ESA-HMI standardized framework for designing Human-Machine Interfaces

***Joanna Modławska**, Krzysztof Samp, **Michał Tanaś**, Dariusz Walczak, Nieves Salor Moral*

ITTI Sp. z o.o. Poznań (PL)
Vitrociset Belgium, Noordwijk (NL)

SESP, 26 March 2015

Agenda

- Company information
- Technical objectives
- Requirements
- Selected implemented controls
- Engineering approach
- Summary and future work

ITTI – company information

- Company mission:
  - **independent consulting** in the area of telecommunications, IT and business,
  - **applied research** in **Information & Communication Technologies**
  - **development of innovative applications and software solutions**
- Main facts:
  - **SME** – ca. **80 employees** with professional certificates, e.g. **PRINCE2, MSP, ITIL, BS 7799/ISO 27001, TOGAF 8, Certificate in Software Testing (SJSI/ISTQB)**
  - prizes and rewards (selected):
    - „**Cristal Brussels Prize 2006, 2010 and 2013**" for the most active and successful Polish SME participating in FP6 and FP7
    - reward for the **high performance in R&D projects for EDA**
  - **membership in international bodies**:
    - Public Safety Communications Europe (PSCE)
    - Integrated Mission Group for Security (IMG-S)

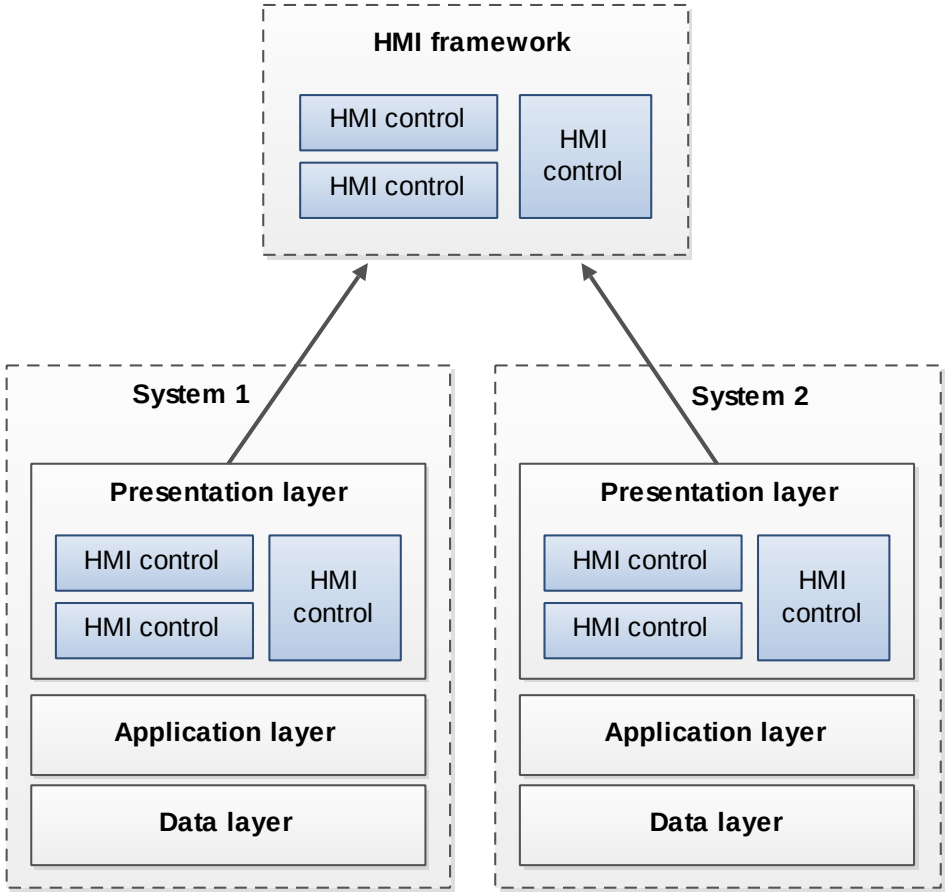
ESA projects

- **HMI** - The technology framework for the development of modular, portable and adaptive Human-Machine Interfaces in ground segment software products (03/2014 – 06/2015)
- **Gaia - GOSA** – An interactive service for asteroids follow-up observations (01/2015 – 01/2017)
- **INSPECTOR** – Integrated space components test platform (01/2015 – 06/2016)
- **SPACEMAN** – A SpaceWire network management tool (01/2014 – 03/2015)
- **SPACER** – Implementation and validation of the SpaceWire-R protocol (02/2015 – 04/2016)
- Study on demand for precise and legal time services distributed via the Galileo system including development of research methodology (11/2013 – 03/2015)

Agenda

- Company information
- Technical objectives
- Requirements
- Selected implemented controls
- Engineering approach
- Summary and future work

HMI idea



HMI decomposition view

- The main idea of the HMI Framework is to create UI control definitions independent of the source code, which allow defining once the required UI controls and use them in several applications, systems and hardware architectures although the implementations requirements differ.
- HMI framework enables a user to define a graphical interface through a common XML file for mobile and desktop platforms.

Project objectives

- Identification of key needs of the space industry in domain of HMI
- Survey of existing software, libraries, APIs (Application Programmer Interfaces) and standards
- Creation of common methodology for design of a space related HMIs
- Design of a reference software framework
- Reference implementation of such framework
- Execution of selected test case scenario using this framework
- Gathering and analysis of observations and lessons learned

Agenda

- Company information
- Technical objectives
- Requirements
- Selected implemented controls
- Engineering approach
- Summary and future work

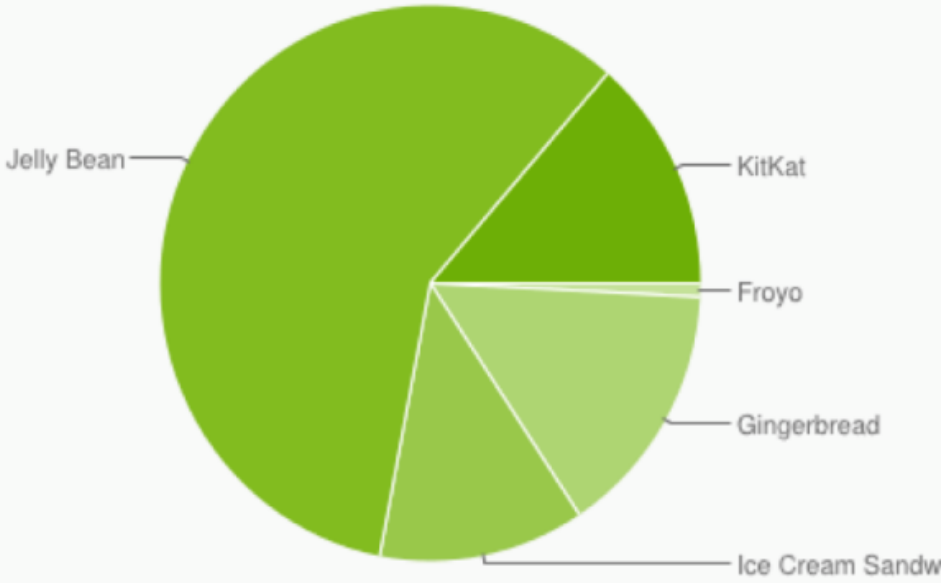
Requirement analysis

- To implement a control a developer needs to follow the requirements:
  - functional
  - non functional
  - UI and UX.
- Developer should know
  - control view (UI): how the control should look like
  - control behavior (UX): how the control should display or manage the data, how it should behave in normal conditions and during unexpected situations
  - control interface (API): attributes, methods, events and exceptions required by the control users

Mobile solution

Android OS fragmentation on the market

Version	Codename	API	Distribution
2.2	Froyo	8	0.8%
2.3.3 - 2.3.7	Gingerbread	10	14.9%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	12.3%
4.1.x	Jelly Bean	16	29.0%
4.2.x		17	19.1%
4.3		18	10.3%
4.4	KitKat	19	13.6%

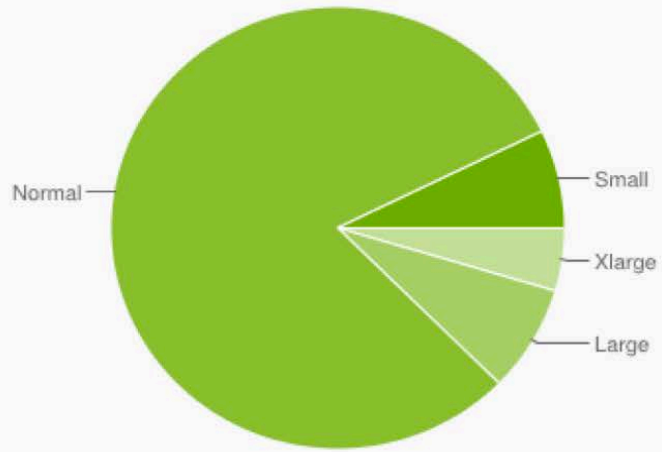


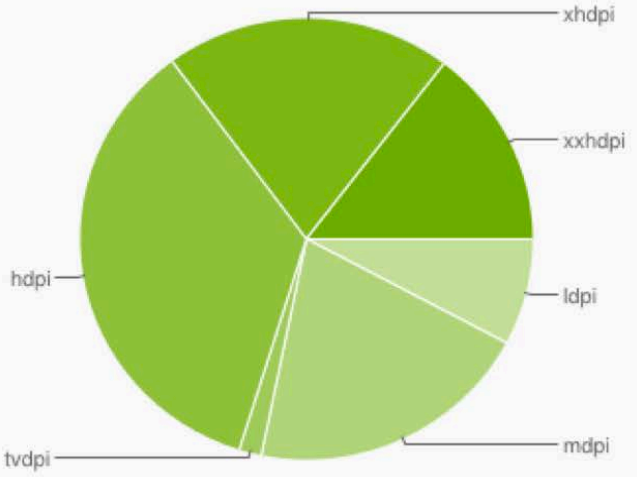
[Source: http://developer.android.com/about/dashboards/index.html]

Mobile solution

Screen resolutions of Android devices

	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	Total
Small	7.2%						7.2%
Normal		12.0%		34.2%	19.6%	14.6%	80.4%
Large	0.6%	4.4%	1.6%	0.6%	0.6%		7.8%
Xlarge		4.0%		0.3%	0.3%		4.6%
Total	7.8%	20.4%	1.6%	35.1%	20.5%	14.6%	





[Source: http://developer.android.com/about/dashboards/index.html]

Requirements analysis

Screen resolution	Number of available devices
1024 x 600 pixels	5
1024 x 720 pixels	1
1280 x 720 pixels	1
1280 x 800 pixels	36
1366 x 768 pixels	9
1920 x 1080 pixels	7
1920 x 1200 pixels	3
2560 x 1600 pixels	11

- Requirements for mobile devices:
  - applications and controls are manipulated with fingers; the size of a control should be adjusted to a fingerprint size
  - screen size - The HMI framework is designed for minimum 10in tablets
  - screen resolution - HMI framework is designed for tables with resolution of at least 1280x800px
  - connectivity - network bandwidth, connection type can change quite often on a mobile device
  - application restarts - mobile applications have different lifecycle than desktop applications, mobile operating system can close the application at any moment and restart it later
  - others: battery life, CPU power, storage capacity

Desktop solution

Operating System hardware requirements

Operating System	Version	CPU	RAM	HDD	Additional
Windows	7	1GHz 32-bit	1 GB	16 GB	DirectX 9 graphics device with WDDM 1.0 or higher driver
		1GHz 64-bit	2 GB	20 GB	
Red Hat Enterprise Linux*	6	1GHz 32-bit	512 MB	2 GB	
		1GHz 64-bit	1GB		
SUSE Linux Enterprise Server	11	-	512 MB per CPU	4 GB	
Ubuntu Long Term Support	12.04	1GHz Pentium 4	512 MB	5 GB	

Developer roles

- *HMI User* – a person who defines the final user interfaces with available controls in XML syntax and the HMI API. HMI user cycles steps the following:
  - Defining the UI in XML format
  - Running the application for generating the UI bundles
  - Testing the UI via demo project
- *Business logic developer* – a person who integrates the HMI framework and adds logical behaviors to the controls
  - The Business logic developer starts development where HMI user ends. Business logic developer's role is to provide the business logic behind the controls and create the final application. He/She uses the application output bundles and integrates them within the final system development environment.

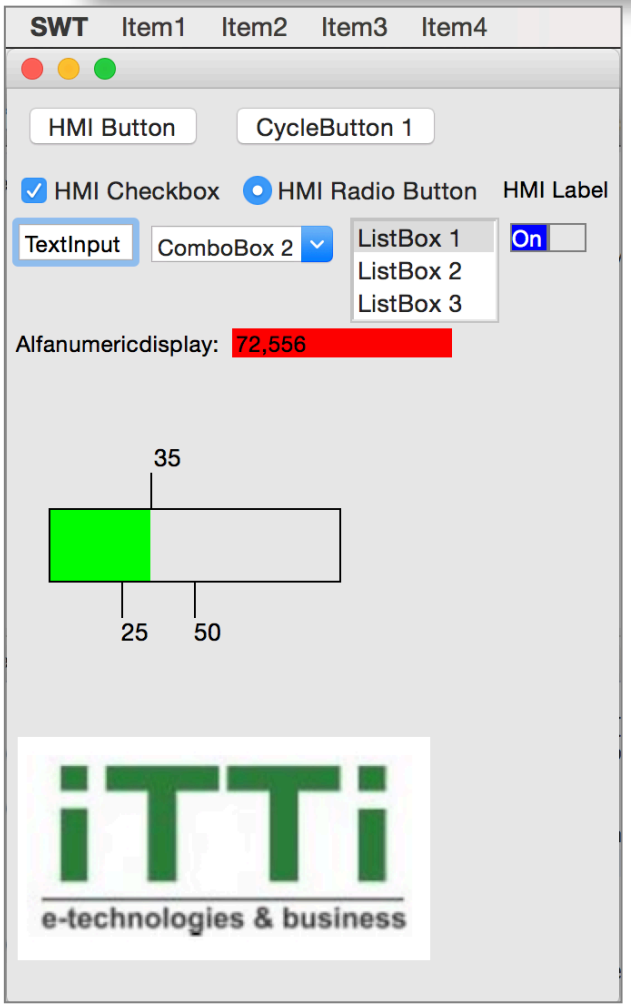
Agenda

- Company information
- Technical objectives
- Requirements
- Selected implemented controls
- Engineering approach
- Summary and future work

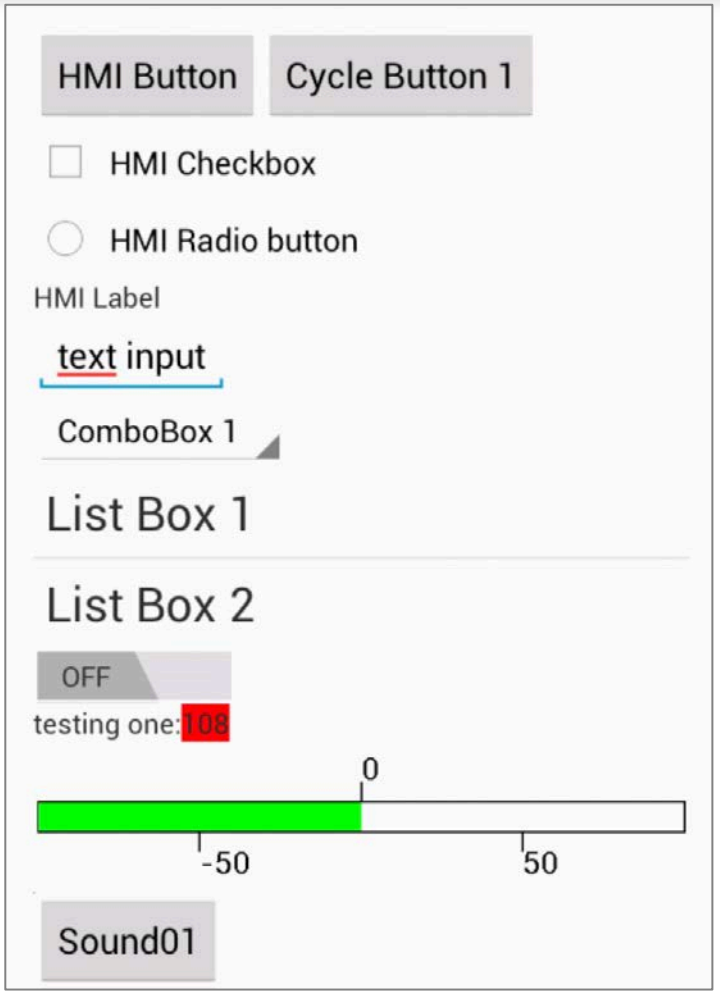
Implemented controls

- Basic control
  - Button
  - Cycle button
  - Text box
  - Label
  - Icon
  - Radio button
  - Check box
  - List box
  - Combo box
  - Menu item
  - Decision (only desktop),
  - Palette item (only desktop),
  - Sound
  - Switches
  - Gauges
- Complex control
  - Palette (only desktop)
  - Toolbar (only desktop)
  - Menu bar
  - Dialog
  - Pluto
  - SSM
  - Alphanumeric display

Basic mobile and desktop controls



Desktop view

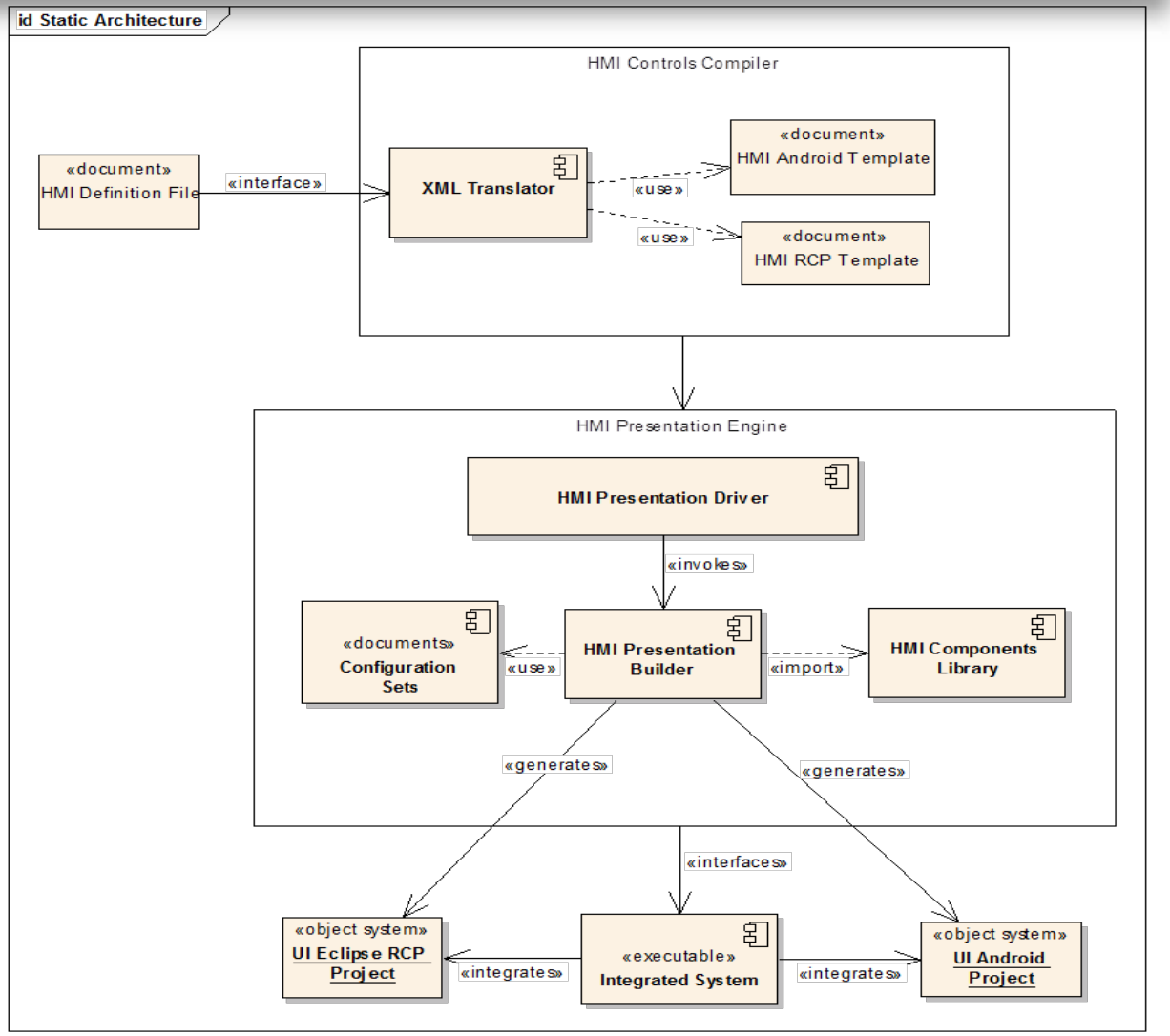


Mobile view

Agenda

- Company information
- Technical objectives
- Requirements
- Selected implemented controls
- Engineering approach
- Summary and future work

Single Definition Multiple Compilations



The process of compilation of an UI control definition

HMI Design and Implementation

- In HMI we use 3 different types of XML descriptions of the UI:
  - User oriented HMI XML following the HMI Controls API
  - Desktop user interface definition in XWT (XML Windowing Toolkit) format, following the Eclipse RCP (Rich Client Platform) UI format
  - Mobile user interface definition in Android layout files

„Gauge” control

- The „gauge” controls are defined by 5 values:
  - observed value - the value to be shown to the user
  - minimum value - lower limit for the observed value
  - minimum threshold - if the observed value goes below this threshold, the control notifies a user
  - maximum threshold - if the observed value goes above the threshold, the control notifies a user
  - maximum value - the maximum value that the control can show, if the value goes above this level the controls returns an exception

The „gauge” XML definitions

User oriented	Android specific	Desktop (XWT) specific
<Panel> <Gauge highValue="100" lowValue="0" highThreshold="50" lowThreshold="30" value="35" /> </Panel>	<pl.com.itti.hmi.mobile.controls.HGauge android:id=„@+id/gauge1” android layout_width=„fill_parent” android:layout_height=„wrap_content” hmi:label=„HMI Gauge” hmi:max=„100” hmi:maxThreshold=„50” hmi:min=„-100” hmi:minThreshold=„-50”>	<?xml version="1.0" encoding="UTF-8"?> <Shell xmlns:x="http://www.eclipse.org/xwt" xmlns:hmi="clr-namespace:pl.com.itti.hmi.desktop" xmlns="http://www.eclipse.org/xwt/presentation"> <Shell.layout> <RowLayout/> </Shell.layout> <hmi:HGauge highValue="100" lowValue="0" highThreshold="50” lowThreshold="30" </Shell>

Model and business logic implementation

- Implementation of a new control – the common part:
  - Define the control interface in *pl.com.itti.hmi.api.interfaces*, by extension of either *BasicControl* or *ComplexControl* ones.
  - Develop default implementation for the interfaces in *pl.com.itti.hmi.api.controls* package. This class will act as a model for your control.
  - Register the control in the singleton *ControlManager* instance, which is a container that stores all controls by their name or id.
  - Check *pl.com.itti.hmi.api.events* package for suitable listener and event classes. If no suitable listener is found a new one must be created from scratch. At least one listener is required to notify view classes about control state change.
  - The *pl.com.itti.hmi.api.exceptions* package is responsible of exceptions mechanisms. If there is no exception satisfying the need, create a new one extending the Exception class contained in the same package.

Presentation Engine Development (1/3)

- In order to develop a desktop control:
  - Create a class which extends Composite class (typical SWT (Standard Widget Toolkit) step).
  - Create and store data model for the control. Use proper data model for the control.
  - Implement the same interface as the data model. Delegate all methods to the data model class. This step enables to use setters and getters from XML in XWT. The framework will call and set default values for the control.
  - Register your class as a listener for data model change. Add proper behaviour for the control.

Presentation Engine Development (2/3)

- The process in Android :
  - Define possible attributes for the XML widget representation
  - Create a class extending the most similar widget
  - Create and store data model for the control. Use proper model for the control
  - Parse input parameters. To parse BasicControl attributes one can use the BasicControlParametersParser class provided in the framework
  - Implement the same interface as the data model. Delegate all methods to the model
  - Register listeners to react to model changes

Presentation Engine Development (3/3)

Step	Desktop	Mobile
1	Define possible attributes for the XML widget representation	
2	Create a class which extends Composite class (typical SWT (Standard Widget Toolkit) step)	Create a class extending the most similar widget
3	Create and store data model for the control	
4	Parse input parameters	
5	Implement the same interface as the data model. Delegate all methods to the data model class	
6	Register your class as a listener for data model change. Add proper behaviour for the control	Register listeners to react to model changes

Agenda

- Company information
- Technical objectives
- Requirements
- Selected implemented controls
- Engineering approach
- Summary and future work

Summary

- Provides reusable HMI controls and thus
  - Reduce redundant work
  - Speeds up development process
- Provides „of the shelf” library of HMI controls and thus
  - Makes applications much more error resistant
  - Enforces similarities in applications look & feel, which help end-users decrease the learning curve on how to use them
- Separates HMI design from application development and thus
  - Allows HMI designers to focus on ergonomic aspects
  - Allows programmers to focus on application performance and code quality

Future work

- Layouts – we need to define common layout for Eclipse RCP and Android applications
- Android 5.x compatibility test and framework optimization for Android RunTime (ART)

Thank you for your attention

Contact persons:

Joanna Modławska and Michał Tanaś
e-mails: *joanna.modlawska@itti.com.pl*,
michal.tanas@itti.com.pl