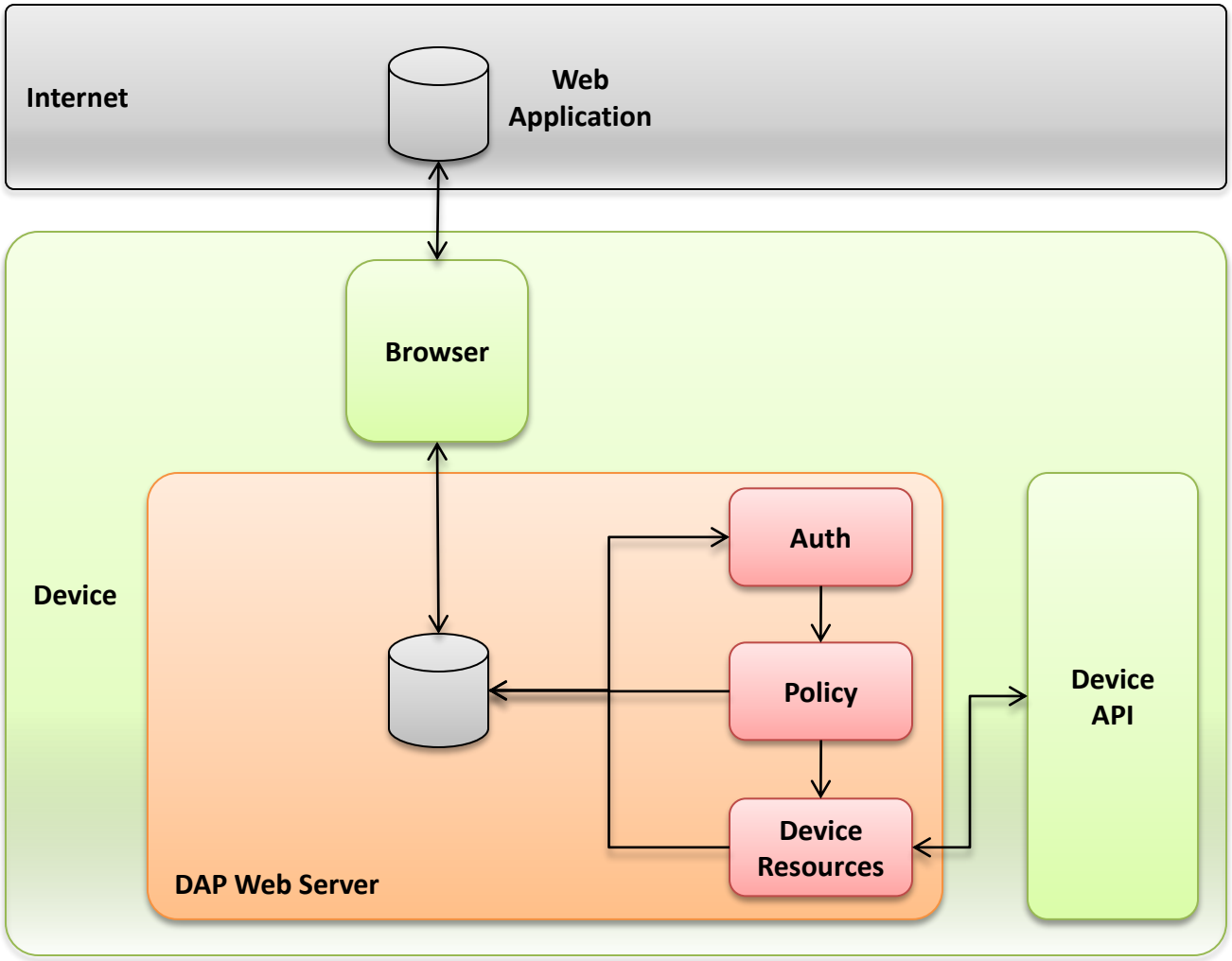
# Access to local device functionality through REST APIs

Johan Apelqvist
Research Manager
R&T Advanced Concepts

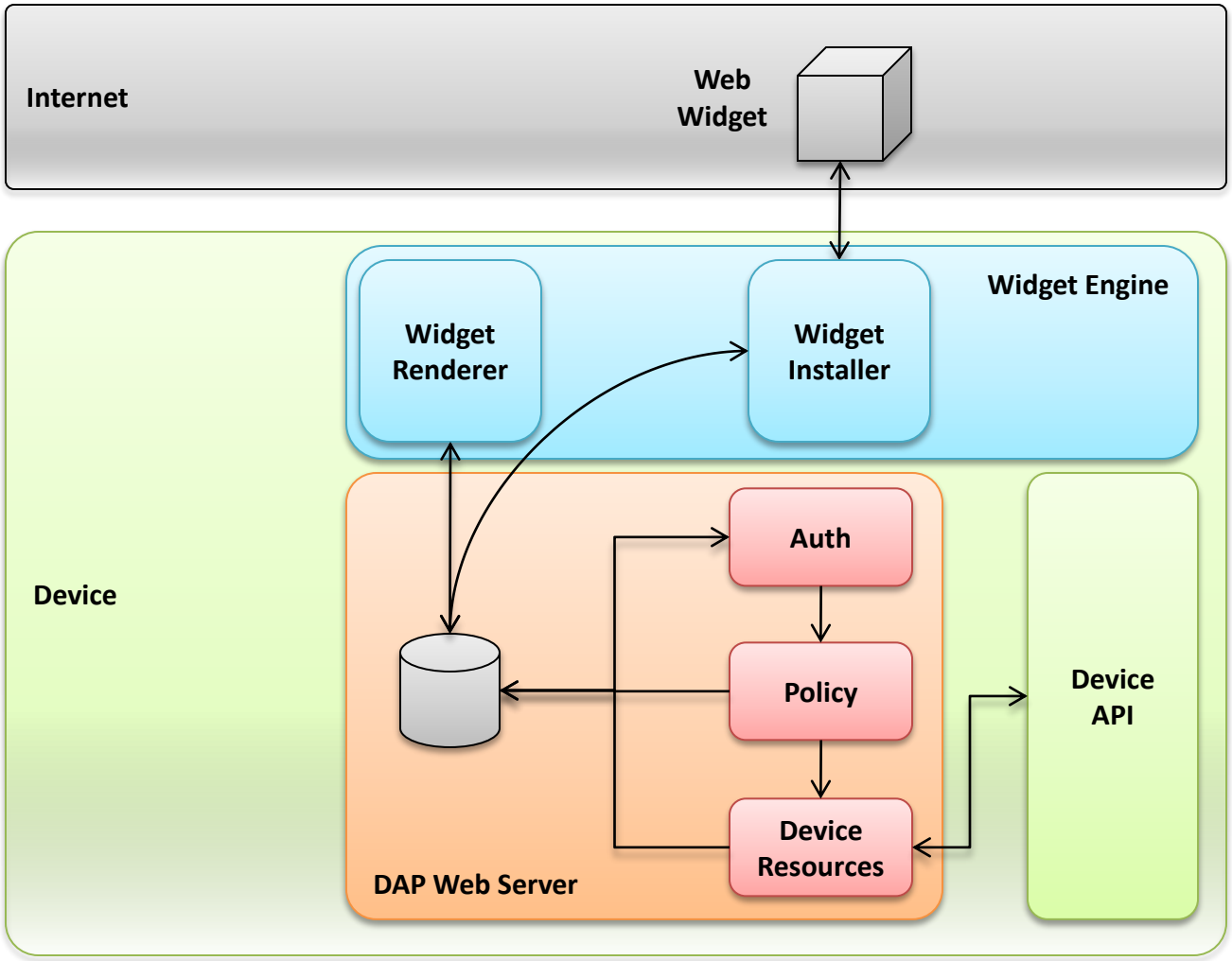
Claes Nilsson
Senior Technology Strategist
R&T UI/App/Web

Sony Ericsson

# Web Application Architecture



# Web Widget Architecture



# Access Control Chain

- Authentication method chosen MUST ensure identification between Customer (**Web Application**) and Provider (**DAP Web Server**).
- **Auth** provides user interaction to grant access to requested resources.
- **Policy** is an optional component that provides a “prearranged trust relationship” between **Web Application** and **Device Resource**
  - For example validates each request against one or more policy documents to decide access to the protected resource.
- **Device Resources** is the API provider interpreting the request, invoking the correct native device API(s) and formatting the response.

# Web Application Scenario

1. The user accesses the application from the browser.

2. Authentication process commence by setting up a (trusted) relation between the **DAP Web Server** and the server hosting the **Web Application**.

3. The application asks for permission to a set of **Device Resources** (as parameters in request or via link to a manifest file). If a “prearranged trust relationship” is provided by a **Policy** framework the policy framework grants (all or a subset of) the requested resource(s) based on the policy document(s).

   Alternatively **Auth** provides user interaction to grant access to requested resources. Once the requested resource(s) are granted the user do not have to clear access next time the application is accessed as long as the resource(s) are not changed.

4. The application requests a resource that is passed through the access control chain for validation until the request is answered and the response is returned to the application.

# Web Widget Scenario

1. The user downloads the widget from the internet ending up in the widget installer subscribing to widget packages.
2. The subsequent steps are identical to web applications described above with the exception of the manifest read from the widget package instead of provided as a link on the content page.

# Management

- The **DAP Web Server** must provide a “manager” function.
- The main task for the “manager” is to allow the user to manually withdraw previously granted permissions for web applications access to **Device Resources.**
- In addition the “manager” may provide the possibility for the user to pre-configure an access policy. For example, the user could state that only web applications whose origin is “operator” and “vendor” are allowed to access the contacts API.

# REST-style Contacts Examples

- Create a new contact:

**Request**

```
http://<authority>/dap/contacts/create.json?...&name=Mr.%20Robert%20Smith%20Jr
&nicknames=Bob
```

**Response**

```
{name: 'Mr. Robert Smith Jr', nicknames: ['Bob'], phones: [], emails: [],
addresses: [], impps: [], serviceId: null, categories: []}
```

- Find contacts:

**Request**

```
http://<authority>/dap/contacts/find.json?...&name=Robert&nicknames=Bob
```

**Response**

```
[{name: 'Mr. Robert Smith Jr', nicknames: ['Bob'], phones: [], emails: [],
addresses: [], impps: [], serviceId: null, categories: []}]
```