Service-Oriented Architectures for Embedded Systems Using Devices Profile for Web Services

Elmar Zeeb, Andreas Bobek, Hendrik Bohn,
and Frank Golatowski

SOA for **Embedded Systems** Using DPWS

1. **What is DPWS? Why you need DPWS?**

4. **Outview**

Outline

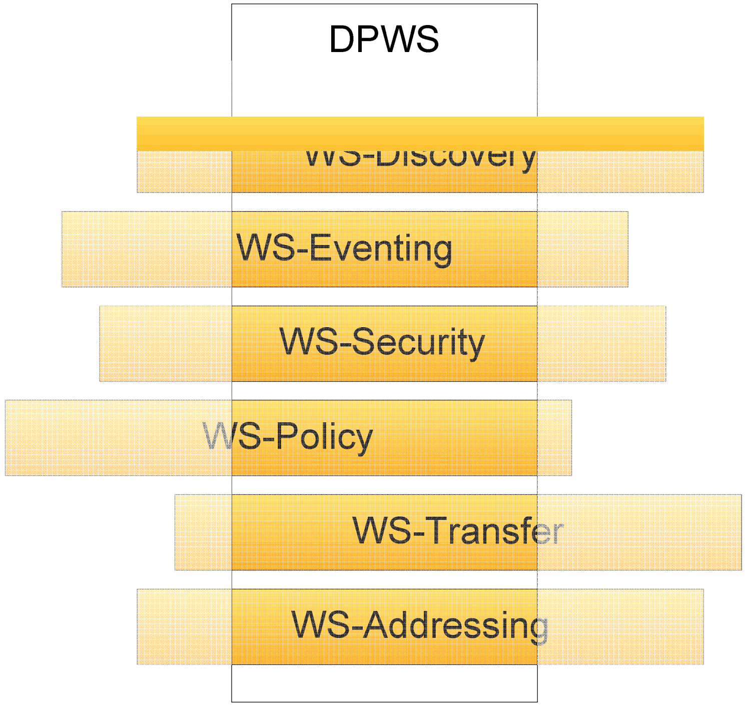
2. Implementation Pitfalls

3. **ws4d Initiative**

Devices Profile for Web Services

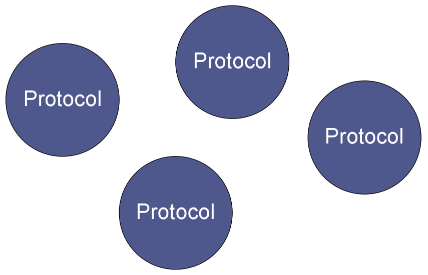
- What is DPWS?
  - Specification for Distributed Embedded Systems based on Web Services technology
  - Specification which describes way how to bring Web Services to the devices level
  - D**P**WS is a Profile

- Initially UPnP V2.0
- Basis for European project SIRENA

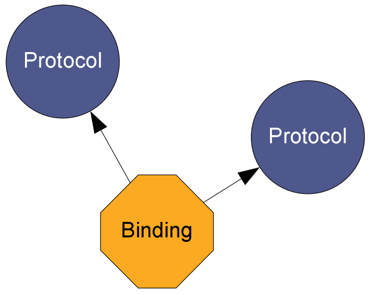


Web Services Technology

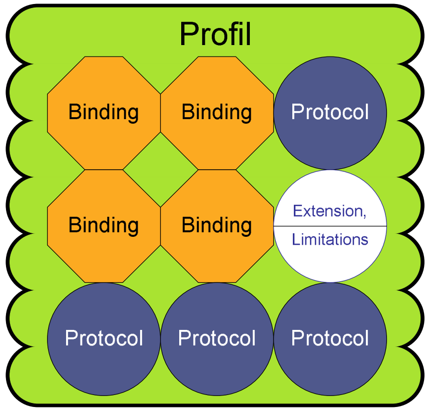
- Collection of protocols which are loosely coupled: **Protocols** specify messages and its semantics to cover particular functionalities.
- **Bindings** specify collaboration of some protocols to overcome loose coupling.
- **Profiles** consisting of a set of protocols and bindings; they enhance and limit them.



- WS-Addressing
- SOAP 1.2



- „SOAP over UDP“
- WSDL 1.1/HTTP Binding

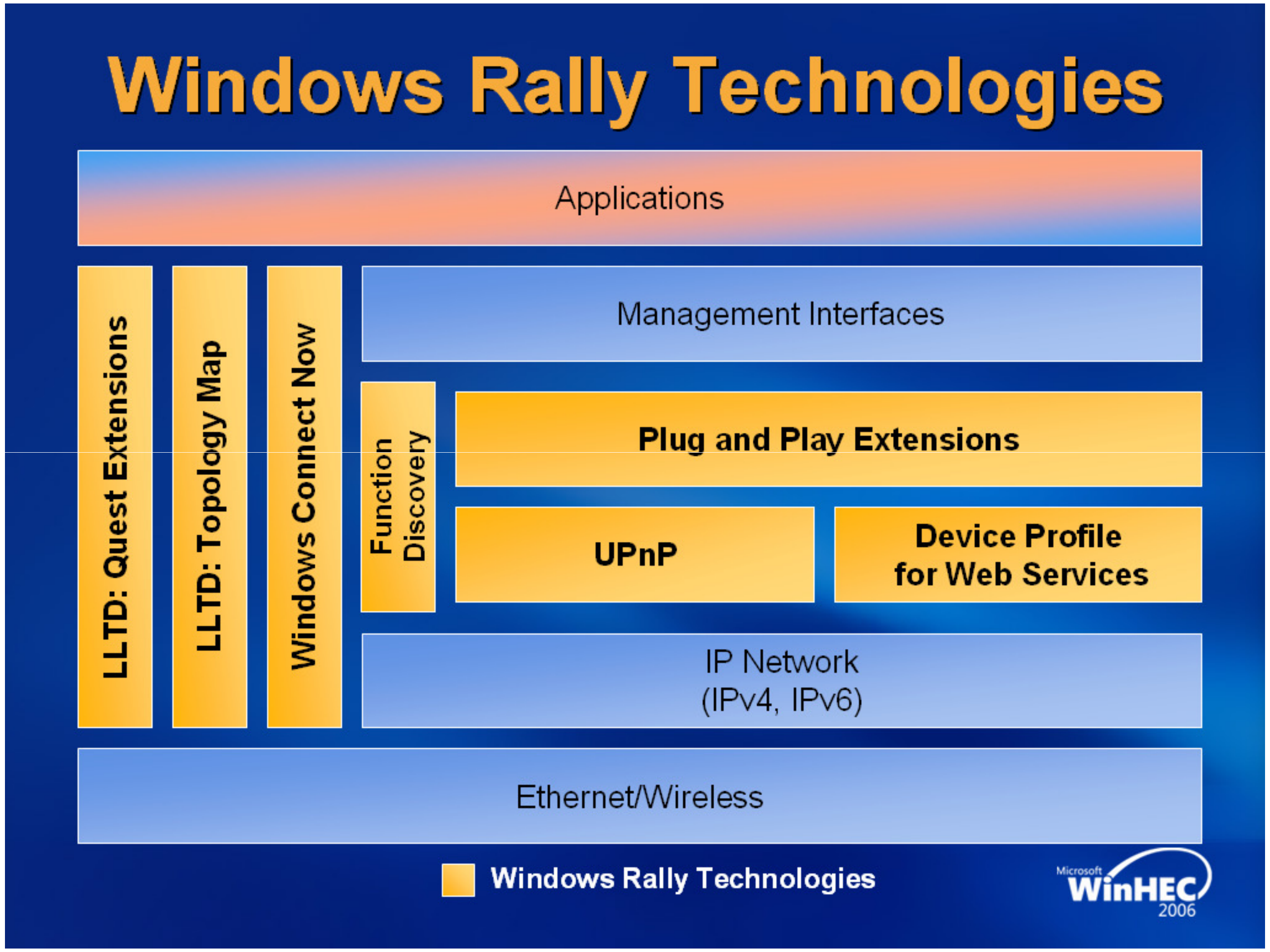


- WS-Basic Profile
- DPWS

Devices Profile for Web Services

- Why do you need DPWS?
  - Easy integration of embedded devices into IT-infrastructures
  - Advanced management and configuration of distributed embedded systems
  - Free to use (no costs)

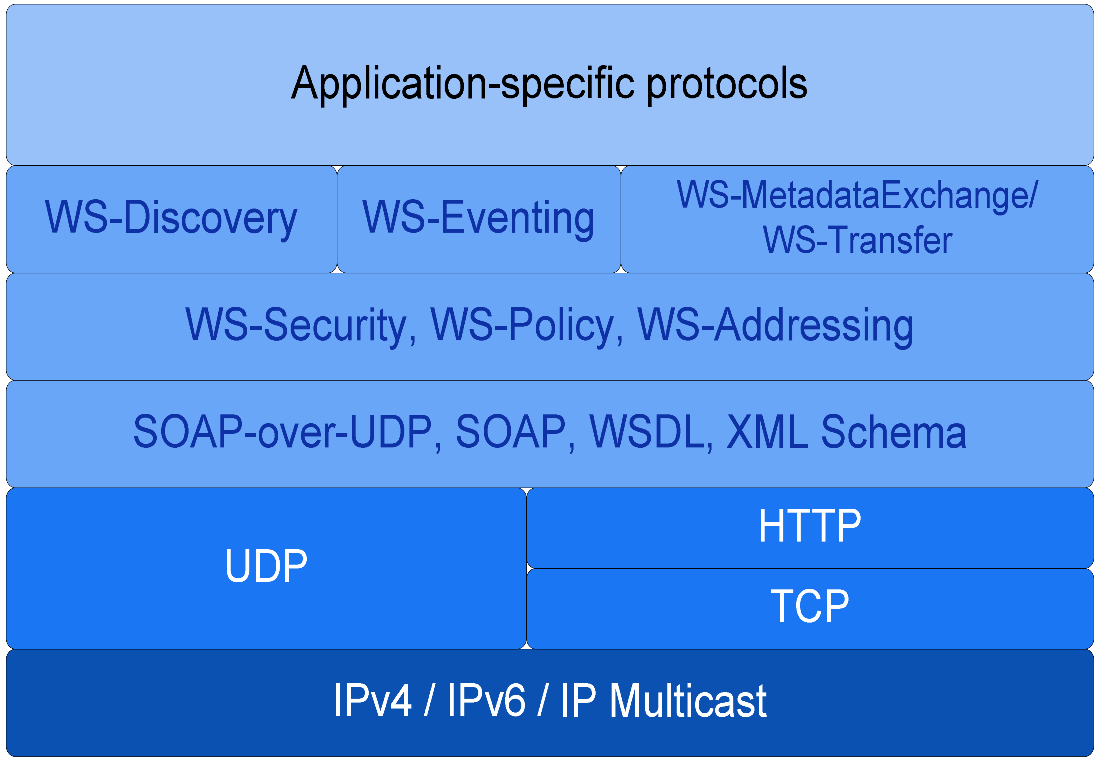
DPWS in MS-VISTA



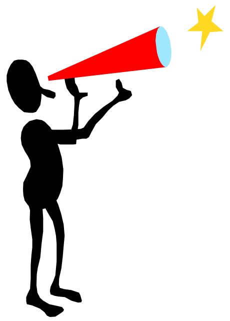
Source: Dave Roth, Web Services on Devices, WinHec2006

Devices Profile for Web Services

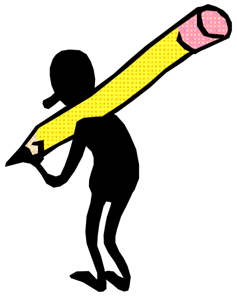
- Secure Web service capabilities on resource-constraint devices
- Dynamic device discovery
- Device and Service Description
- Eventing


Application-specific protocols
WS-Discovery
WS-Eventing
WS-MetadataExchange/ WS-Transfer
WS-Security, WS-Policy, WS-Addressing
SOAP-over-UDP, SOAP, WSDL, XML Schema
UDP
HTTP
TCP
IPv4 / IPv6 / IP Multicast

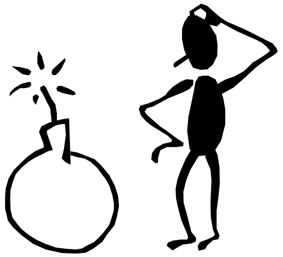
Messaging



Discovery



Description



Eventing

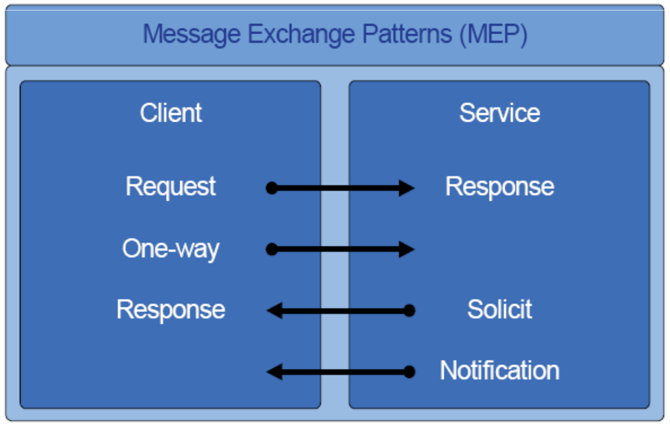
Messaging

- DPWS uses SOAP 1.2 and WS-Addressing
- SOAP-over-UDP and IP-Multicast for Discovery
- Service on device must at least support SOAP 1.2 over HTTP
- SOAP features are restricted (e.g. message size)
- Attachments for bigger messages

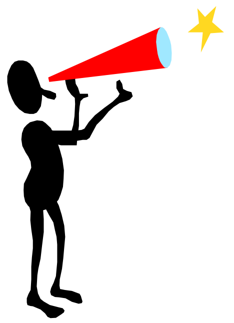
Messaging

- DPWS restricts the WS specifications
- Only needed functionality
  - to implement DPWS on embedded systems
  - to hold message size small

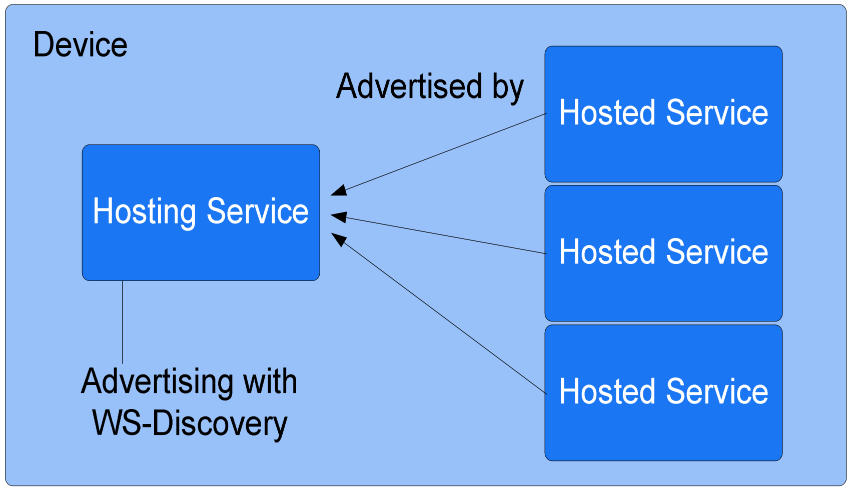
- <u>Must</u> support HTTP chunked transfer coding
- May support MTOM
- <u>Must</u> support receiving and sending SOAP1.2 envelopes over HTTP
- <u>Must</u> support request-response MEP (message exchange patterns)
- <u>Must</u> respond to one-way MEP
- <u>Must</u> support WS-adressing by including a relationship field in message information header
- …



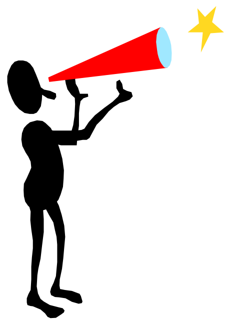
Discovery



- Uses WS-Discovery
- Only used for device discovery
- Done by Hosting Service
- Implicit (Hello/Bye) and explicit (Probe) discovery

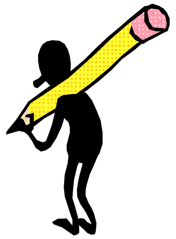


Discovery



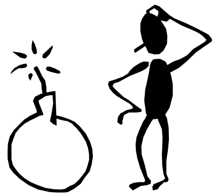
- For basic interop
  - a device <u>must</u> support sending and receiving discovery messages over UDP unicast and multicast.
- Static scenarios (HTTP address of a device is known)
  - a device <u>must</u> support
    - receiving discovery messages over HTTP and
    - respond at least with *HTTP 202 Accepted*
- *wsdp:Device (*target service type defined by DPWS <u>should</u> be included in discovery messages if types are included.
- A device <u>must</u> at least support the *rfc2396* and *strcmp0* scope matching rules to simplify a resource-constraint device implementation

Description



- Uses WS-Transfer and WS-MetadataExchange
- Description consists of several parts
  - Characteristics (model and device specific
  - Hosting (relationship between hosting and hosted services)
  - WSDL (Web Services Definition Language)
  - Policy
- Describes device at runtime

Eventing



- Uses WS-Eventing
- Used for managing event channels
- EventSource, EventSink, SubscriptionManager
- DPWS defines event delivery mode and event filter mode
  - Push delivery mode by notification operations
    - An operation implemented by the event sink
  - Action filter mode by WS-Addressing action
    - Action of operation implemented by the event sink

Implementation Pitfalls

- Hard to figure out basic functionality needed for compliance
- Discovery is used only for device discovery
- Functional discovery is shifted to the client (can be done with Description)
- Semantic of device and service type system is weak (leads to unclear functional discovery)
- Well defined device side and vaguely defined client side in specification

WS4D Implementations

- WS4D-gSOAP
  - Target: Embedded Systems, C
- WS4D-JavaME
  - Target: Embedded Systems, Java
- WS4D-Axis2
  - Target: Enterprise Systems, Java

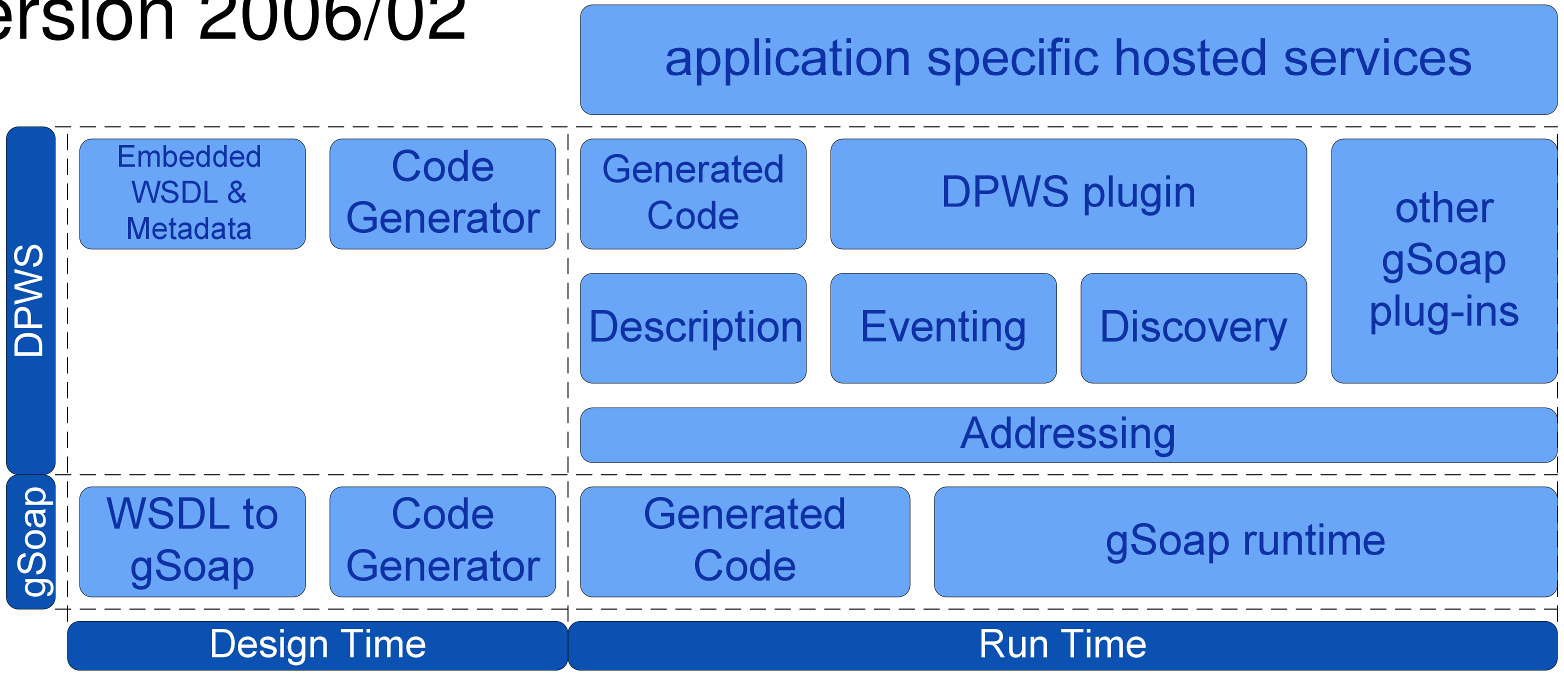
Web Services for Devices

- www.ws4d.org
- Run by three parties: University of Rostock, University of Dortmund and Materna

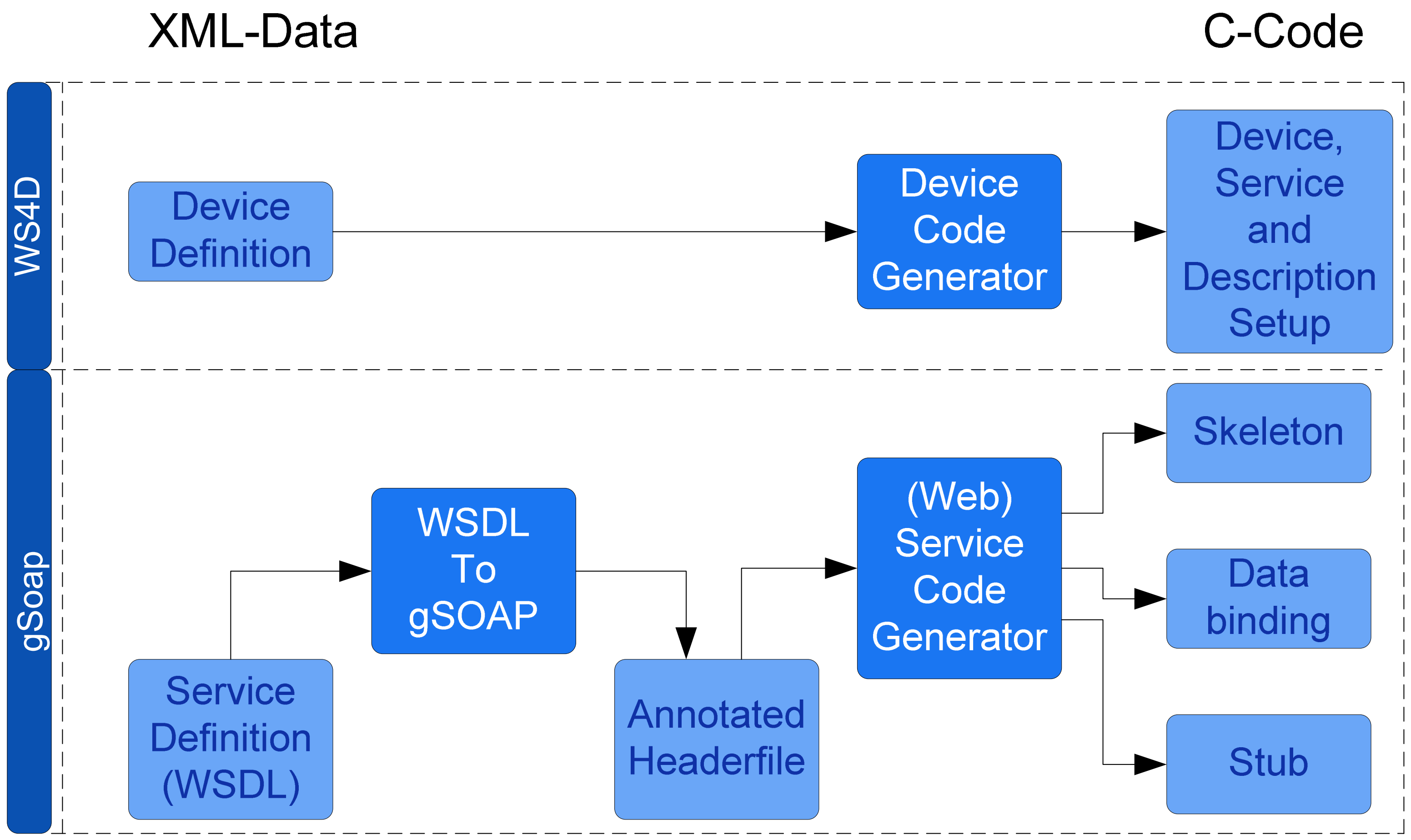
- Basis for a community for building Web services on devices based on the „Devices Profile for Web Services“ (DPWS)
- Platform to distribute know how and results of the ITEA SIRENA project
- Toolkits to build heterogeneous digital device ecosystems (platform and language independent)

WS4D-gSOAP

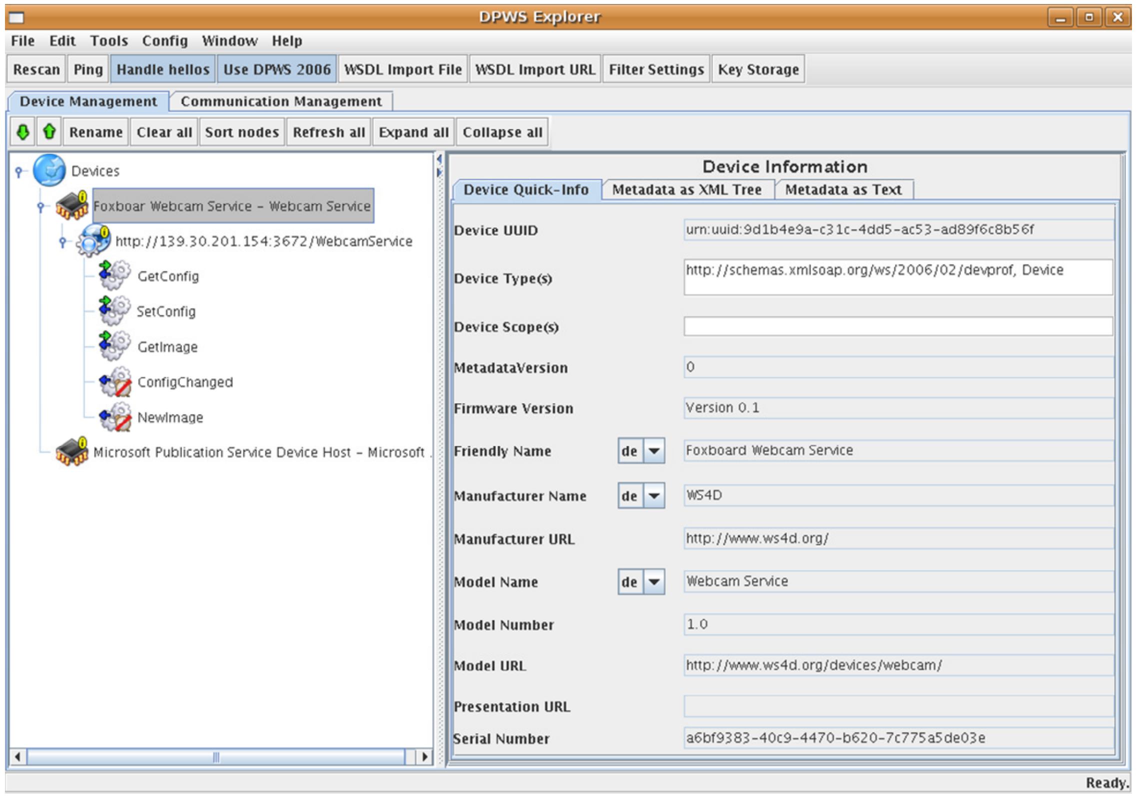
- Based on gSOAP
- Supports DPWS Version 2006/02



WS4D Device Development Work-Flow



Device description at runtime


DPWS Explorer
File Edit Tools Config Window Help
Rescan Ping Handle hellos Use DPWS 2006 WSDL Import File WSDL Import URL Filter Settings Key Storage
Device Management Communication Management
Rename Clear all Sort nodes Refresh all Expand all Collapse all
Devices
Foxboar Webcam Service - Webcam Service
http://139.30.201.154:3672/WebcamService
GetConfig
SetConfig
GetImage
ConfigChanged
NewImage
Microsoft Publication Service Device Host - Microsoft
Device Information
Device Quick-Info Metadata as XML Tree Metadata as Text
Ready.

Device UUID		urn:uuid:9d1b4e9a-c31c-4dd5-ac53-ad89f6c8b56f
Device Type(s)		http://schemas.xmlsoap.org/ws/2006/02/devprof, Device
Device Scope(s)		
MetadataVersion		0
Firmware Version		Version 0.1
Friendly Name	de	Foxboard Webcam Service
Manufacturer Name	de	WS4D
Manufacturer URL		http://www.ws4d.org/
Model Name	de	Webcam Service
Model Number		1.0
Model URL		http://www.ws4d.org/devices/webcam/
Presentation URL		
Serial Number		a6bf9383-40c9-4470-b620-7c775a5de03e

But for code generation we need it at development time!

Device definition in implementation

```
[...]

int main(int argc, char **argv)
{
  int ret = DPWS_OK;

  DPWS_INT_CFG(DPWS_INT_BOOT_SEQ, 0);
  DPWS_INT_CFG(DPWS_INT_METADATA_VERSION, 100);
  DPWS_INT_CFG(DPWS_INT_HTTP_PORT, 8888);
  DPWS_INT_CFG(DPWS_INT_HTTP_BACKLOG, 10);
  DPWS_STR_CFG(DPWS_STR_MANUFACTURER, "Universität Rostock");
  DPWS_STR_CFG(DPWS_STR_MANUFACTURER_URL, "http://www.uni-rostock.de");
  DPWS_STR_CFG(DPWS_STR_MODEL_NAME, "LinuxDPWSTimer");
  DPWS_STR_CFG(DPWS_STR_MODEL_NUMBER, "0.1");
  DPWS_STR_CFG(DPWS_STR_MODEL_URL, "http://www.uni-rostock.de");
  DPWS_STR_CFG(DPWS_STR_UPC, "677652530787");
  DPWS_STR_CFG(DPWS_STR_PRESENTATION_URL, "http://www.uni-rostock.de/");
  DPWS_STR_CFG(DPWS_STR_FRIENDLY_NAME, "LinuxDPWSTimer");
  DPWS_STR_CFG(DPWS_STR_FIRMWARE_VERSION,         "0.1");
  DPWS_STR_CFG(DPWS_STR_SERIAL_NUMBER, "79785654");
  DPWS_INT_CFG(DPWS_INT_MANAGEMENT_SERVICE_ENABLE, 1);

  if ((ret = dpws_add_device_types("http://www.uni-rostock.de/", "ns", DeviceTypes))
      || (ret = dpws_add_service(DEVICE_SERVICE_ID, "http://nonamespace", "sniffer",
                                 noPortTypes , NULL, handle_device_event))
      || (ret = dpws_add_service("urn:timeService", "http://www.uni-rostock.de/", "ns",
                                 portTypes, "http://www.uni-rostock.de/timeservice.wsdl",
                                 handle_event))
      || (ret = dpws_server_init(&dpws_serv, NULL)))
    {
      dpws_print_error_msg(&dpws_serv, stderr, ret);
      exit(-1);
    }

  [...]
```

metadata, device and service must be defined and setup by developer in device implementation

Separate device definition

Device definition in XML

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<wsm:Metadata xmlns:wsm="http://schemas.xmlsoap.org/ws/2004/09/mex" xmlns:ws4d="http://www.ws4d.org/
devices/webcam" xmlns:wdp="http://schemas.xmlsoap.org/ws/2006/02/devprof" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xmlns:wsd="http://schemas.xmlsoap.org/ws/2005/04/discovery"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsm:MetadataSection Dialect="http://schemas.xmlsoap.org/ws/2006/02/devprof/Relationship">
    <wdp:Relationship Type="http://schemas.xmlsoap.org/ws/2006/02/devprof/host">
      <wdp:Host>
        <wdp:Types>ws4d:webcam</wdp:Types>
        <wdp:ServiceId>HostingService</wdp:ServiceId>
      </wdp:Host>
      <wdp:Hosted>
        <wdp:Types>ws4d:WebcamPortType</wdp:Types>
        <wdp:ServiceId>WebcamService</wdp:ServiceId>
      </wdp:Hosted>
    </wdp:Relationship>
  </wsm:MetadataSection>
  <wsm:MetadataSection Dialect="http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisModel">
    <wdp:ThisModel>
      <wdp:Manufacturer lang="de">WS4D</wdp:Manufacturer>
      <wdp:Manufacturer lang="en">WS4D</wdp:Manufacturer>
      <wdp:ManufacturerUrl>http://www.ws4d.org/</wdp:ManufacturerUrl>
      <wdp:ModelName lang="de">Webcam Service</wdp:ModelName>
      <wdp:ModelName lang="en">Webcam Service</wdp:ModelName>
      <wdp:ModelNumber>1.0</wdp:ModelNumber>
      <wdp:ModelUrl>http://www.ws4d.org/devices/webcam/</wdp:ModelUrl>
    </wdp:ThisModel>
  </wsm:MetadataSection>
  <wsm:MetadataSection Dialect="http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisDevice">
    <wdp:ThisDevice>
      <wdp:FriendlyName lang="de">Foxboard Webcam Service</wdp:FriendlyName>
      <wdp:FriendlyName lang="en">Foxboar Webcam Service</wdp:FriendlyName>
      <wdp:FirmwareVersion>Version 0.1</wdp:FirmwareVersion>
      <wdp:SerialNumber>a6bf9383-40c9-4470-b620-7c775a5de03e</wdp:SerialNumber>
    </wdp:ThisDevice>
  </wsm:MetadataSection>
</wsm:Metadata>
```

WS4D Codegen

Generated device, service and description setup

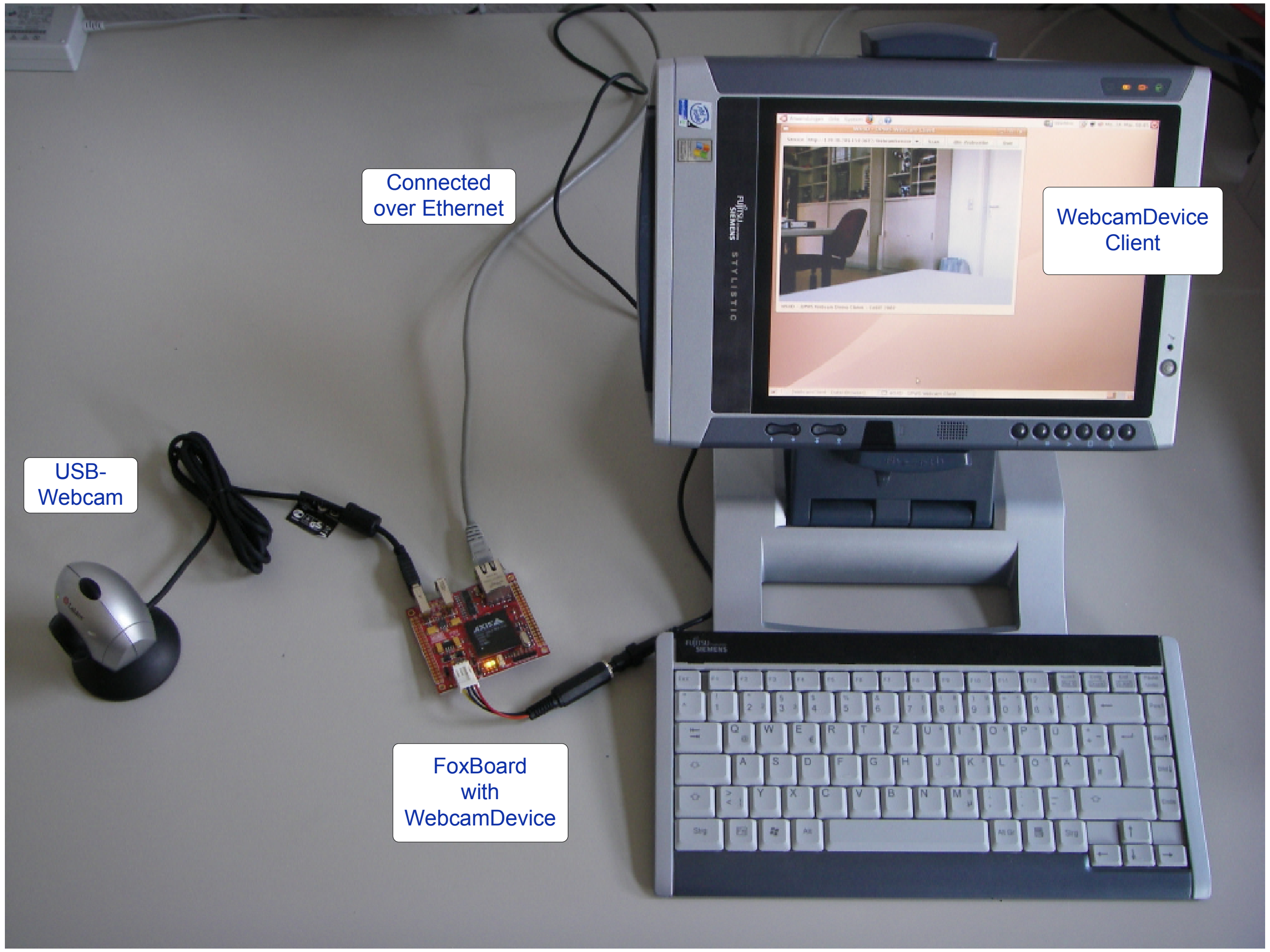
```c
#ifndef DPWS_SERVER
#define DPWS_SERVER
#endif
#include "dpwsH.h"
#include "stddpws.h"

/* Device Metadata */

const dpws_device_FriendlyName_var(FriendlyName) = {

        dpws_init_localized_string("de", "Foxboard Webcam Service"),

        dpws_init_localized_string("en", "Foxboar Webcam Service"),

 };

const dpws_device_FirmwareVersion_var(FirmwareVersion) =
        "Version 0.1";

const dpws_device_SerialNumber_var(SerialNumber) =
        "a6bf9383-40c9-4470-b620-7c775a5de03e";


void dpws_setmetadata_ThisDevice(struct dpws_s *device)
{

  dpws_device_set_FriendlyName(device, FriendlyName,2);

  dpws_device_set_FirmwareVersion(device, FirmwareVersion);

  dpws_device_set_SerialNumber(device, SerialNumber);

}
[..]
```

Device with WS4D-Codegen

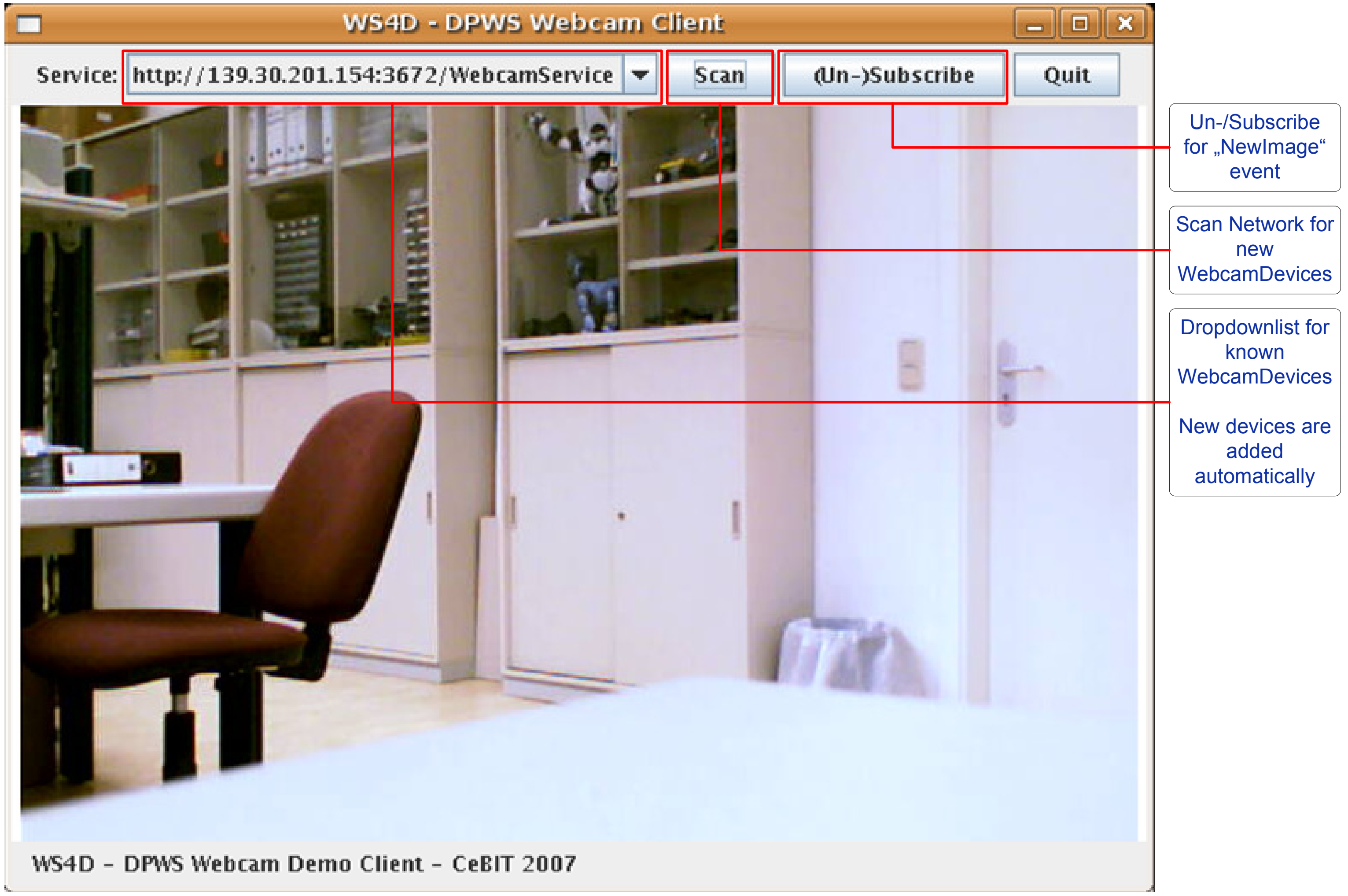
```
[...]

  /* initialize device and services */
  if (dpws_init (&device, host)
      || dpws_setup_HostingService (&device, service, uuid, 100)
      || dpws_setup_WebcamService (&device, service, "webcam.wsdl", 100))
    {
      printf ("\nWebcamDevice: Can't init device and services\n");
      dpws_done (&device);
      exit (0);
    }

  /* Set Metadata */
  dpws_set_Metadata (&device);

  /* Update Metadata */
  if (dpws_update_Metadata (&device))
    {
      printf ("\nWebcamDevice: Can't init metadata\n");
      dpws_done (&device);
      exit (0);
    }

[...]
```

- DPWS specific part of initialization is small
- Developer can concentrate on implementing functionality

WebcamDevice Example



WebcamDevice Example


WS4D - DPWS Webcam Client
Service:
http://139.30.201.154:3672/WebcamService
Scan
(Un-)Subscribe
Quit

WS4D - DPWS Webcam Demo Client - CeBIT 2007

Un-/Subscribe for „NewImage" event

Scan Network for new WebcamDevices

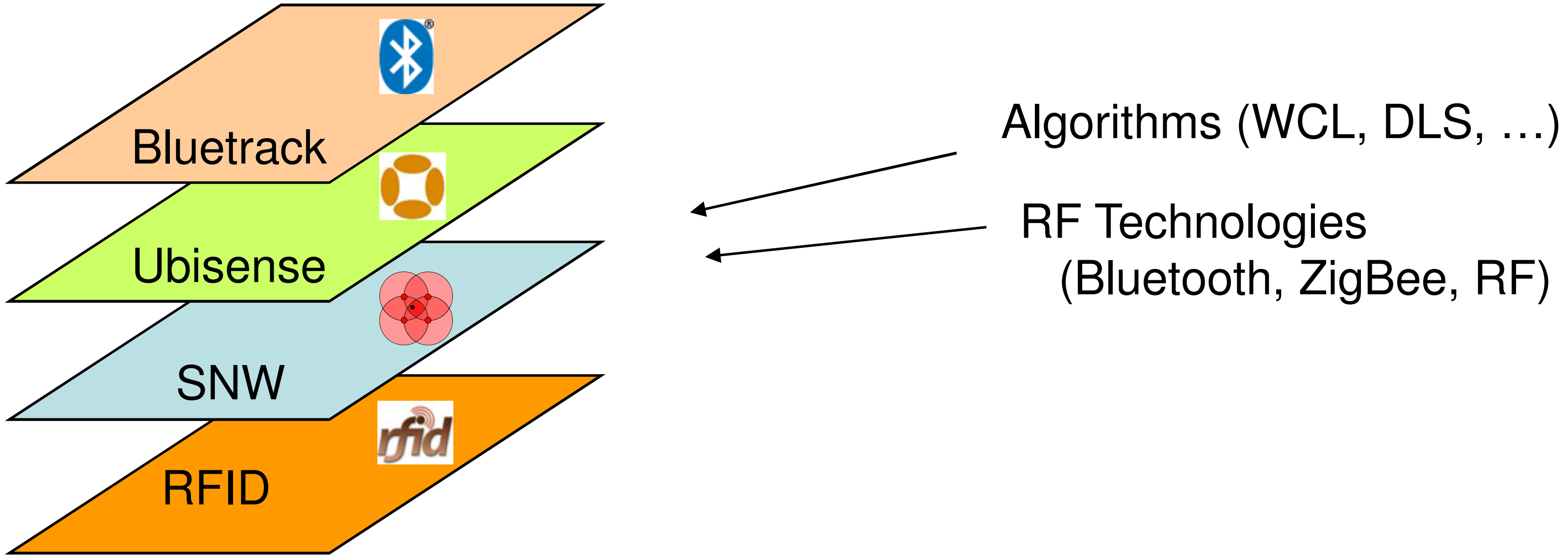
Dropdownlist for known WebcamDevices

New devices are added automatically

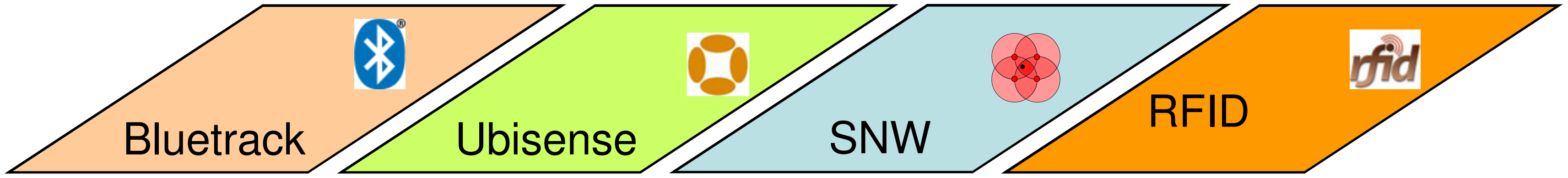
Outlook

- Workflow for Devices
- Implement Discovery Proxy with an UDDI interface
- Device and Service Templates to improve Device specification Work-Flow

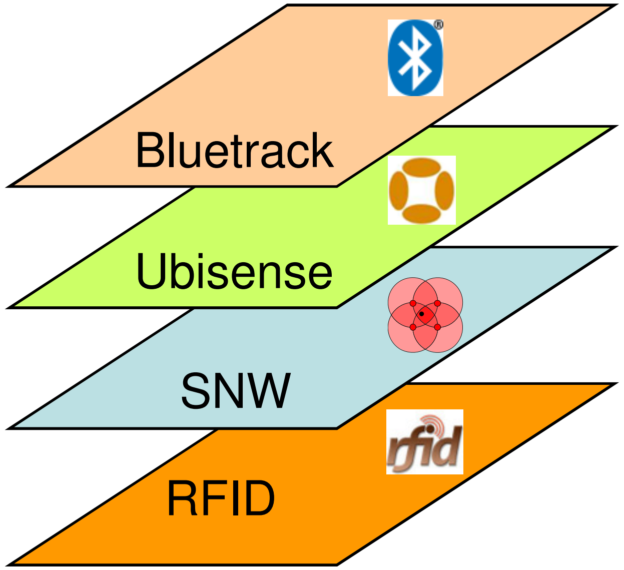
Localization system

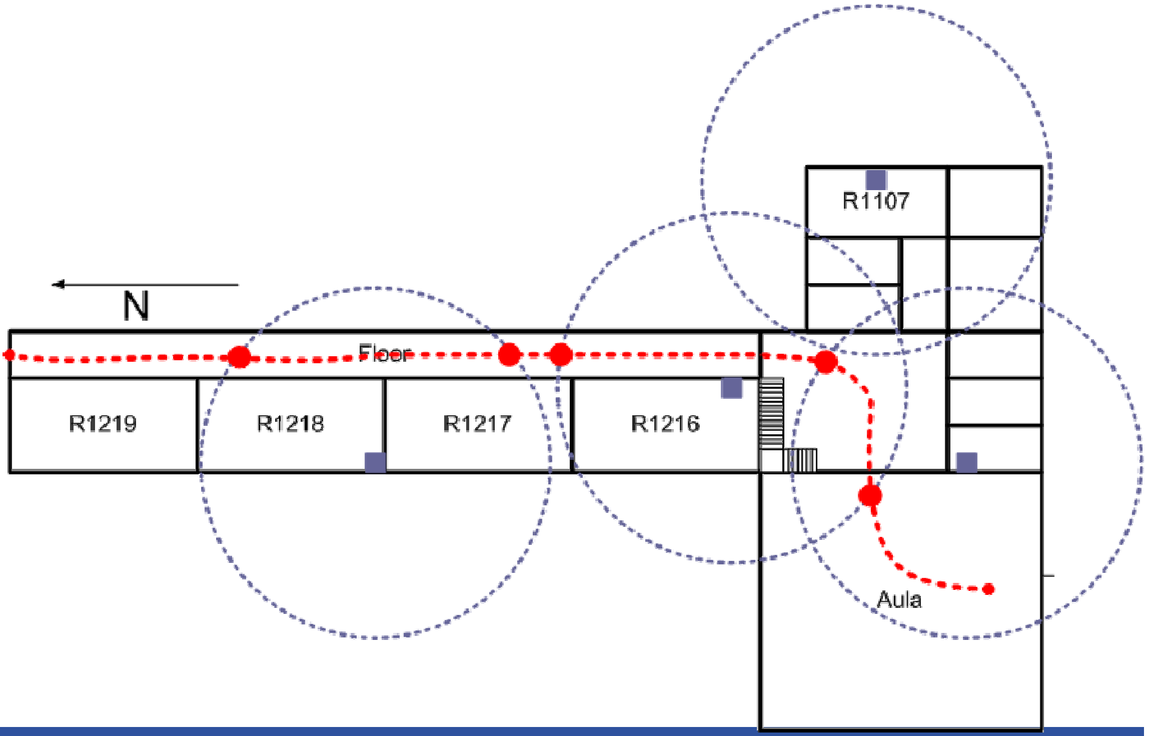


Localization system

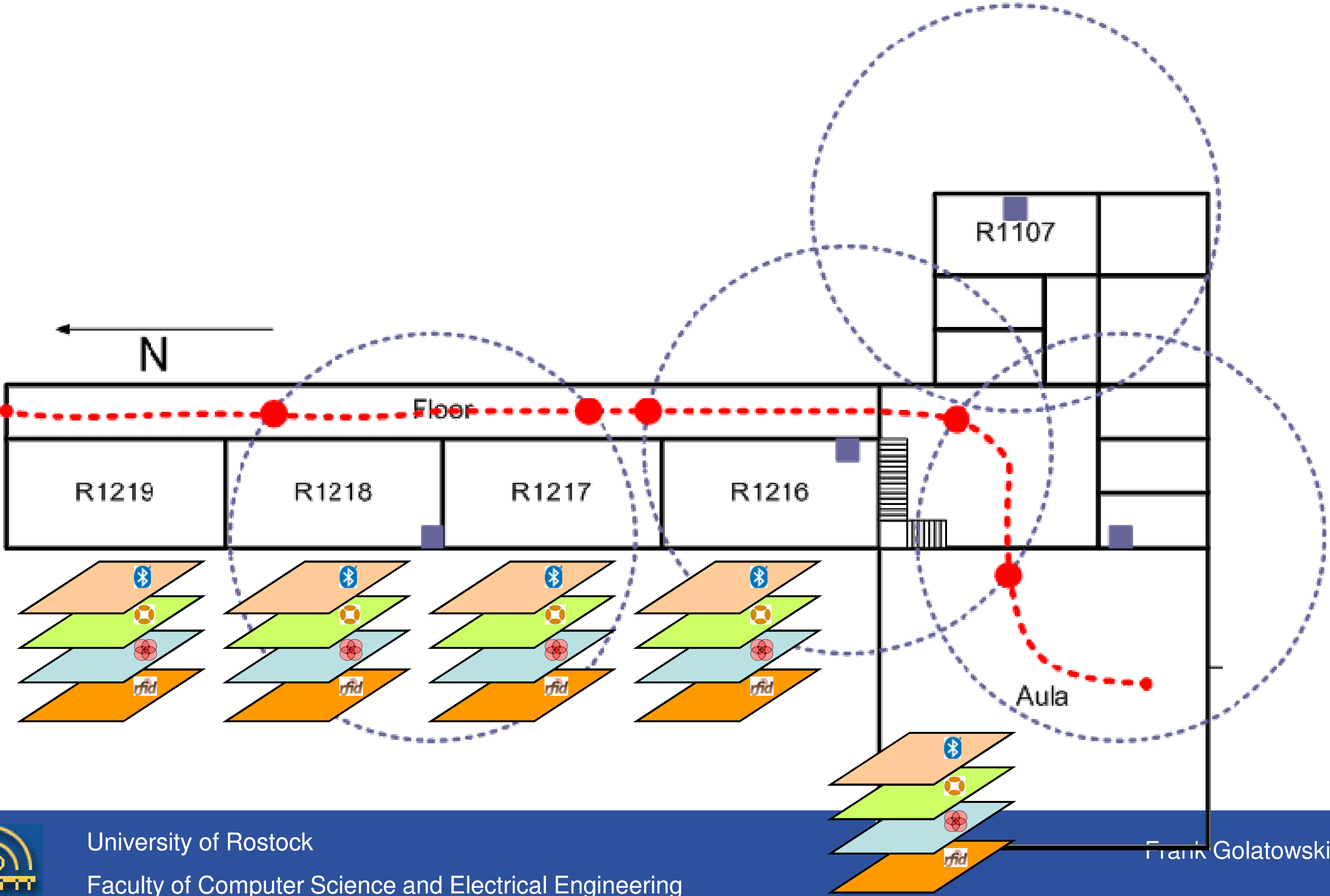


Fusion of localization systems

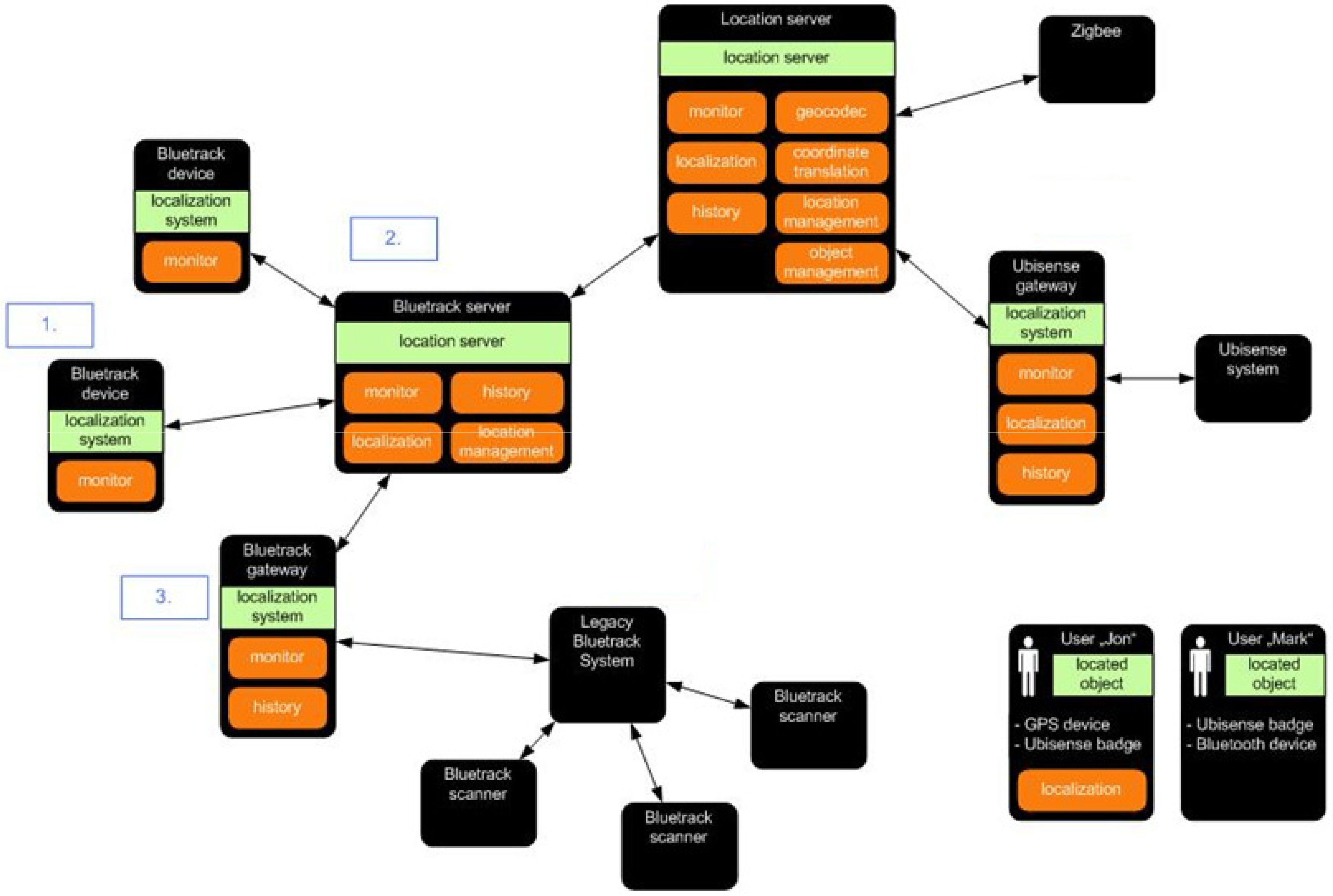




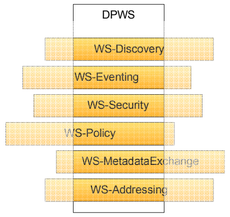
Fusion of localization systems



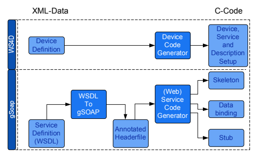
)A based Integration of different location systems using an generic localization interfa



Conclusion

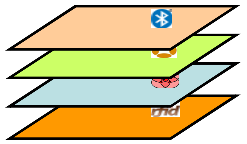


- DPWS



- Improved development flow
- Pitfalls

- WS4D Initiative



- DPWS as integration technology