API architecture
- OpenOffice.org as a service provider
- Extending OpenOffice.org
- User Interface programmability
- AddOns
- OpenOffice.org AddOns (Demo)
- Q & A

Speaker introductions

- At Sun Microsystems:
  - Working on StarOffice/OpenOffice.org since 1995
  - Application Framework, 3rd party integration
  - Manager Software Engineering
    - Application Framework
    - UNO
    - SDK
    - UCB
- For OpenOffice.org:
  - Working on the project since its foundation
  - Application Framework Project Lead
  - Member of Engineering Steering Committee

Features for developers

- Support for many platforms and languages
  - 11 languages available for standard builds
  - More languages provided by the community
  - OOo 2.0: New tool support for localizing help content
- Stable language independent API based on UNO
  - Usable from Java, C++, OpenOfficeBASIC, Python
  - Support for OLE automation on Windows (f.e. VB, Delphi)
  - OOo 2.0: Language binding for CLI (.NET runtime)
  - More programming and scripting languages through a language agnostic scripting framework (f.e. JavaScript)
  - Implement own UNO services in C++, Java, Python
  - OOo 2.0: Much simpler use of UNO services

Features for developers (continued)

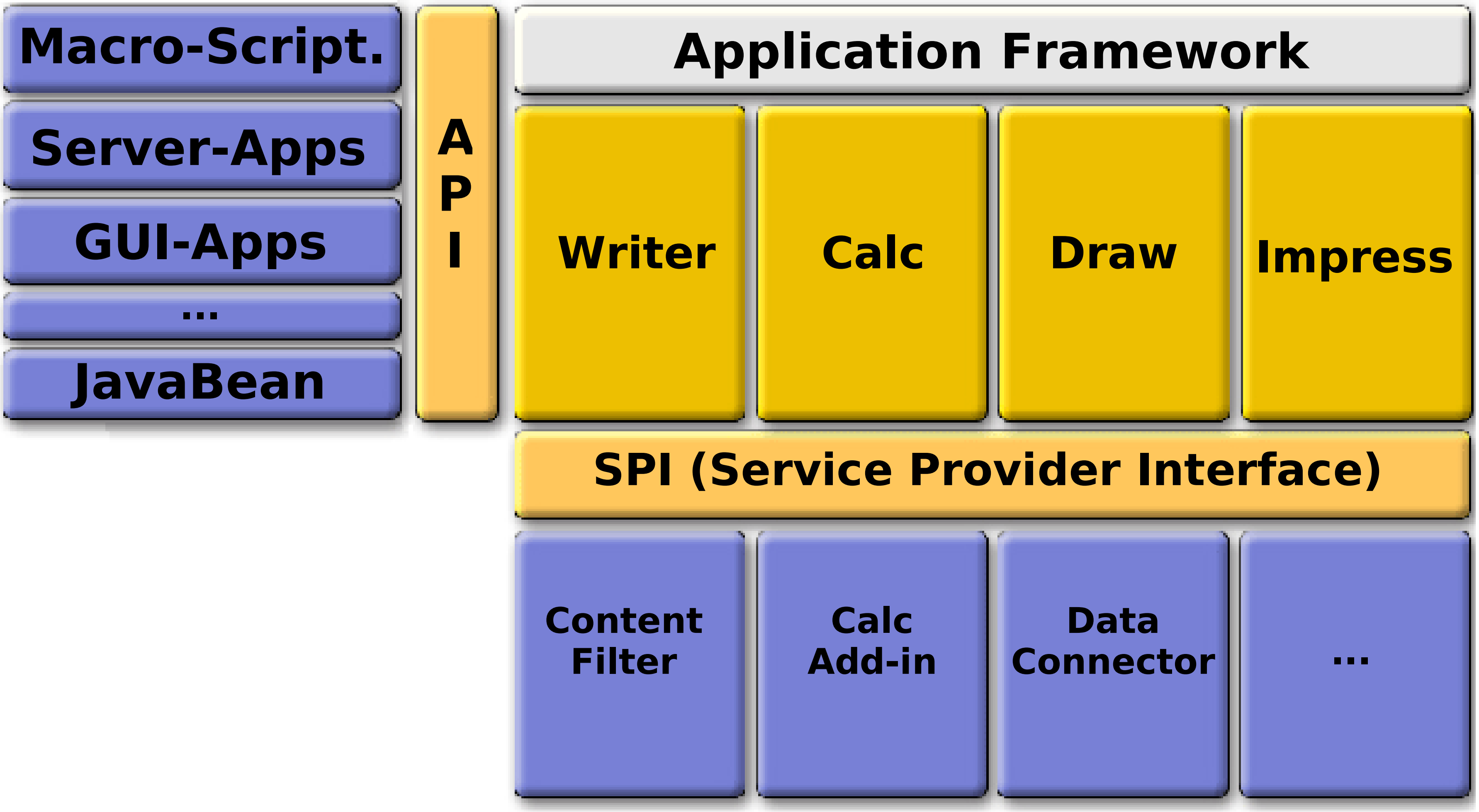
- UNO Component and type registry
  - Extend the OOo API by creating own types
  - Register own UNO components
- XML based expandable configuration
  - Read and write OOo configuration settings
  - Create own configuration templates and settings
  - Access through OpenOffice.org API
- Open, XML based file format
  - Add arbitrary content streams to packages
  - OOo 2.0: Store and read document variables

Features for developers (continued)

- Easy deployment of 3rd party components
  - Based on ZIP packages
  - Deployable on user or installation base
  - OOo 2.0: New Package Manager
    - Live Deployment
    - Package Manager GUI

- Deployable content:
  - Code: jar files, Basic modules, scripts, libraries (for multiple platforms in one file)
  - Configuration files (schema and data)
  - UNO types and services
  - UI elements (Popup menus, toolbars, images)
  - Document templates

OpenOffice.org API architecture



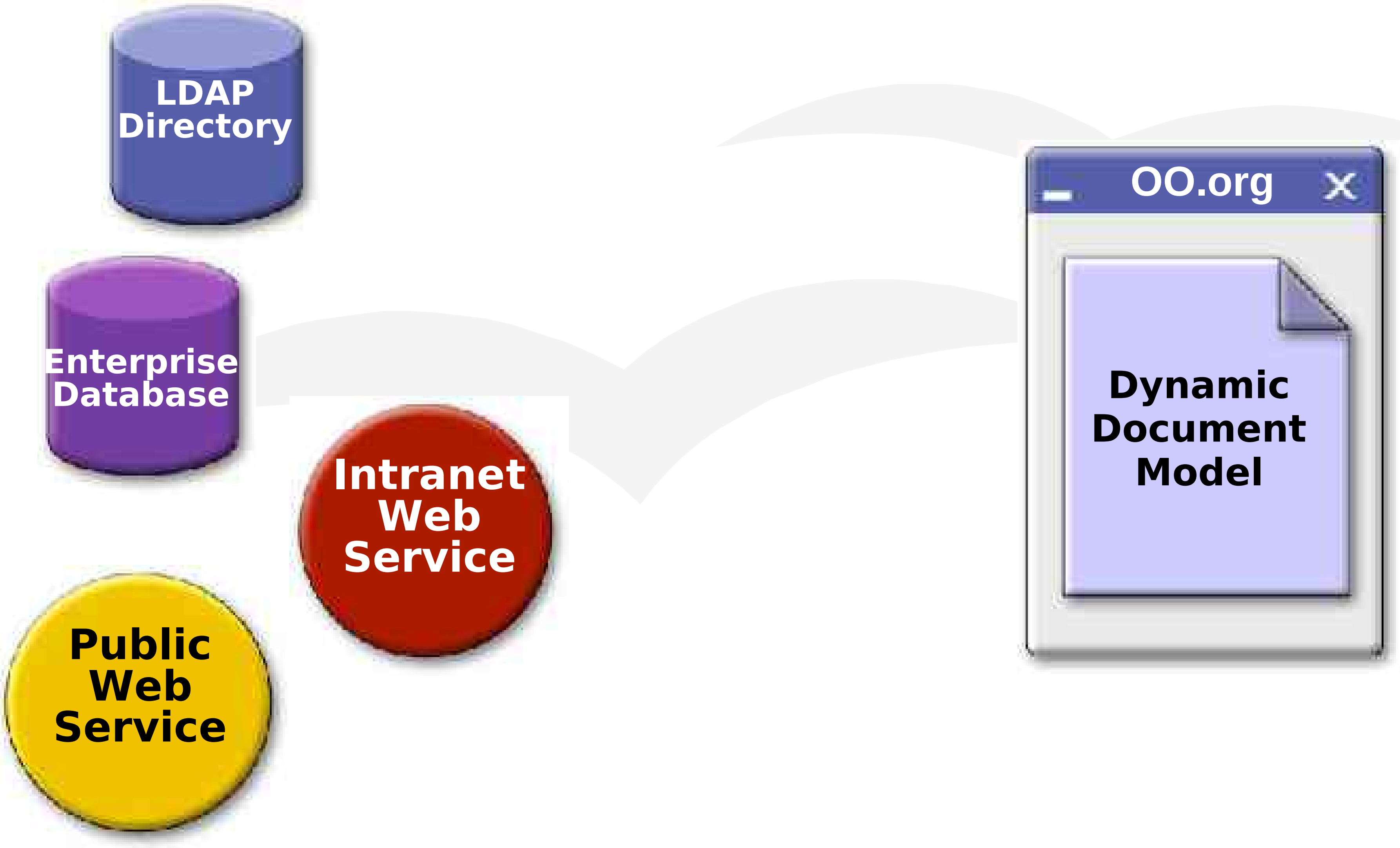
OpenOffice.org as a service provider

- Connect through UNO
  - Use any programming language supporting UNO calls
    - Connect from Python Scripts
    - Connect from Java Applets, Servlets, Java GUI Apps
    - Connect from Native Applications
    - On Windows: Connect via COM
    - OOo 2.0: On Windows: Connect via .NET
- Use existing GUI embedding technologies
  - ActiveX
  - OLE documents
  - Java Applet
  - Java (AWTContainers)
  - OOo 2.0: Netscape Plugin
  - OOo 2.0: Java Beans

Development Opportunities

- OpenOffice.org as content format gateway
  - Provide 1 format, publish to N
  - Server based document conversions
- OpenOffice.org as a point of service integration
  - Relational Databases
  - Address books, LDAP directory
  - Web services
  - File/Content management systems
- OOo 2.o: Workflow management
  - XForms support
  - Document variables
  - Digital signing of documents

Service Integration



LDAP Directory

Enterprise Database

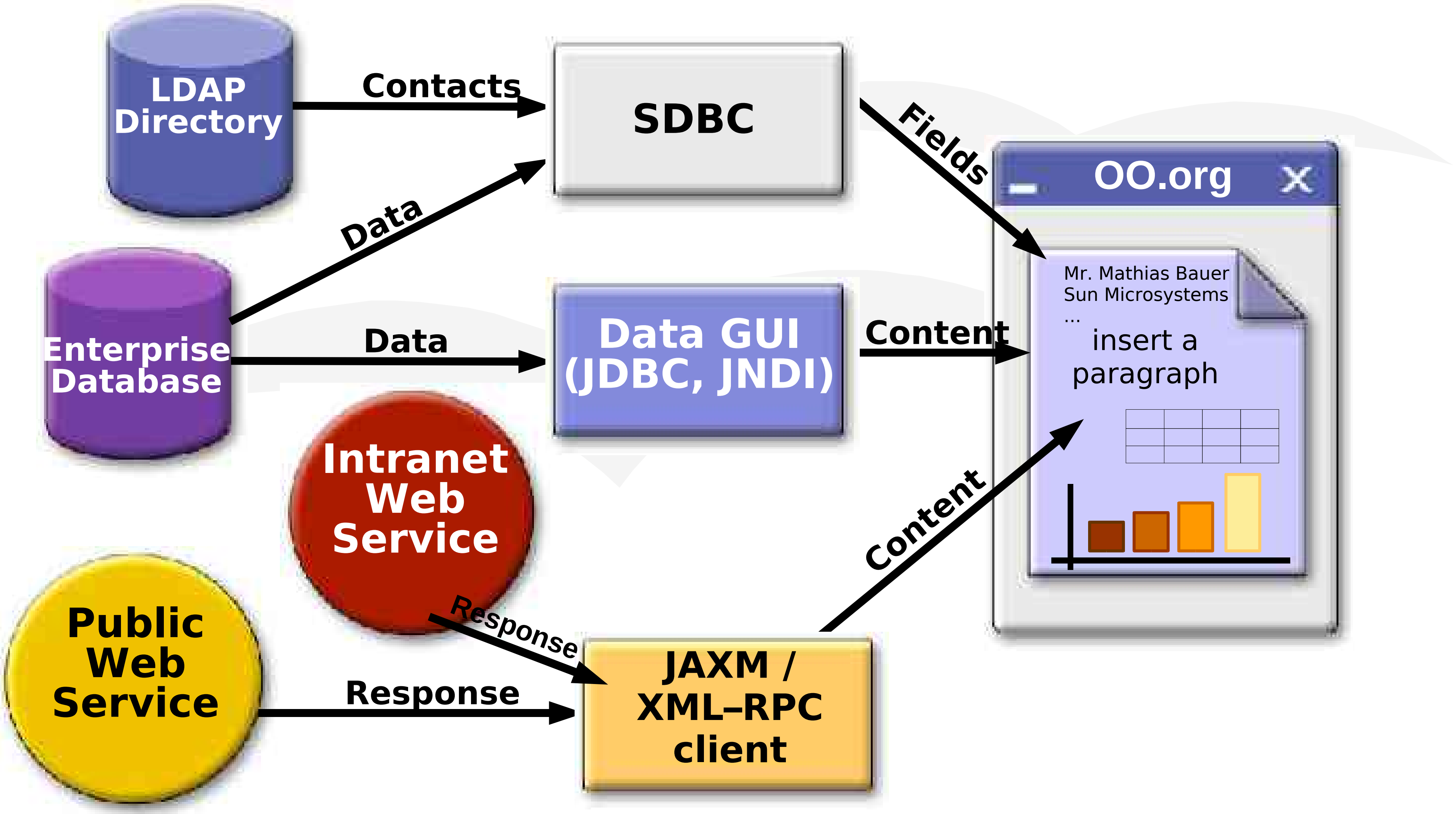
Intranet Web Service

Public Web Service

OO.org

Dynamic Document Model

Service Integration



LDAP Directory

Contacts

SDBC

Fields

Enterprise Database

Data

Data

Data GUI (JDBC, JNDI)

Content

Intranet Web Service

Response

Public Web Service

Response

JAXM / XML–RPC client

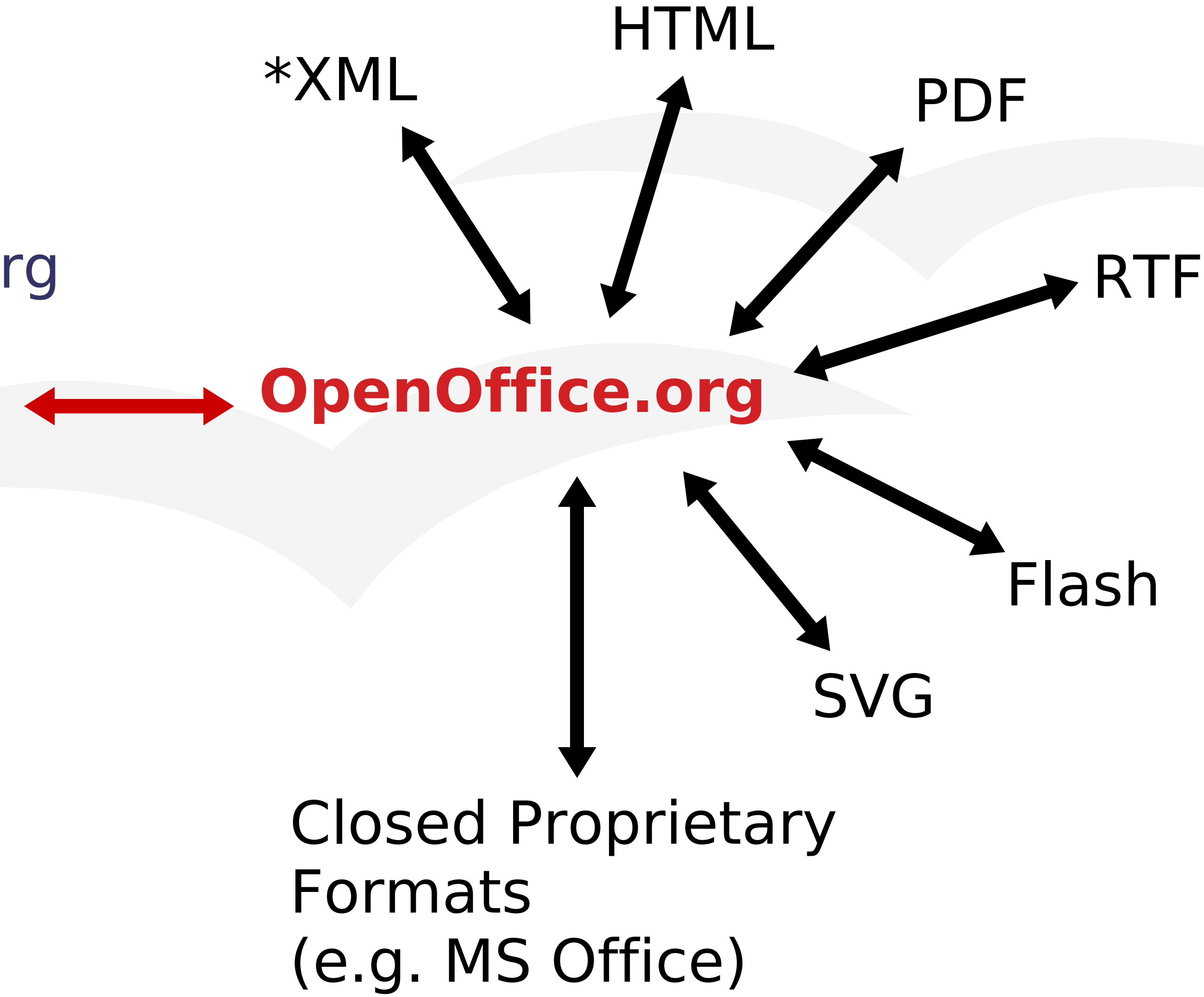
Content

OO.org

Mr. Mathias Bauer
Sun Microsystems
...

insert a paragraph

Content Format Gateway



OpenOffice.org
XML format

OASIS Open
Office XML
format

OpenOffice.org

*XML

HTML

PDF

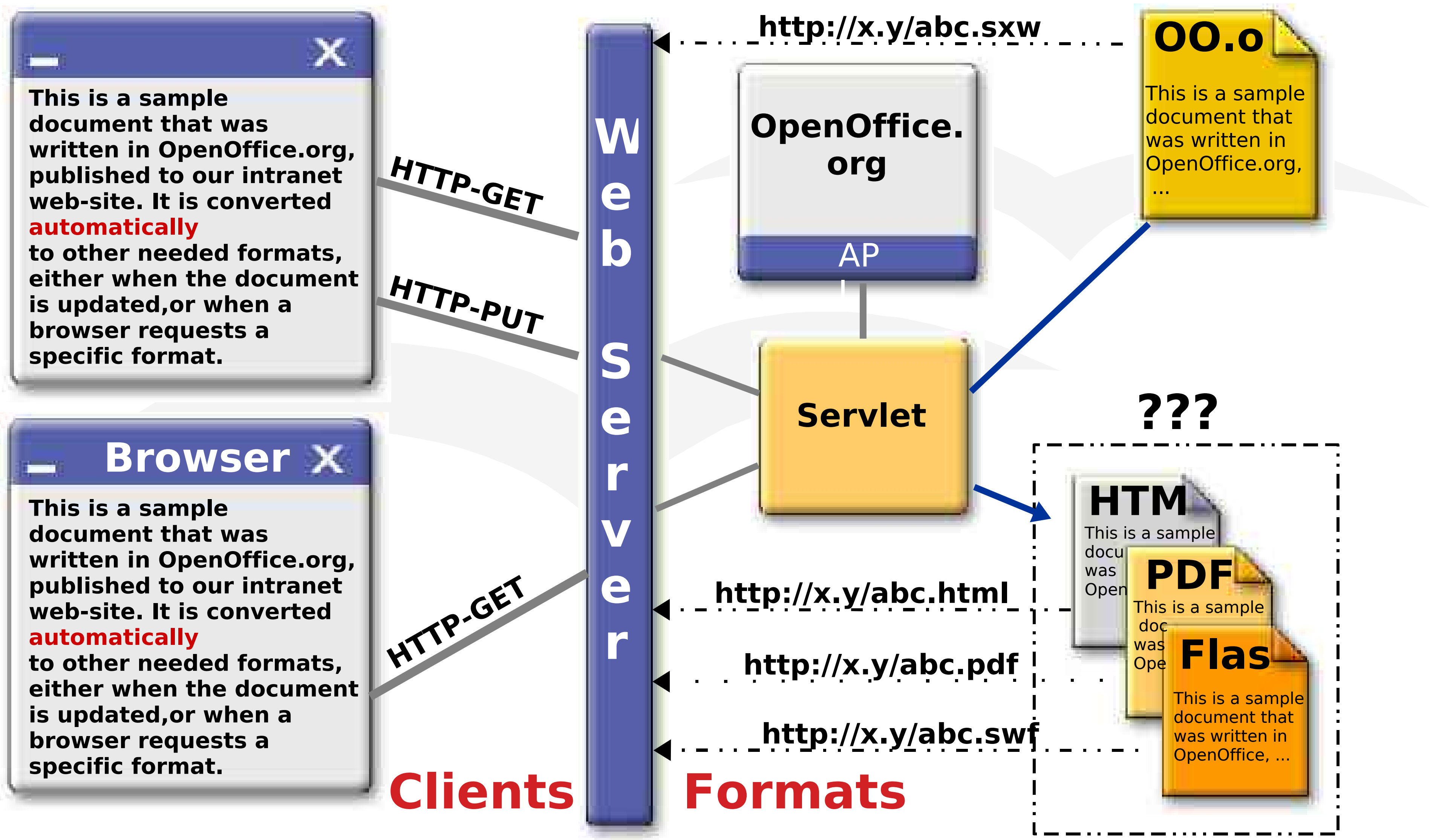
RTF

Flash

SVG

Closed Proprietary
Formats
(e.g. MS Office)

Content Format Gateway – use case



This is a sample document that was written in OpenOffice.org, published to our intranet web-site. It is converted **automatically** to other needed formats, either when the document is updated,or when a browser requests a specific format.

Browser

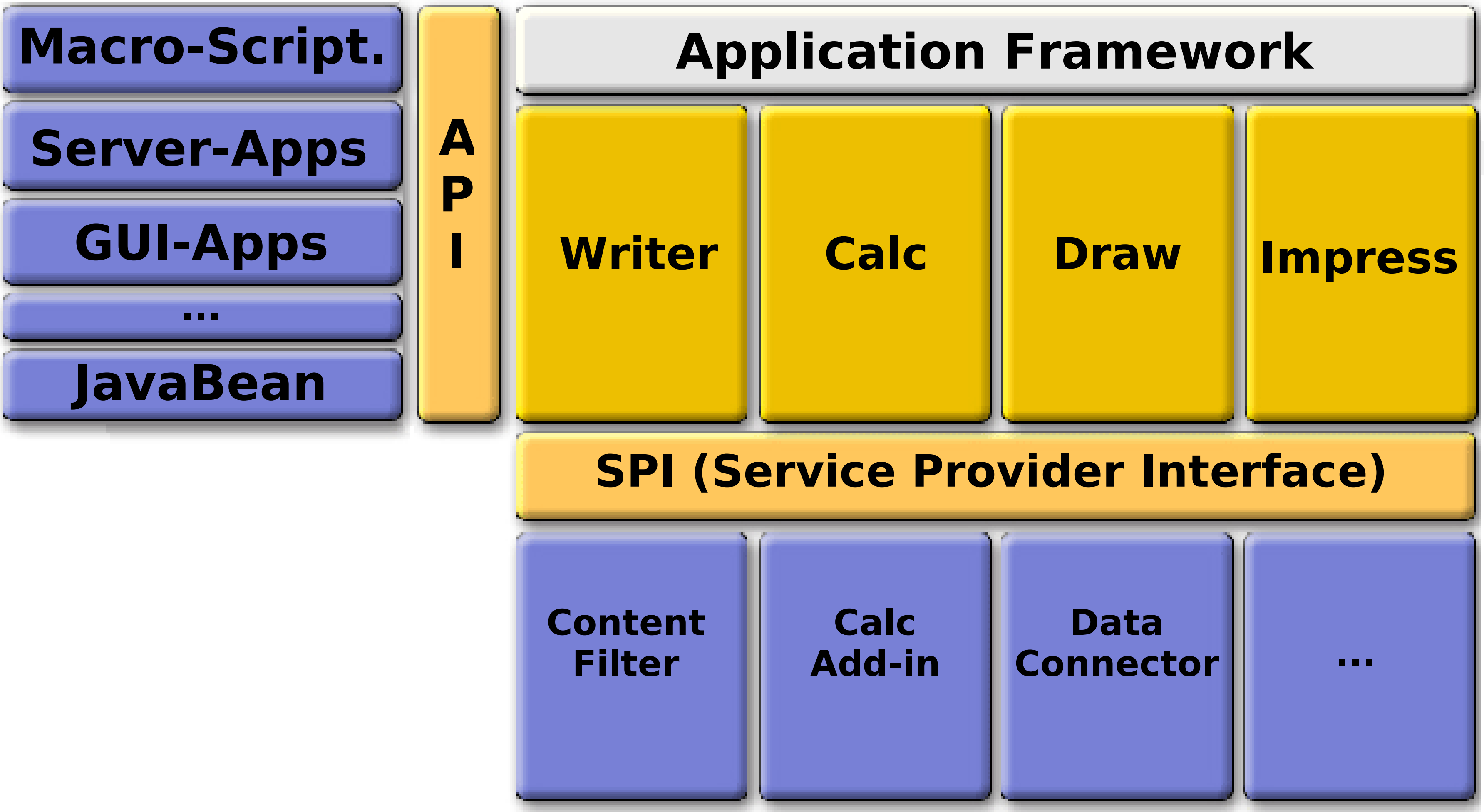
This is a sample document that was written in OpenOffice.org, published to our intranet web-site. It is converted **automatically** to other needed formats, either when the document is updated,or when a browser requests a specific format.

HTTP-GET
HTTP-PUT
HTTP-GET
Web Server
OpenOffice.org
API
Servlet
http://x.y/abc.sxw
OO.o
This is a sample document that was written in OpenOffice.org, ...
???
HTM
This is a sample docu was Open
PDF
This is a sample doc was Ope
Flas
This is a sample document that was written in OpenOffice, ...
http://x.y/abc.html
http://x.y/abc.pdf
http://x.y/abc.swf
Clients
Formats

Extending OpenOffice.org

- Adding external components to OpenOffice.org
  - Provide UNO components (see Developers Guide)
  - Bridge to other technologies (COM, Java etc.)
    - Provide UNO wrappers for existing components
    - Call non-UNO based components
  - Build your components with the OpenOffice.org SDK
  - Currently possible languages: C++, Java, Python
  - Develop platform independent components
    - Use Java
    - Provide libraries for several platforms in one package
    - Use SDK libraries for platform support
  - Replace existing components or add new ones

OpenOffice.org API architecture

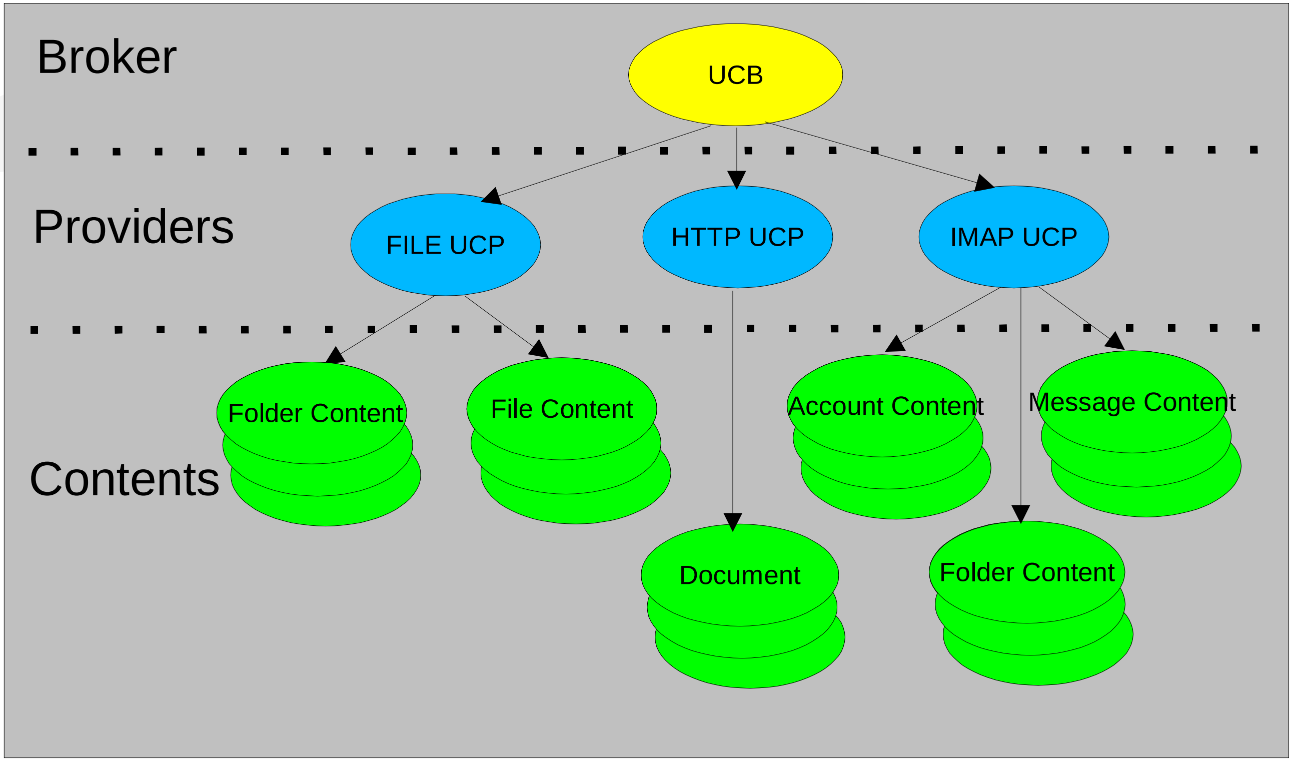


Integrating external components

- Registration of new Service Providers
  - Data Connectors: new data connections
  - Calc Add-ins: new functions in Calc formulas
  - Content Providers: new file sources
  - Document import/export filters
  - OOo 2.0: Impress Shapes
- Exchange current implementations
  - Example: FilePicker and FolderPicker service
  - Spell Checker
  - Replace existing Service Providers
- Call external components from the GUI
  - Add new GUI elements
  - Redirect calls from existing GUI elements

Content Access: UCB

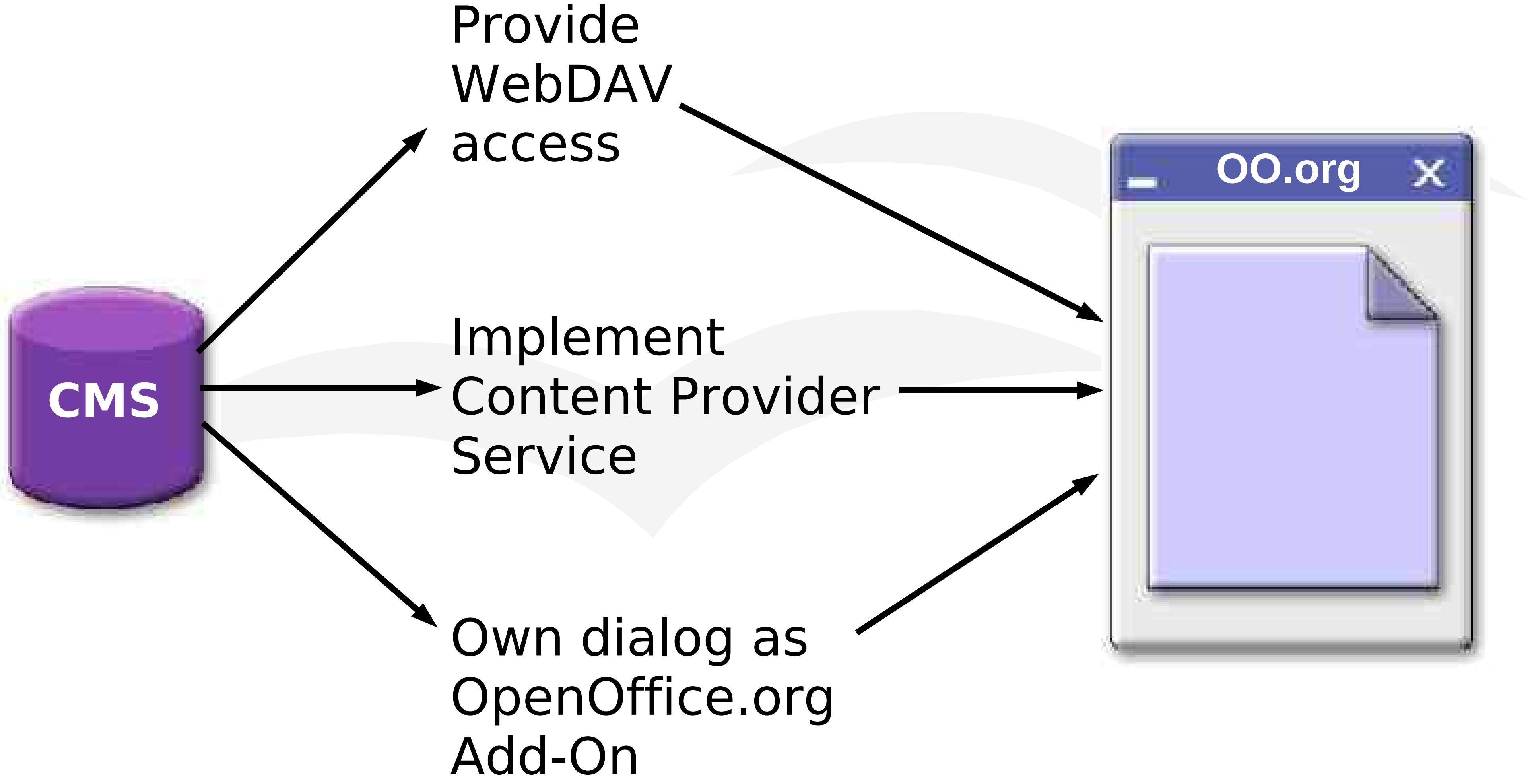
- Makes content hierarchies accessible through URLs (“http”, “ftp”, “file”, “webdav”)
- Extend it by defining new protocol schemes
- Implement access to “folders” and “files” following the UCP API



Special: Hierarchy UCP

- Implements a UCP on a registry backend
- Default registry backend available working on the OOo configuration
- Can map arbitrary hierarchies
  - Fixed hierarchies: provide configuration file
  - Moderately changing hierarchy: same as above, but provide service that updates configuration on demand
  - Dynamic hierarchy: provide your own backend

Integration with CMS



Provide WebDAV access

Implement Content Provider Service

Own dialog as OpenOffice.org Add-On

User Interface: Programmability

- Generic OpenOffice.org UI is XML based
  - Menubar
  - Toolboxes
  - Keyboard shortcuts
  - Event bindings

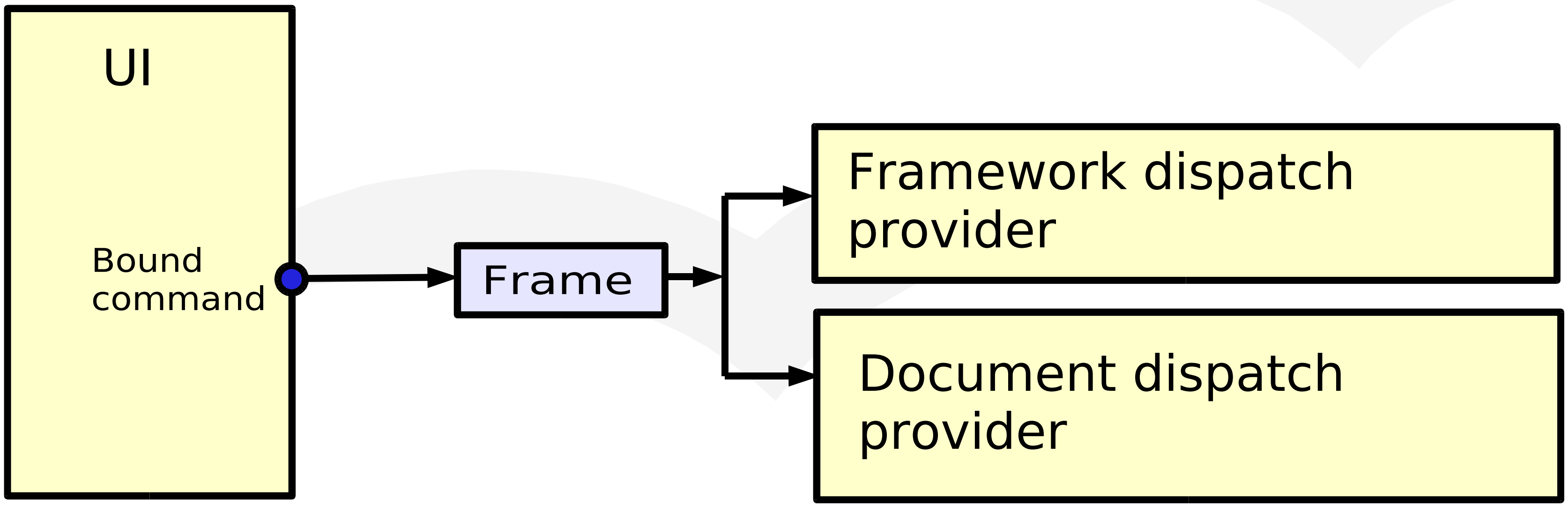
- Modification of UI elements
  - Modify the XML files directly
  - Add toolbars as many as you like with new XML files
  - Add toolbars and menubar popups through AddOn configuration files
  - OOo 2.0: API for accessing UI elements at runtime
  - OOo 2.0: Replace menu entries by registered popups

User Interface: Dispatching

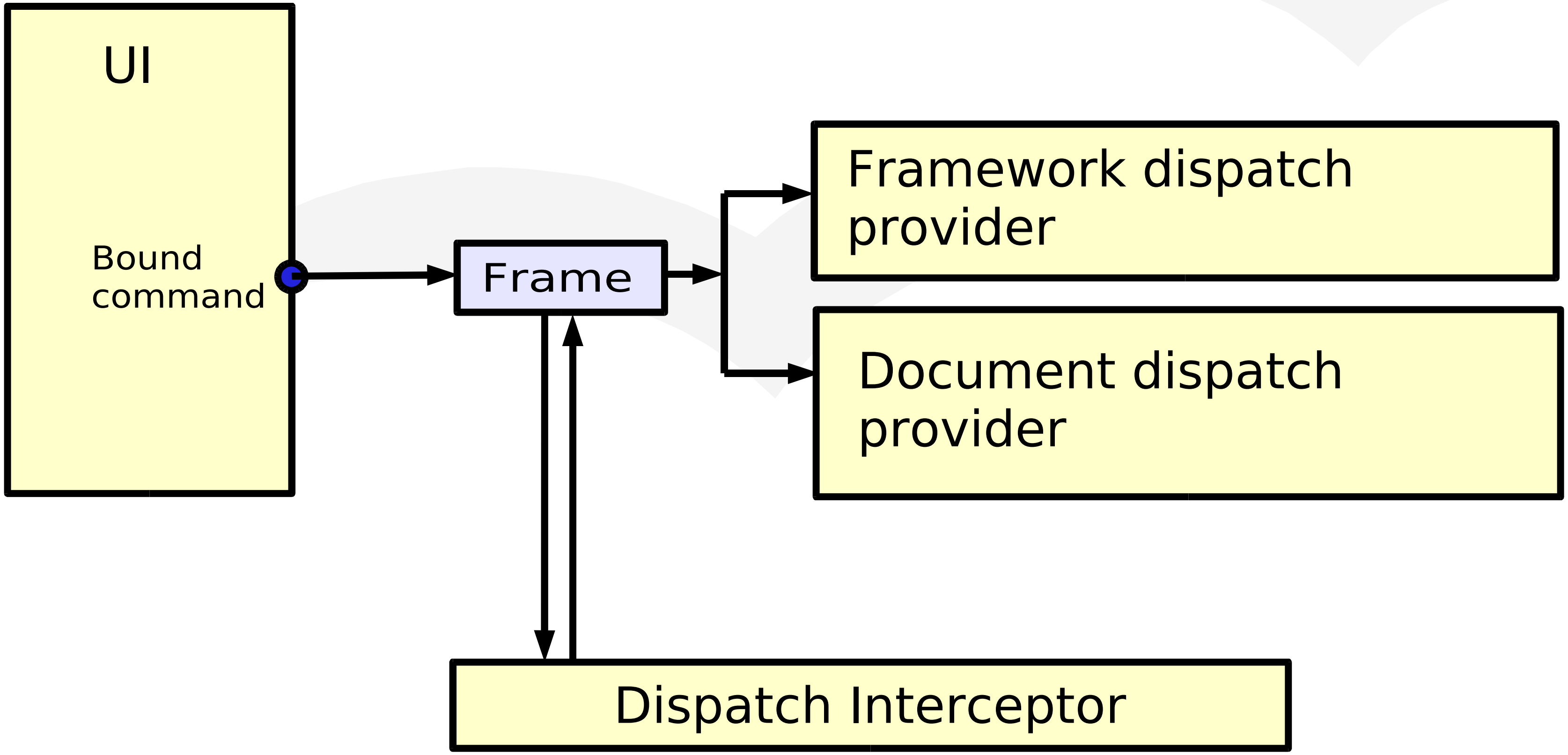
- Modify or extend the UI through Dispatch Objects
  - Disabling of selected commands: Configuration means
  - Redirecting of internal commands: Dispatch Interception
  - Introducing new commands: Protocol Handlers

- Functionality is described by commands
  - Commands are strings with URL like syntax
  - OpenOffice.org searches for command handlers implementing the DispatchProvider service
  - Dispatch Providers create Dispatch objects that are bound to the User Interface element
  - Dispatch objects send status information
  - Dispatch objects execute commands
  - Dispatch objects are very lightweight objects

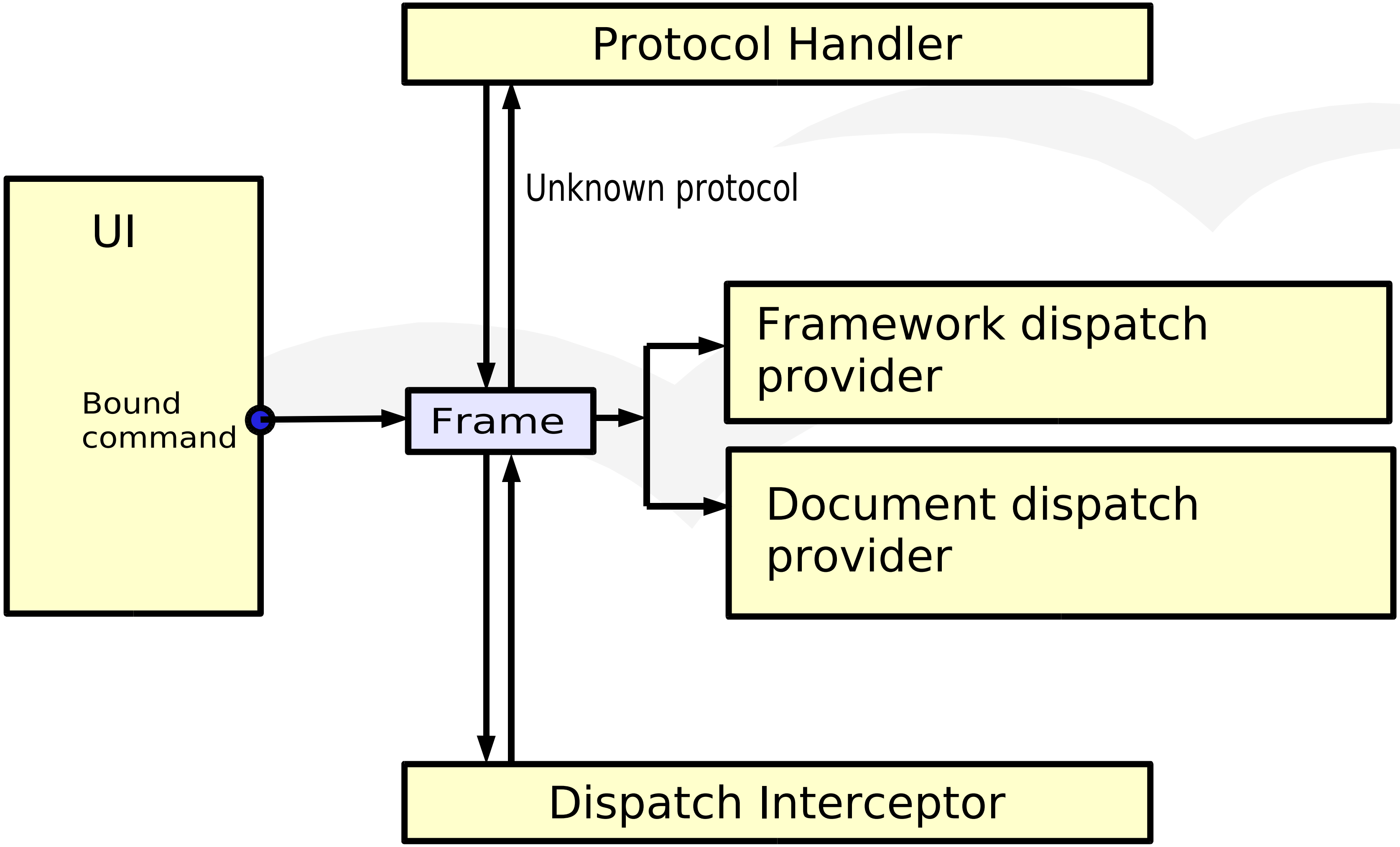
Default Dispatching process



Intercepted dispatching process



Complete dispatching process



Protocol Handler

Unknown protocol

UI

Bound command

Frame

Framework dispatch provider

Document dispatch provider

Dispatch Interceptor

OpenOffice.org AddOn concept

- Chapter 4.7.3 in the OOo 1.1 Developers Guide
- Sample code for Demo Addon in OOo 2.0 SDK
  - Java
  - C++

- Basic elements:
  - GUI configuration files for menu, toolbars
  - Images for toolbar and menu entries
  - Protocol Handler
  - Dispatch Interceptor, Context Menu interceptor
  - Event Listeners
  - Jobs
  - Pack up everything into a zip file