ES Software Engineering

Lecture 6

Implementation and testing

KATEDRA INŻYNIERII KOMPUTEROWEJ

Sponsor specjalności

In the previous lecture

1. Deployment diagram
2. Modeling Use Cases
3. Application logic design
4. Data Interfaces

Plan of the lecture

1. Use case modeling:

- application logic design
- user interface design
- testing

2. From design to implementation

- implementation
- framework
- documentation
- reverse engineering

KATEDRA INŻYNIERII KOMPUTEROWEJ

intel

Sponsor specjalności

User interface

Main rules of user interface design:

- **Consistency** – intuitive and user friendly
  - Interface similar to other application interfaces
  - Separate parts of interface similar to each other
  - All forms and reports with the same glossary, format and navigation
- **Awareness of content** – user should know what is seen and what can be done
  - Title of each window identifies its content
  - Visible path to any place (i.e. menu)
  - Clear icons and buttons, clear messages in windows
  - Different format (fonts, colors)

User interface

Main rules of user interface design:

- **Aesthetics** – balance between functionality and visual attractiveness
  - White space amount fitted to user's experience (50% for new, 10% for experienced)
  - Not to much fields to fill
  - Form size fitted to user processing ability
  - 8-10 pts fonts; sanserif fonts forms and serif fonts reports; max 2 different fonts
  - Mild colors in graphics (contrasts make it hard to stay in focus)
  - Color-blind users

User interface

Main rules of user interface design:

- **User experience** –ease to learn for new users and operational for experienced users
  - Ease for a new user and should offer automation for experiences users
  - Up to 3 levels of main menu and 2 levels of submenus
  - Up to 7 menu items (otherwise grouped items)
  - Fast access to frequently used items
  - Configurable menus and toolbox
  - Tooltips and application help

User interface

Main rules of user interface design:

- **Effort minimization** – as little steps for a user to complete a task as possible
  - Mouse click counts minimized to 3
  - As little keyboard usage as possible
  - Automation of repeating frequently used functions
  - Simple operations grouped to a single one

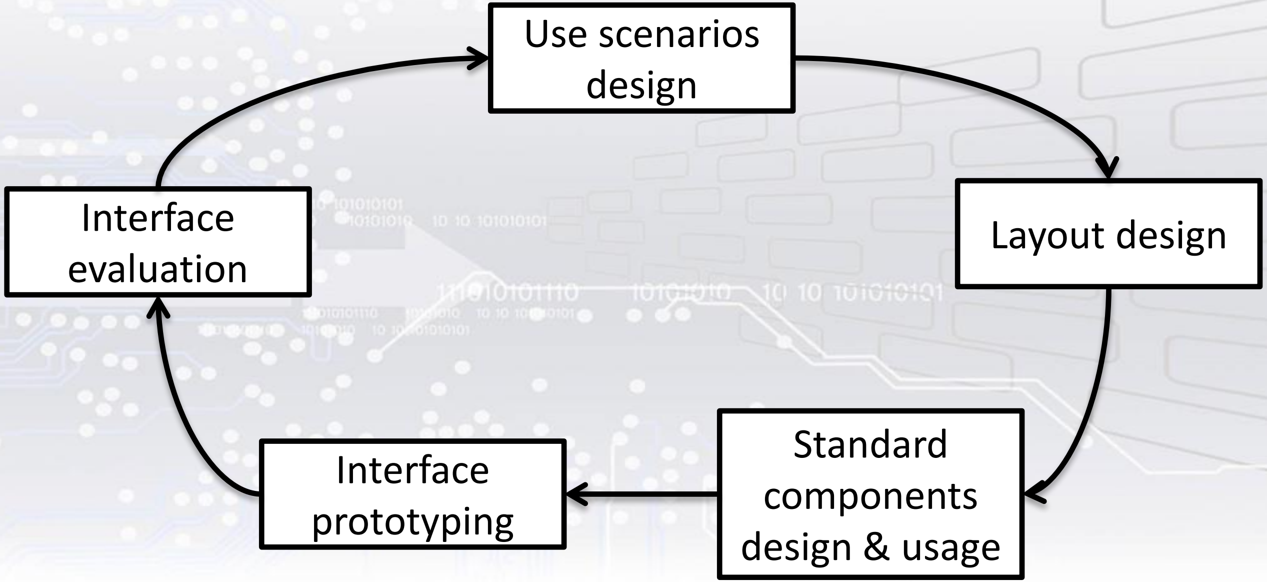
User interface

User characteristics:

- User role
- Objectives
- Properties
  - organization position
  - ability to decide
  - experience and skills
  - age
  - level of education
  - constraints
- Critical success factors
  - needs and opportunities
  - preferences and exclusions

User interface

Process of user interface design:



Use scenarios design

Layout design

Standard components design & usage

Interface prototyping

Interface evaluation

User interface

Use scenario (a description of steps that a user should go to complete a task):

- Foundation: use case model, sequence diagrams
- Only main scenarios considered
- Presentation: text description with numbered steps

User interface

Standard user interface components:

- Standard components are used in many forms and reports
- Standard components are taken from some framework library
- Missing components are designed by developers

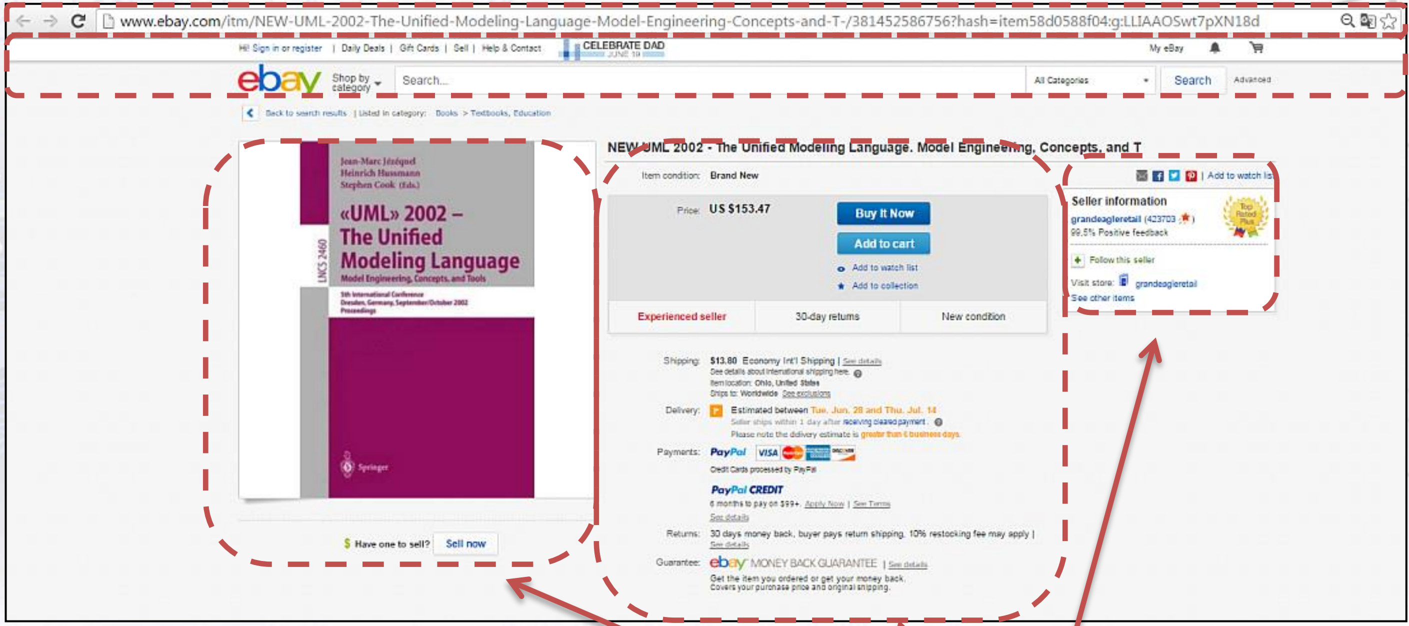
User interface

User interface evaluation:

- Heuristic evaluation
  - check guidelines
  - expert evaluation
- Evaluation by a developer and a user together
- User observation

User interface

Layout – interface visual organization


www.ebay.com/itm/NEW-UML-2002-The-Unified-Modeling-Language-Model-Engineering-Concepts-and-T-/381452586756?hash=item58d0588f04:g:LLIAAOSwt7pXN18d
Hi! Sign in or register | Daily Deals | Gift Cards | Sell | Help & Contact
CELEBRATE DAD JUNE 19
My eBay
ebay
Shop by category
Search...
All Categories
Search
Advanced
Back to search results | Listed in category: Books > Textbooks, Education
Jean-Marc Jézéquel
Heinrich Hussmann
Stephen Cook (Eds.)
«UML» 2002 – The Unified Modeling Language
Model Engineering, Concepts, and Tools
5th International Conference
Dresden, Germany, September/October 2002
Proceedings
LNCS 2460
Springer
Have one to sell? Sell now
Buy It Now
Add to cart
Add to watch list
Add to collection
Add to watch list
Top Rated Plus
Follow this seller
See other items
See details
See exclusions
Apply Now | See Terms
See details
See details
See details

NEW UML 2002 - The Unified Modeling Language. Model Engineering, Concepts, and T

Item condition: Brand New

Price: US $153.47

Experienced seller | 30-day returns | New condition

Seller information
grandeagleretail (423703 ★)
99.5% Positive feedback
Visit store: grandeagleretail

Shipping: $13.80 Economy Int'l Shipping |
See details about international shipping here.
Item location: Ohio, United States
Ships to: Worldwide

Delivery: Estimated between Tue. Jun. 28 and Thu. Jul. 14
Seller ships within 1 day after receiving cleared payment.
Please note the delivery estimate is greater than 6 business days.

Payments: PayPal VISA
Credit Cards processed by PayPal
PayPal CREDIT
6 months to pay on $99+.

Returns: 30 days money back, buyer pays return shipping. 10% restocking fee may apply |

Guarantee: ebay MONEY BACK GUARANTEE |
Get the item you ordered or get your money back.
Covers your purchase price and original shipping.

Navigation area (out of design area)

Navigation area

Information area

User interface

Layout guidelines

- information and navigation area with clear frontiers
- each area with clear purpose
- each area with only such information that is within its purpose
- User processing order (area importance):
  - from top do bottom
  - from left to right

User interface

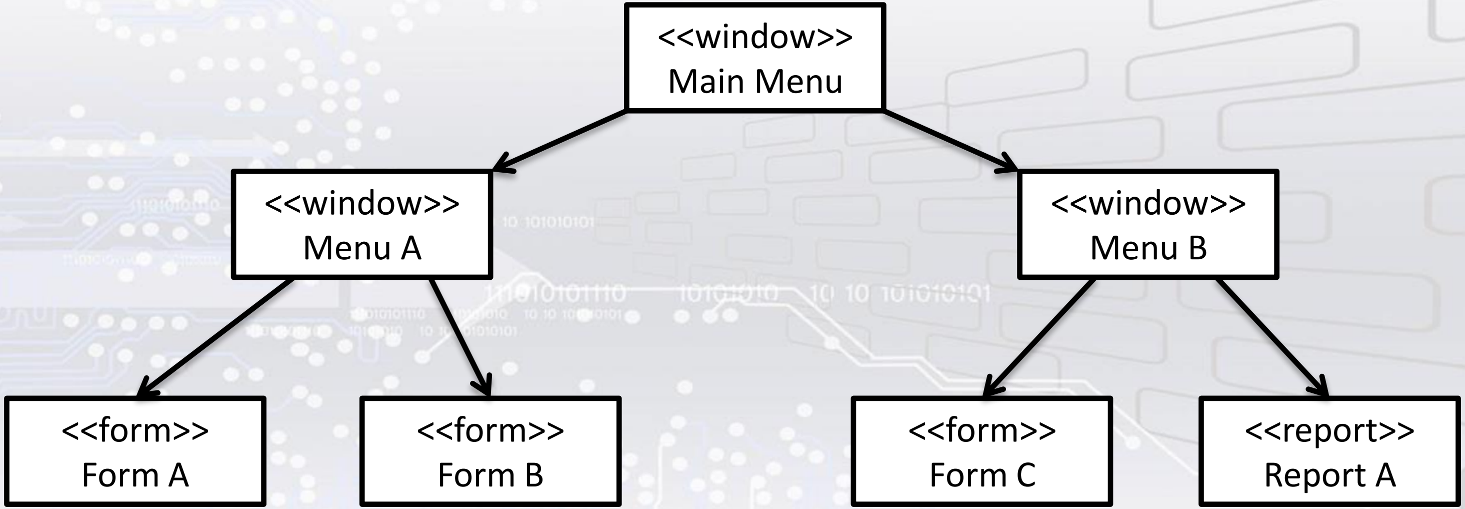
User interface design:

1. Window Navigation Diagram (NWD)
2. Mockup and Storyboard
3. HTML or target programming language prototyping:

- HTML
  - interactive prototype
  - fast to achieve
  - not precisely models a target interface
- Target language
  - interactive prototype
  - slow to achieve
  - precisely models a target interface

User interface

Window Navigation Diagram

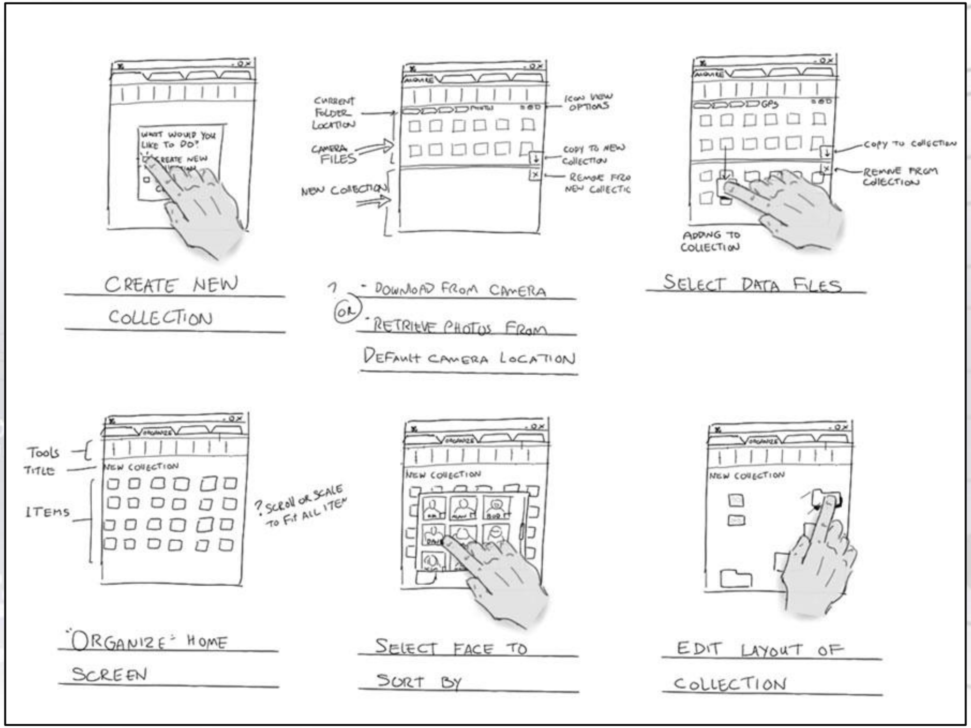


User interface

Storyboard – comes from film and multimedia industries and shows how does a user interface respond to user's actions.

Example 1 (Microsoft [5])

„a storyboard showing a creation of a new collection of items":



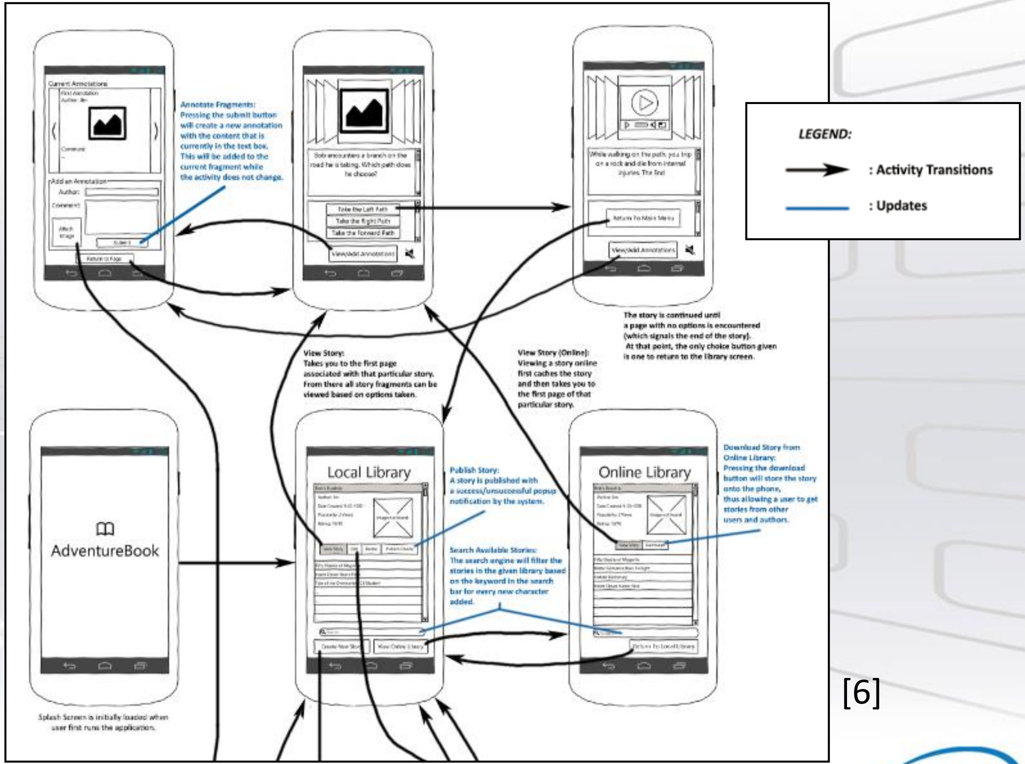
[5]

User interface

Storyboard – comes from film and multimedia industries and shows how does a user interface respond to user's actions.

Example 2 (Github [6])

a part of „*Main UI Layout*":



Annotate Fragments:
Pressing the submit button will create a new annotation with the content that is currently in the text box. This will be added to the current fragment while the activity does not change.

The story is continued until a page with no options is encountered (which signals the end of the story). At that point, the only choice button given is one to return to the library screen.

View Story:
Takes you to the first page associated with that particular story. From there all story fragments can be viewed based on options taken.

View Story (Online):
Viewing a story online first caches the story and then takes you to the first page of that particular story.

Publish Story:
A story is published with a success/unsuccessful popup notification by the system.

Search Available Stories:
The search engine will filter the stories in the given library based on the keyword in the search bar for every new character added.

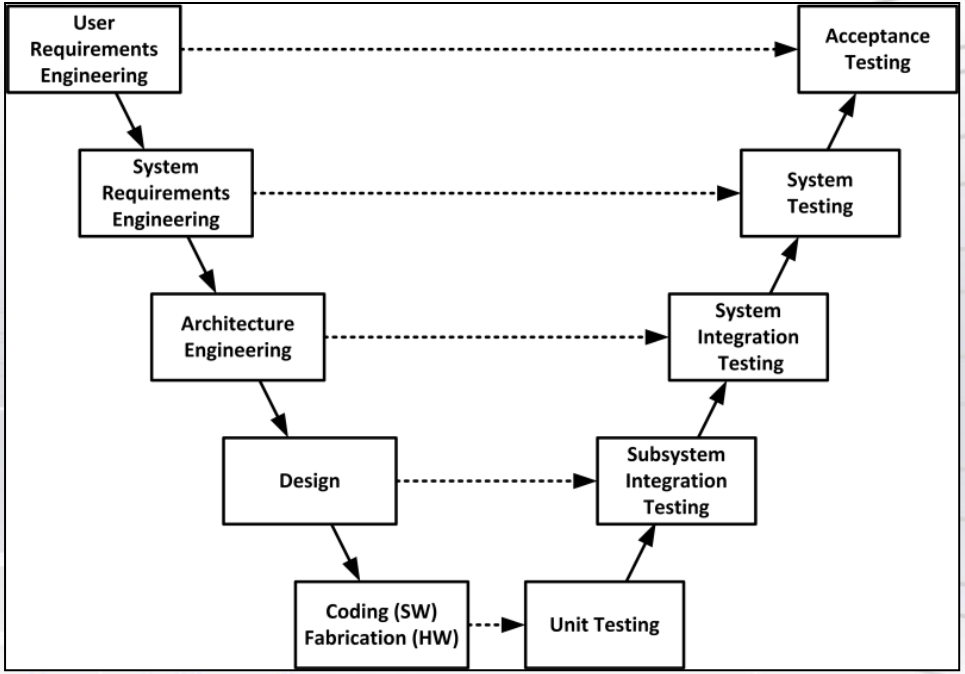
Download Story from Online Library:
Pressing the download button will store the story onto the phone, thus allowing a user to get stories from other users and authors.

Splash Screen is initially loaded when user first runs the application.

[6]

Testing and maintenance

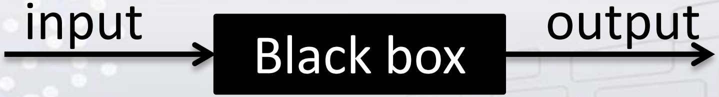
- V-model
- Unit testing
- Integration testing
- System testing
- Acceptance testing
- Maintenance strategy


User Requirements Engineering
Acceptance Testing
System Requirements Engineering
System Testing
Architecture Engineering
System Integration Testing
Design
Subsystem Integration Testing
Coding (SW) Fabrication (HW)
Unit Testing

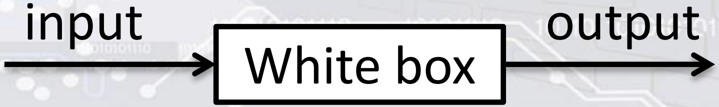
Testing and maintenance

Unit testing methods

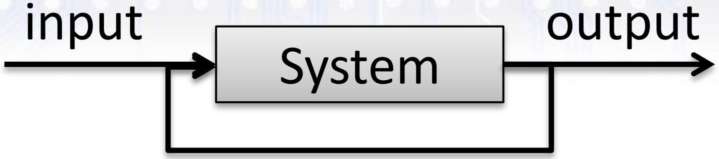
- „black-box" methods (Monte-Carlo, error seeding)



- „white-box" methods („transparent-box")



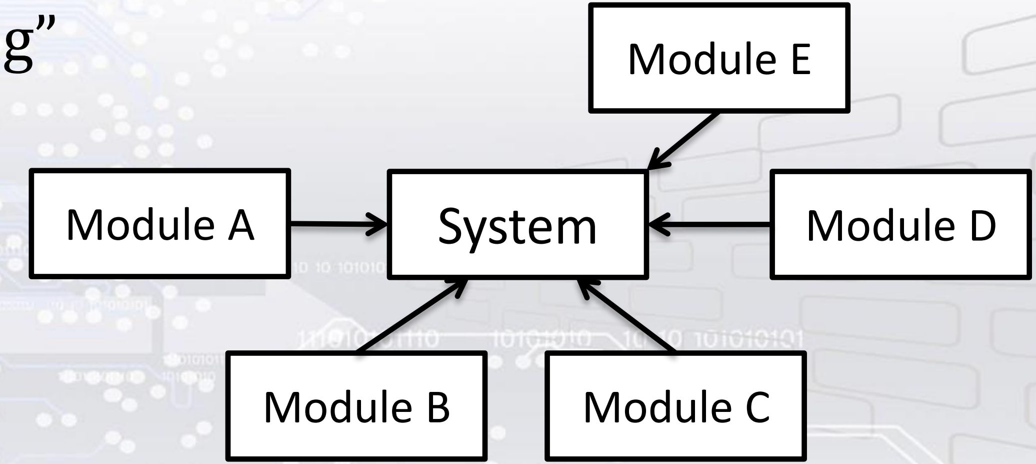
- Test automation (test repeat)



Testing and maintenance

Integration methods

- „Big bang”



Advantage: Each module is finished before integration testing has started

Testing and maintenance

Integration methods

- „Big bang”

Disadvantages:

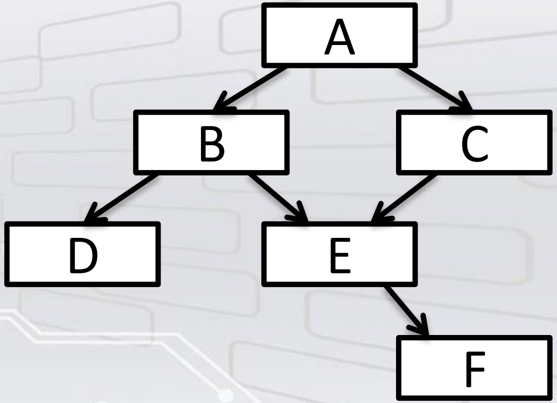
- Time
- Errors difficult to locate because of a late integration
- Integration of all modules at the same time increases risk of critical error

Testing and maintenance

Integration methods

- Bottom-Up Incremental

Step	Tested module	Additional module needed
1	F	F controller
2	E, F	E controller
3	D	D controller
4	B, D, E, F	B controller
5	C, D, E, F	C controller
6	Whole	-

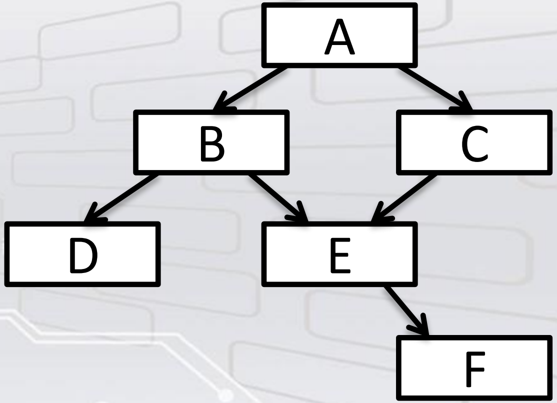


Testing and maintenance

Integration methods

- Top-Down Incremental

Step	Tested module	Additional module needed
1	A	Stub for B, C
2	A, B	Stub for C, D, E
3	A, B, C	Stub for D, E
4	A, B, C, D	Stub for E
5	A, B, C, D, E	Stub for F
6	Whole	-



Testing and maintenance

System testing

System category	Test for	Activity
Rich functionality system	usability	Functionality check
Data processing system	data size	Large data processing
Real time system	speed	Time measure
	capability	High data frequency
Interactive system	usage	Learnability and operability check
High security system	security	Security break trial
High memory usage system	memory consumption	Memory consumption measurement

Testing and maintenance

System testing

System category	Test for	Activity
Configurable system	configuration	Various configuration trial
New system version	compatibility	Testing after each release
Installable system	installation	Installation trial
High-reliability system	reliability	Reliability statistics measurement
High error-tolerance system	recovery	Error simulations
Administration needed system	usability	Personnel observation
	documentation	Documentation evaluation
Decision making system	procedures	Checking procedures executed by people

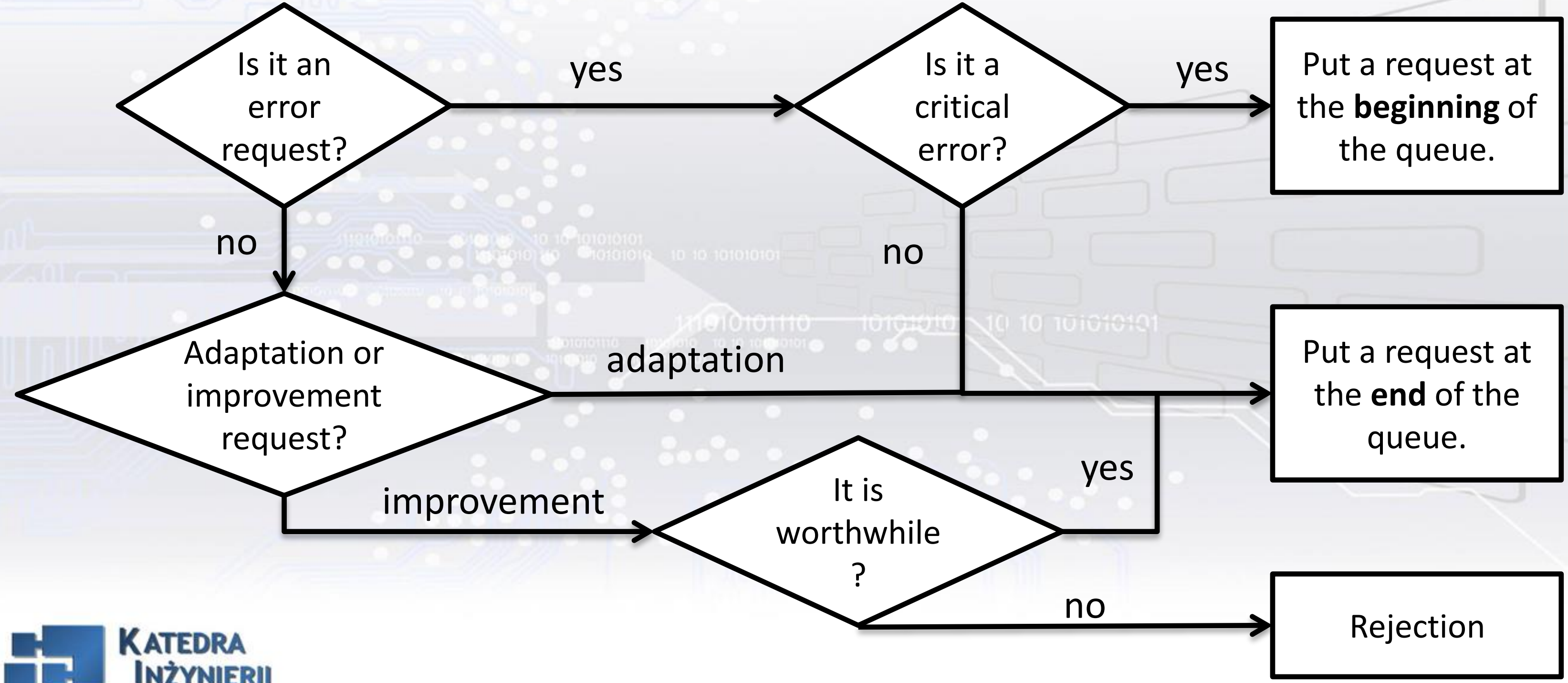
Testing and maintenance

Acceptation testing

Alpha testing	Beta testing
Performed by developers	Performed by customers
Sometimes performed by Independent Testing Team	Never performed by Independent Testing Team
Not open to the market and public	Always open to the market and public
Performed in virtual environment	Performed in real environment
Kind of White Box and Black Box Testing	Kind of Black Box Testing only

Testing and maintenance

Maintenance


Is it an error request?
yes
Is it a critical error?
yes
Put a request at the **beginning** of the queue.
no
no
Adaptation or improvement request?
adaptation
Put a request at the **end** of the queue.
improvement
It is worthwhile ?
yes
no
Rejection

From design to implementation

Passing from design to implementation

- Choosing the technology
- Choosing the framework
- Programming vs. script language
- Automatic code generation
- Method implementation
- Mapping abstract components to concrete classes
- Code documentation
- Reverse engineering

From design to implementation

Choosing the technology

- Programming language
- Programming environment
- User interface technology
- Database technology
- Target execution environment

From design to implementation

Technology constraints

- Price
- License
- Knowledge (of language and tools)
- Existing parts of systems
- User requirements
- other requirements

From design to implementation

Choosing language

Object-oriented programming languages:

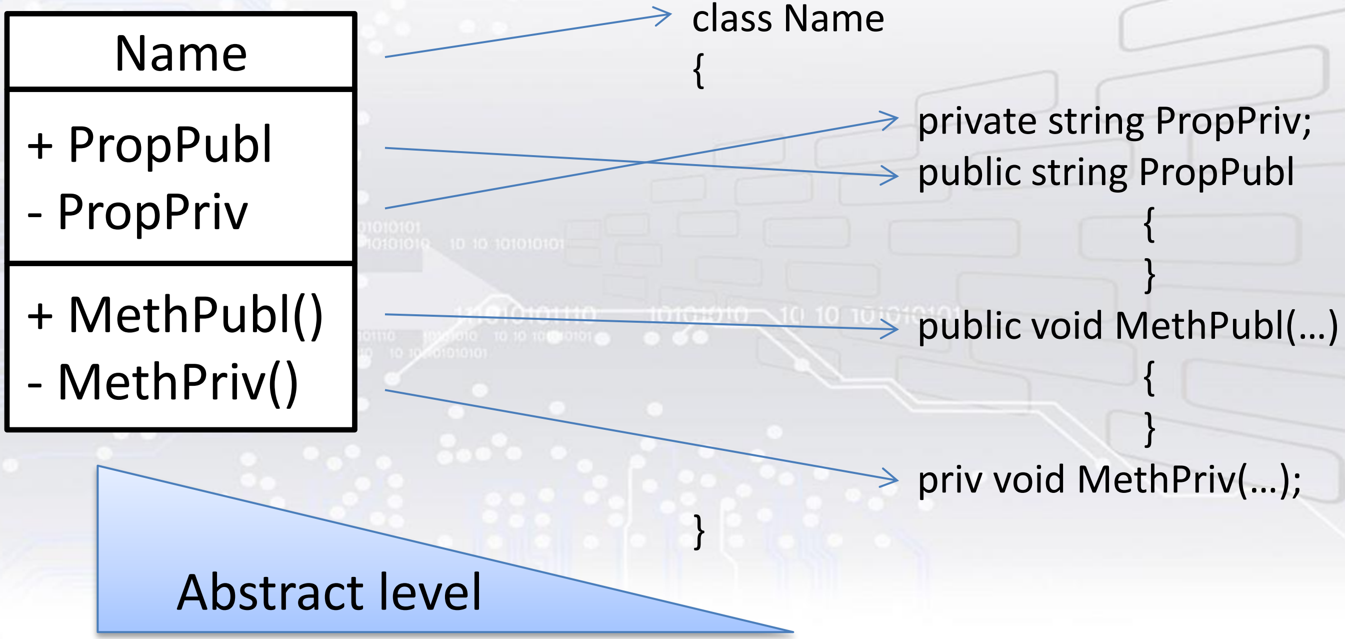
- data abstraction
- encapsulation
- modularity
- polymorphism
- inheritance

Script languages:

- weak typing
- easy fitting to external components
- portability
- slower execution

From design to implementation

Synthesis – automatic code generation



```
class Name
{
          private string PropPriv;
          public string PropPubl
                    {
                    }
          public void MethPubl(…)
                    {
                    }
          priv void MethPriv(…);
}
```

From design to implementation

Method implementation

- parameter list completion
- overloaded methods vs. default parameter values
- virtual vs. abstract methods
- data type and size
- optimization (final, sealed, const)

Abstract level

From design to implementation

Framework – a set of libraries and components which are the backbone to build an application.

Framework is not a library:

- Components of a framework are extendable with new user functionalities
- The default configuration of a framework is already useful, it is not merely a set of methods
- The structure of the framework is immutable – the default configuration cannot be modified
- Framework controls the operation of an application (*“Hollywood Principle"*)– in case of libraries – user manages the application

From design to implementation

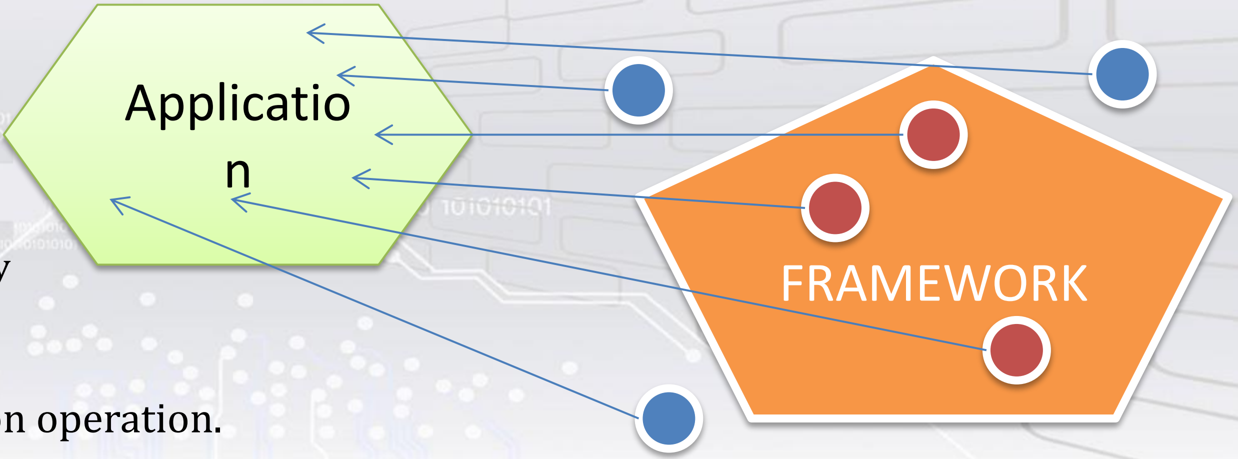
Framework – operating principle

"Hollywood Principle" – "Don't call us, we'll call you."

🔵 *cold point* – immutable
a point of the framework
not changed by a developer

🔴 hot point – extendable
concerning additional functionality
point of the framework

Both points manage application operation.



From design to implementation

Framework – features:

+ Reliability

+ Less code to write

+ Impose better quality of code

- Large work input (frameworks are complex)

- Lower application performance

- Weak documentation

- Lack of source code

- Unknown internal framework mechanism

From design to implementation

Code documentation

- Self-documentation – meaningful names
- Special documentation comments (documentation generation)
- Code structure overview description:
- files and folders hierarchy
- component list
- class reference

From design to implementation

Reverse engineering:

- Support of undocumented peripherals
- Security audit
- Copyright and patent issues
- Decompilation
- Analysis of messages
- Adapting to a multilingual version

KATEDRA INŻYNIERII KOMPUTEROWEJ

intel
Sponsor specjalności

Bibliography

[1] Jarosław Kuchta, *User interface design*, Embedded Systems Software Engineering, 2015

[2] Dennis A., Wixom B.H., TegardenD., *Systems Analysis & Design. An Object-Oriented Approach with UML*, John Wiley and Sons, USA, 2002

[3] Rolf Hennicker, Nora Koch: *Modeling the User Interface of Web Applications with UML*(PDF)

[4] Coad P., Yourdon E.: *Object-oriented design (book)*

[5] https://msdn.microsoft.com/en-us/library/windows/desktop/ff800706.aspx

[6] https://github.com/CMPUT301F13T11/Team11AdventureBook/wiki/User-Interface-Mockup-and-Storyboard

[7] Jarosław Kuchta, *Testowanie i pielęgnacja*, Embedded Systems Software Engineering, 2015

[8] Roger S. Pressman: *Software Engineering. A Practitioner's Approach* (book, PDF)

[9] http://testingbasicinterviewquestions.blogspot.com/2015/03/best-difference-between-alpha-and-beta.html

[10] Jarosław Kuchta, *Passing from design to implementation,* Embedded Systems Software Engineering, 2015

[11] Roger S. Pressman: *Software Engineering. A Practitioner's Approach* (book, PDF)

[12] Coad, Yourdon: *Object-Oriented Programming* (book)