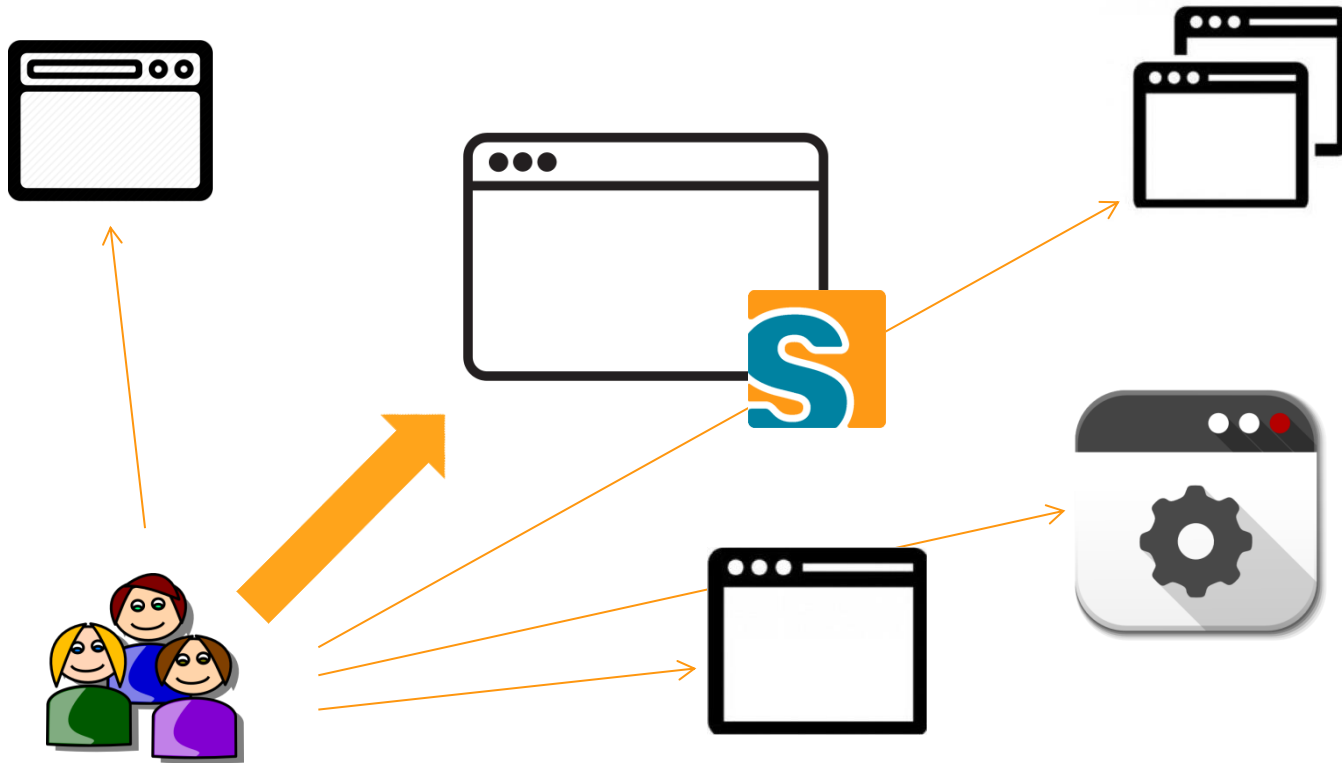


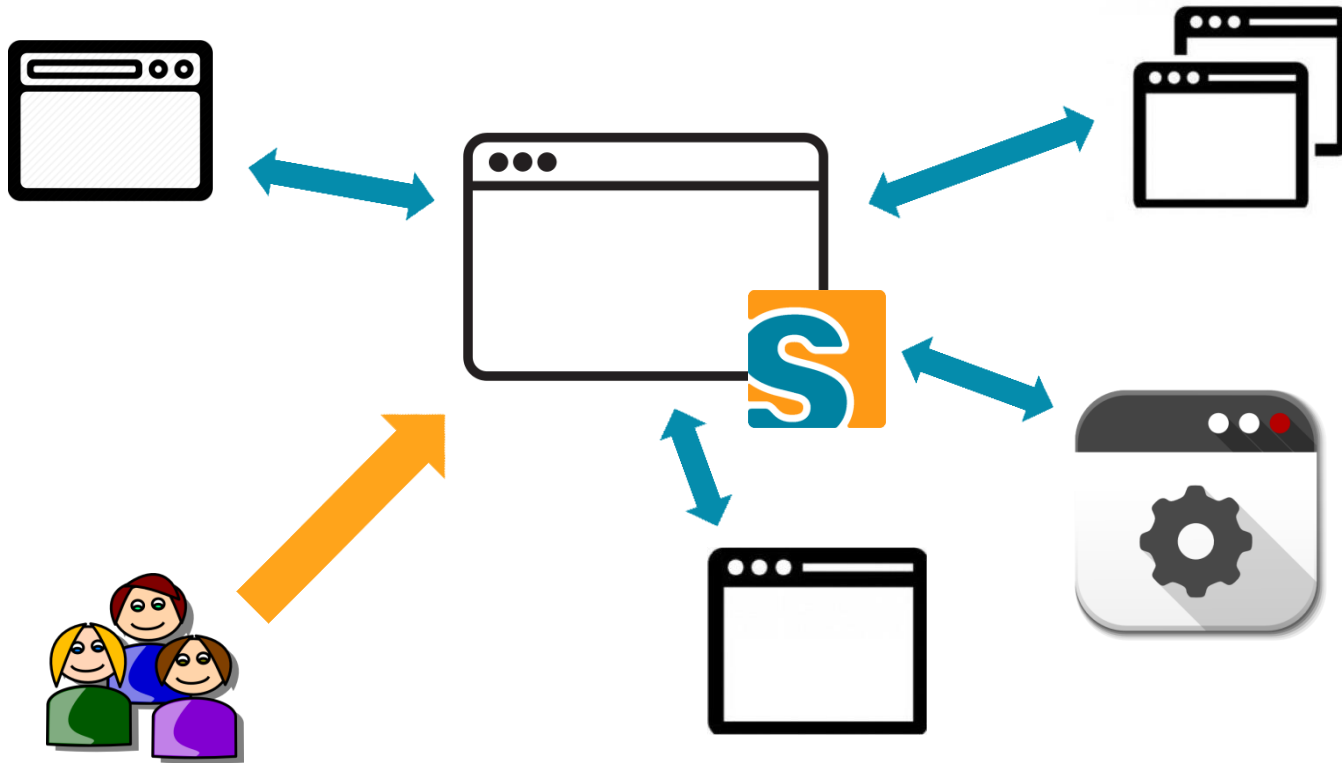
Integration of 3rd Party Apps

Scout UGM 2016, Ludwigsburg

Once upon a time...



Once upon a time...



Integration with other applications

- Link to external application
- Link to a scout application
- Application inframing

Link to external applications

With a menu

dev Config [Jenkins] x Contacts Application x

localhost:8082

Contacts

Persons

> Organizations

Events

Quick access v

^ v New Edit View in external Application

Name	City	Country
Alphatom	London	United Kingdom
Bignix	Birmingham	United Kingdom
Flextexon	Paris	France

With a menu

```
@Order(3000)
public class ViewExternalAppMenu extends AbstractMenu {

    @Override
    protected String getConfiguredIconId() {
        return Icons.CaretRight;
    }

    @Override
    protected String getConfiguredText() {
        return TEXTS.get("ViewExternalApp");
    }

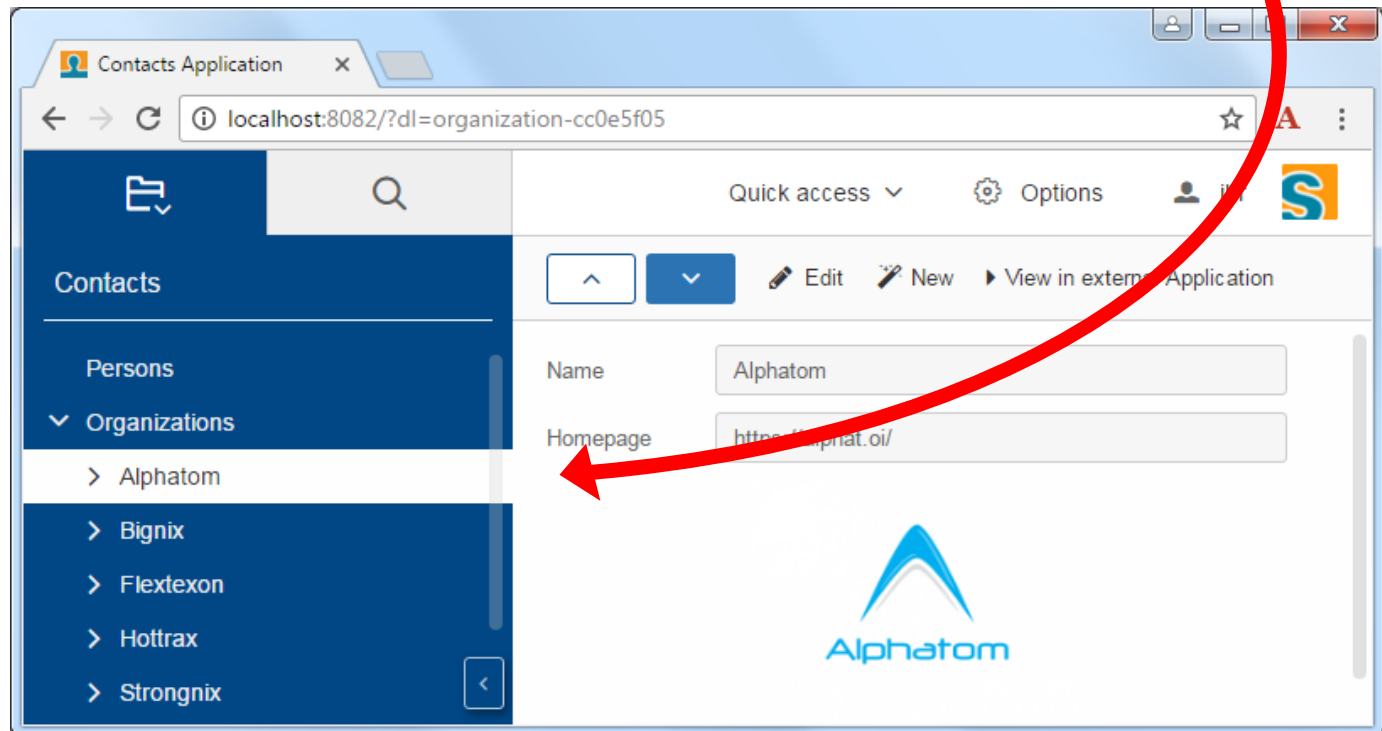
    @Override
    protected void execAction() {
        String url = "http://localhost/extapp/" +
                    getOrganizationIdColumn().getSelectedValue() + ".html";
        ClientSession.get().getDesktop().openUri(url, OpenUriAction.NEW_WINDOW);
    }
}
```

Link to a scout application

Deep linking

<http://application:8082/?dl=organization-cc0e5f05>

business convention => organization id: **cc0e5f05**



Deep link handler

→ Classes implementing:



`IDeepLinkHandler`

→ Example implementation:

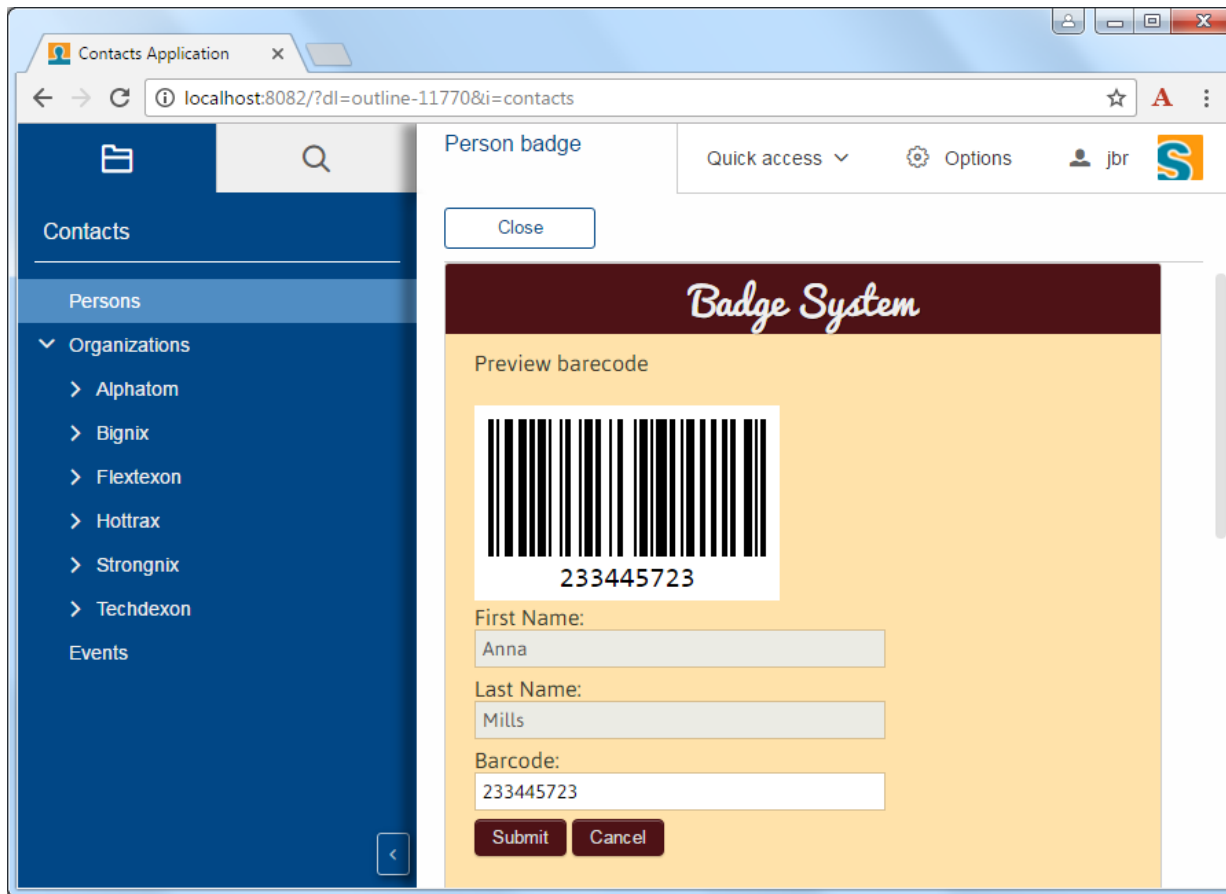


`OrganizationDeepLinkHandler`

- Provide a pattern for matches
- Navigate into the outline tree if there is a match
- Compute the link when a page is selected

Application inframing

Integration of a web application in an iframe



Scout Form

→ Classic Scout Form (example)



PersonBadgeForm

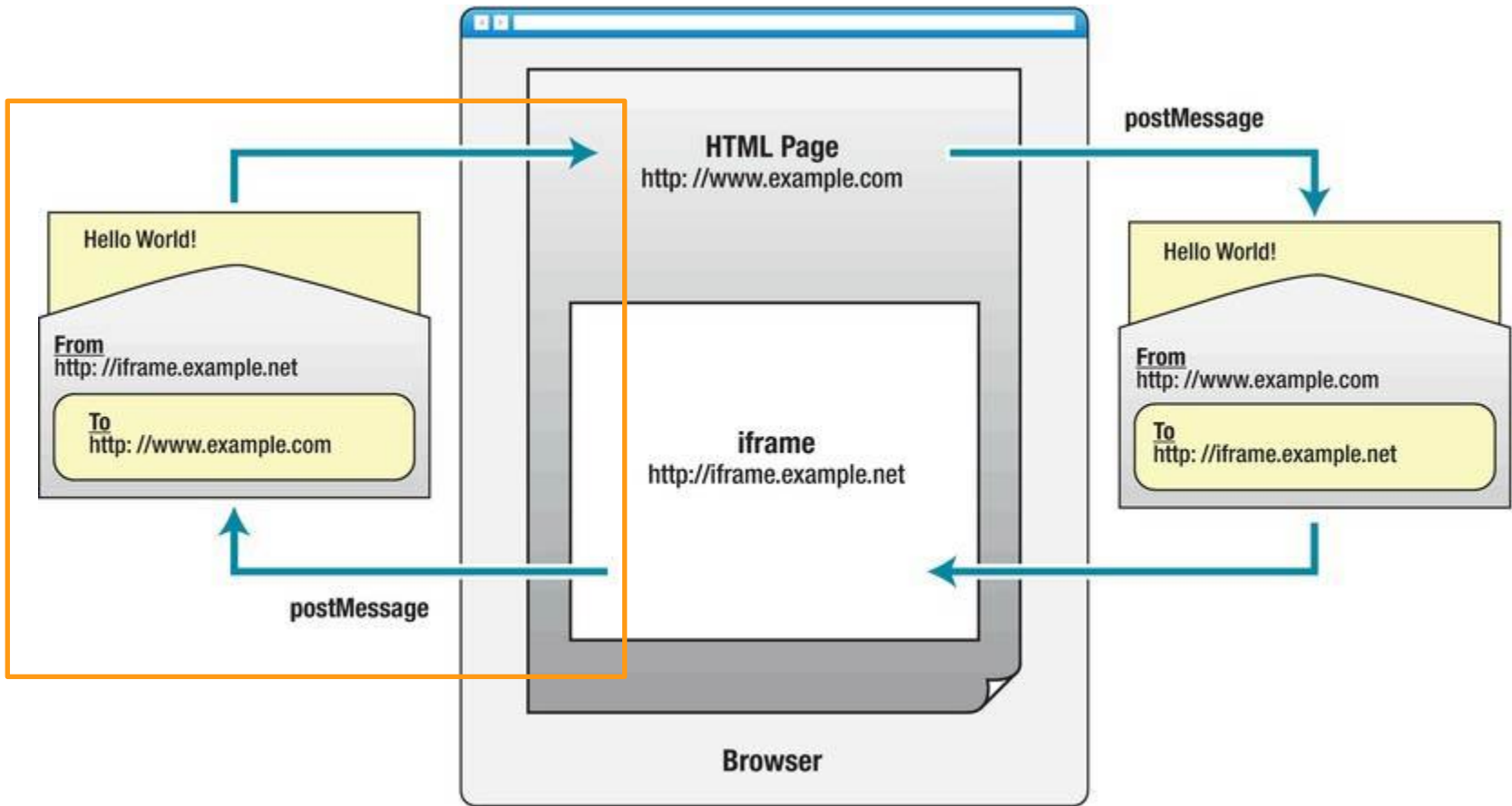
→ Containing a browser field, a class extending:



AbstractBrowserField

- Sandbox: you need to disable it in order to execute JS code
- Communication with the Scout Application (notification)

HTML5: window.postMessage



HTML5: window.postMessage

→ JavaScript code (3rd Party):

```
$('#barcodeForm').submit(function() {  
    top.postMessage("SUBMIT_CLICKED", "*");  
});
```

→ Java Scout code in the client (UI-Server):

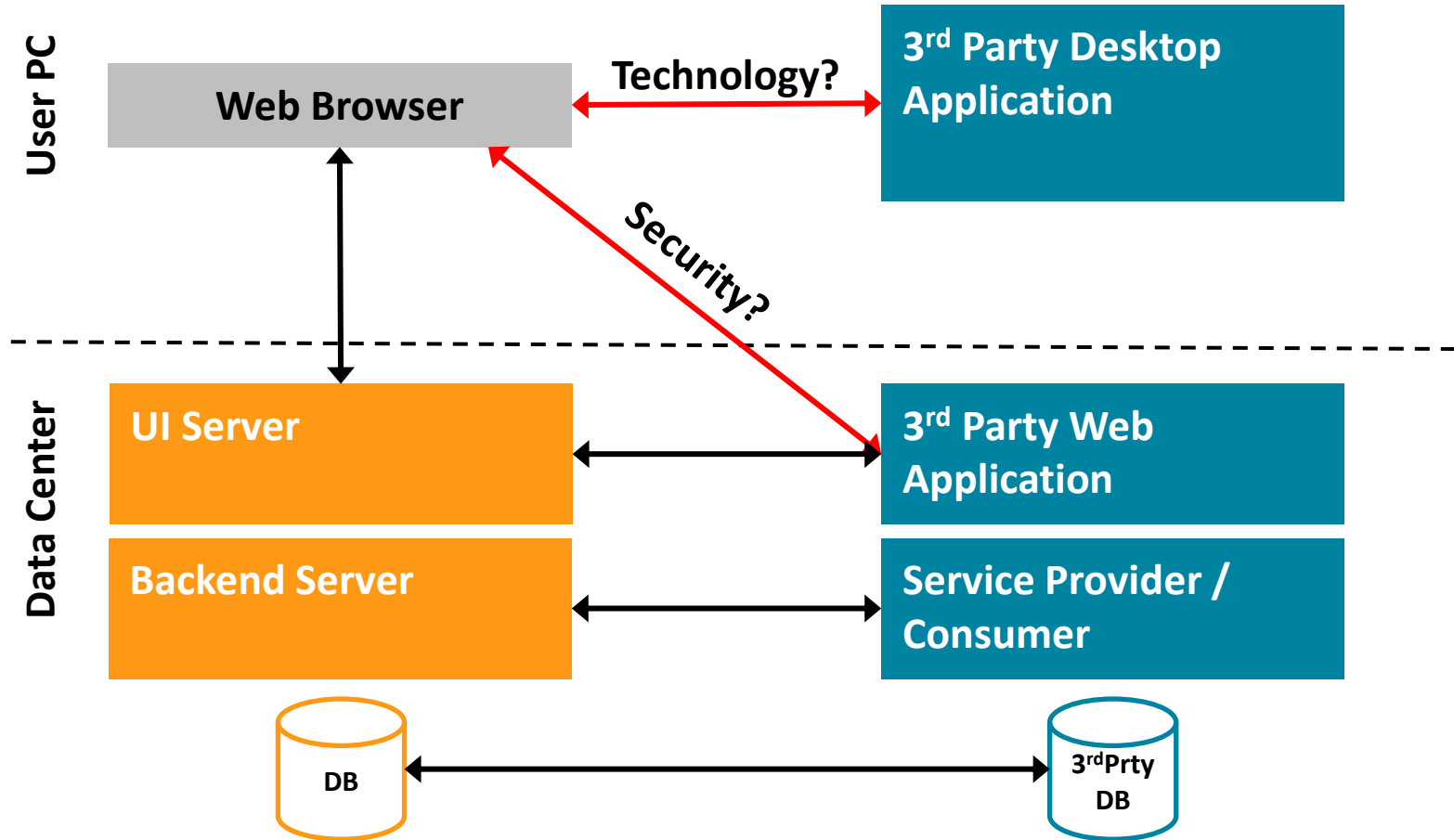
```
@Override  
protected void execPostMessage(String data, String origin) {  
if ("SUBMIT_CLICKED".equals(data)) {  
    doClose();  
}  
//...  
}
```

Going further

- Integration in the Scout Form live cycle
 - Mark form as containing modification
 - Ask “if save needed”
- Parsing and handling the result in the Scout application

Do not forget security!

Integration Points



Thanks

@EclipseScout