

**Partners Technical Report**

**WP8.2 Defining metrics for PT&SCHE**

**D8.2.5**

**PROCEDURE FOR DEVELOPMENT OF CURRICULA OF SHORT CYCLE PROGRAMMES AT BELGRADE METROPOLITAN UNIVERSITY**

|  |  |
| --- | --- |
| Project Acronym: | PT&SCHE |
| Project full title: | Introduction of part-time and short-cycle studies in Serbia |
| Project No: | 561655-EPP-1-2015-1-EE-EPPKA2-CBHE-SP(2015-3431/001-001) |
| Funding Scheme: | ERASMUS+ |
| Coordinator: | TLU – Tallinn University |
| Project start date: | October 15, 2015 |
| Project duration: | 36 months |

|  |  |
| --- | --- |
| Abstract | This is QA document describes two procedures: 1) for development and acceptance of SCHE programs, and 2) for implementation and assessment of SCHE programs. The document also describes major activities of these procedures and should be used for initiation and development of new SCHE programs and for monitoring of their implementations. The aim of the document is to contribute to the quality of SCHE programs, by providing specifications how to prepare a good SCHE program in area of ICT. |

*"The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsi­ble for any use which may be made of the information contained therein."*

DOCUMENT CONTROL SHEET

|  |  |
| --- | --- |
| Title of Document: | D8.2.5 Procedure for development of curricula of short cycle programs at Belgrade Metropolitan University |
| Work Package: | WP8. Quality plan |
| Last version date: | 17/03/2019 |
| Status : | Final |
| Document Version: | 2.0 |
| File Name | D8.2.5 Procedure for development of curricula of SCHE programs at BMU |
| Number of Pages | 36 |
| Dissemination Level | Institutional |

VERSIONING AND CONTRIBUTION HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Revision Description | Responsible Partner |
| 1.0 | 1.2.2017 | Fist Draft | Dragan Domazet, BMU |
| 1.1 | 1.4.2017 | Pilot version | Dragan Domazet, BMU |
| 2.0 | 10.12.2018 | Modification based on th experience gained during pilot implementation of SCHE Programming in Java | Dragan Domazet, BMU |
| 2.1 | 20.2.2019 | Adopted final version | Dragan Domazet, BMU |
|  |  |  |  |
|  |  |  |  |

**TABLE OF THE CONTENT**

1. Introduction 4

2. Development process of curricula for short cycle programs 4

3. Feasibility Study 4

3.1. Job market analysis 4

3.2. Survey of employers 5

3.3. Proposal for the target ICT 5

3.4. Decision making 5

3.4.1. Teaching-Scientific Council of the Faculty 5

3.4.2. Deans’ proposal 5

3.4.3. Employers Council of the Faculty 5

3.4.4. Rector’s decision of the job profile of the short-cycle program 5

4. Specification of the job profile 6

4.1. Relevant policy documents (EU, national, standrads) 6

4.1.1. European ICT Professional Profiles 6

4.1.2. The European e-Competence Framework 6

4.2. The role and competences of the job profile 9

4.2.1. The specification of the profile 9

4.2.2. e-competences required 9

4.3. The Body of Knowledge 10

4.3.1. The European Foundational ICT Body of Knowledge 11

4.3.2. The Body of Knowledge of the SCHE program for a job profile 15

5. Specification of the short-cycle program for a job profile 22

5.1. Organizational structure of a short cycle program 22

5.2. Acquiring e-competence 23

5.3. Mapping of BOM into e-competences 23

6. Courses of the short-cycle program 24

6.1. List of courses and their sequence 24

6.2. Syllabi of courses 25

7. Pedagogical approach for implementation of curriculum of a SCHE program 29

8. Implementation plan of a short-cycle program 32

9. The procedure of development and acceptance of a SCHE program 33

10. The procedure of implementation and analysis of SCHE programs 33

11. Quality assurance 34

11.1. Instructions to lecturers 34

11.2. Monitoring 34

11.3. Reports 35

11.4. Corrective actions 35

11.5. Students’ surveys 35

11.6. Employers’ surveys 35

12. Conclusions 36

13. References 36

# Introduction

This documents aims to provide the specification of QA activities of development and implementation of a SCHE program that affect its quality. It is a process centric document, as the specified processes for development and implementation of a SCHE program

# Development process of curricula for short cycle programs

Marketing department and Head of ProAcademia launch an initiative for a SCHE program, after a preliminary assessment of market needs of a job profile. If the dean of a relevant faculty accepts the initiative, he/she nominate a working group for preparation of a feasibility study for a SCHE program the needs to be developed for the target ICT profile. The working group will also need to perform other activities of the development process process of curricula for short-cycle programs. It consists of the following major activities:

1. Feasibility Study
2. Specification of the job profile
3. Specification of the short-cycle program for a job profile
4. Courses of the short-cycle program and their syllabi
5. Pedagogical approach for implementation of curriculum of a SCHE program

The following four sections briefly specify these major actions that must be taken to accomplish the each activity.

# Feasibility Study

## Job market analysis

The initiator of a SCHE program can be a professor, an assistant or dean of a faculty, the head of ProAcademia, the organizational unit in charge for SCHE programs at BMU. The initiator has to prepare a report on needs analysis that should include the following information:

* External reports of needs for some ICT jobs in Serbia, including the results of available surveys;
* Official reports of the National Employment Agency (Nacionalna služba za zapošljavanje)
* Analyses of the competition providing job profiles for the target ICT job
* Number of graduates that could take these ICT jobs
* Number of learners that completed short cycle programs for ICT jobs
* Existing needs for the target ICT job and predictions for next period of 3-5 years, given as number per year.

## Survey of employers

BMU should make a survey of target employers for candidate ICT jobs in order to find the most needed ICT job, from employers’ perspective.

## Proposal for the target ICT

The target ICT job profile is the one that has the highest priority crated based on the analysis done in 3.1 and 3.2. The proposal should also take into account financial aspects of implementation a new SCHE program.

## Decision making

### Teaching-Scientific Council of the Faculty

The initiator submits the proposal for the target ICT job profile for a SCHE program. The Teaching-Scientific Council of the Faculty analyses and discuss the proposal and make a conclusion - as a recommendation for the target job profile (yes or not).

### Deans’ proposal

Based on the recommendation of the Teaching-Scientific Council of the Faculty (section 2.1.4.1), the dean of the relevant faculty rejects the proposal, or accept it and prepare the official proposal for new SCHE program for a new ICT job profile.

Dean also has to provide an analysis of human recourse’s (available lectures) for realization of the proposed SCHE program and to include it in his/her official proposal.

### Employers Council of the Faculty

The Employers Council of the Faculty analyses the Deans’s proposal and generates it opinion regarding the proposal.

### Rector’s decision of the job profile of the short-cycle program

Taking into consideration of analysis and conclusion of the activities in 2.1.4.1-3, the Rector make a final decision to proceed with further planning and analyses or to stop further activities.

# Specification of the job profile

## Relevant policy documents (EU, national, standrads)

### European ICT Professional Profiles

Dean and the initiator have to prepare the specification of the job profile using **“European ICT Professional Profiles”,** CWA 16458, (Figure 1) as the base for specification of the target ICT job profile for the proposed SCHE program. The specification ahs to be a Generation 3 ICT profile

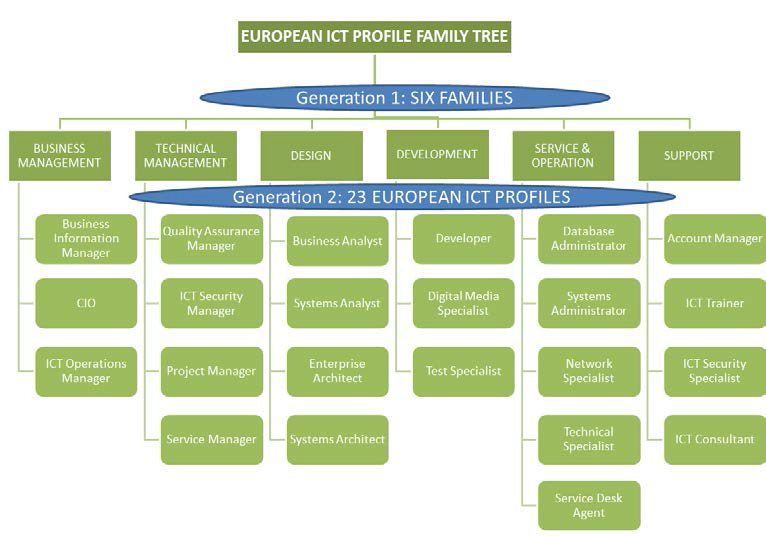


Figure 1 European ICT profile family tree (Generation 1 and 2)

### The European e-Competence Framework

Dean and the initiator have to include also the CWA (CEN Workshop Agreement) document: “**The European e-Competence Framework (e-CF) version 3.0**” that provides a reference of 40 competences as required and applied at the Information and Communication Technology (ICT) workplace (Figure 2), using *a common language for competences, skills and capability levels t*hat can be understood across Europe. As the first sector-specific implementation of the European Qualifications Framework (EQF), the e-CF was created for application by ICT service, user and supply companies, for managers and human resource (HR) departments, for education institutions and training bodies including higher education, for market watchers and policy makers, and other organisations in public and private sectors.

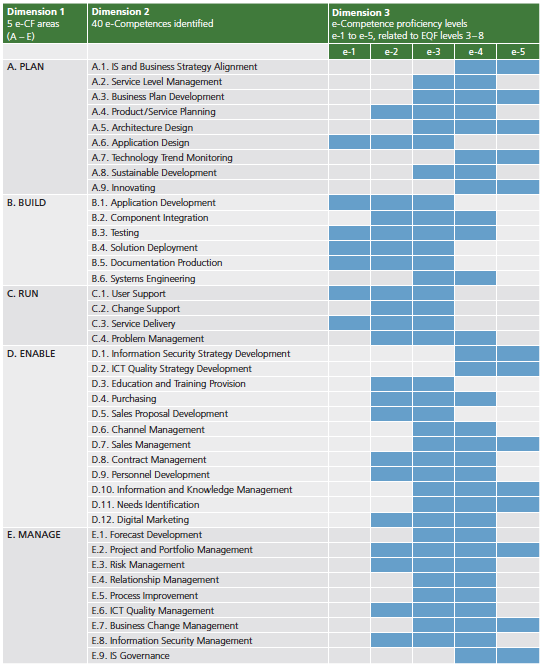


Figure 2 40 e-Competences defined by the European e-Competence Framework

Whilst competence definitions are explicitly assigned to dimension 2 and 3 and knowledge and skills samples appear in dimension 4 of the framework, attitude is embedded in all three dimensions. Figure 3 shows the table that relates EQF (European Qualification Framework) levels and e-CF levels.

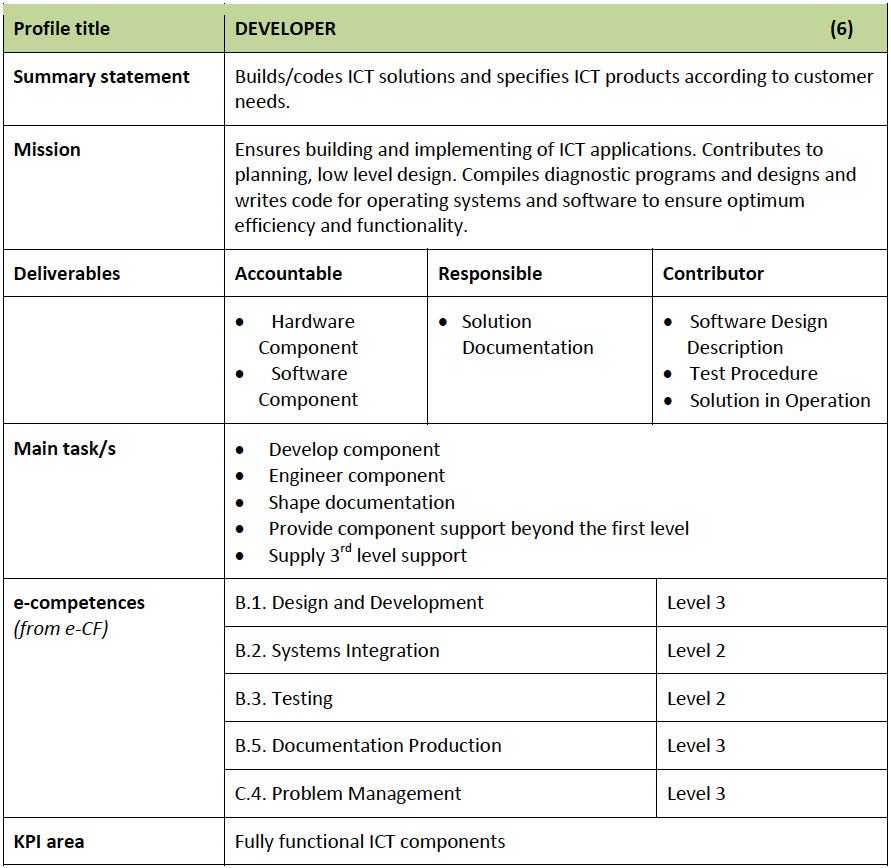
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EQF Levels | EQF | e-CF Levels | e-CF Levels descriptions | Typical Tasks |
| **8** | Knowledge at the most advanced frontier, the most advanced and specialised skills and techniques tosolve critical problems in research and/or innovation, demonstrating substantial authority, innovation, autonomy, scholarly or professional integrity. | **e-5** | **Principal**  Overall accountability and responsibility; recognised inside and outside the organisation for innovative solutions and for shaping the future using outstanding leading edge thinking and knowledge. | IS strategy or program management |
| **7** | Highly specialised knowledge, some of which is at the forefront of knowledge in a field of work or study, as the basis for original thinking, critical awareness of knowledge issues in a field and at the interface between different fields, specialised problem-solving skills in research and/or innovation to develop new knowledge and procedures and to integrate knowledge from different fields, managing and transforming work or study contexts that are complex, unpredictable and require new strategic approaches, taking responsibility for contributing to professional knowledge and practice and/or for reviewing the strategic performance of teams | **e-4** | **Lead Professional / Senior Manager**  Extensive scope of responsibilities deploying specialised integration capability in complex environments; full  responsibility for strategic development of staff working in unfamiliar and unpredictable situations | IS strategy/ holistic solutions |
| **6** | Advanced knowledge of a field of work or study, involving a critical understanding of theories and principles, advanced skills, demonstrating mastery and innovation in solving complex and unpredictable problems in a specialised field of work or study, management of complex technical or professional activities or projects, taking responsibility for decision-making in unpredictable work or study contexts, for continuing personal and group professional development. | **e-3** | **Senior Professional / Manager**  Respected for innovative methods and use of initiative in specific technical or business areas; providing leadership and taking responsibility for team performances and development in unpredictabl environments**.** | Consulting |
| **5** | Comprehensive, specialised, factual and theoretical knowledge within a field of work or study and an awareness of the boundaries of that knowledge, expertise in a comprehensive range of cognitive and practical skills in developing creative solutions to abstract problems, management and supervision in contexts where there is unpredictable change, reviewing and developing performance of self and others. | **e-2** | **Professional**  Operates with capability and ndependence in specified boundaries and may supervise others in this environment; conceptual and abstract model building using creative thinking; uses theoretical knowledge and practical skills to solve complex problems within a predictable and sometimes unpredictable context. | Concepts / Basic principles |
| **4** | Factual and theoretical knowledge in broad contexts within a field of work or study, expertise in a range of cognitive and practical skills in generating solutions to specific problems in a field of work or study, self-manageme nt within the guidelines of work or study contexts that are usually predictable, but are subject to change, supervising the routine work of others, taking some responsibility for the evaluation and improvement of work or study activities. |
| **3** | Knowledge of facts, principles, processes and general concepts, in a field of work or study, a range of cognitive and practical skills in accomplishing tasks. Problem solving with basic methods, tools, materials and information, responsibility for completion of tasks in work or study, adapting own behaviour to circumstances in solving problems. | **e-1** | Associate  Able to apply knowledge and skills to solve straight forward problems; responsible for own actions; operating in a stable environment. | Support / Service |

Figure 3 Relationships between EQF levels and e-CF levels

## The role and competences of the job profile

### The specification of the profile

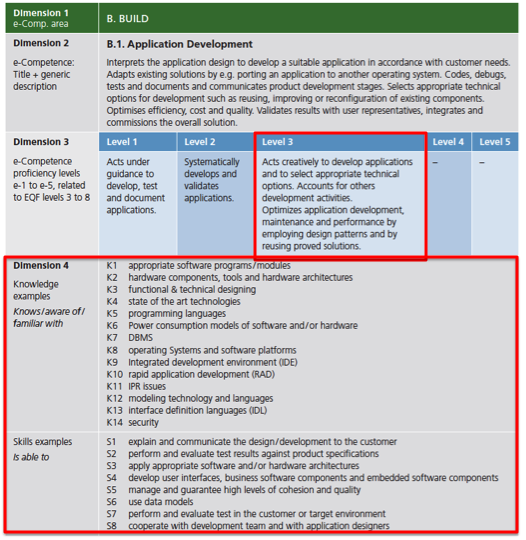
It is necessary to spcify job profile as suggested in **“European ICT Professional Profiles”,** such as presented in Figure 4.

Figure 4 Job profile specification of a Developer

### e-competences required

For each job profile (Figure 1), the **European e-Competence Framework 3.0** specifies a set of key e-competences, shown in Figure 2 with different proficiency levels (1-5). The document **“European ICT Professional Profiles”** specifies needed e-competences and their proficiency levels, for each job profile specified in Generation 2, as shown in Figure 4. The document **European e-Competence Framework 3.0** specifies each e-competence, for each proficiency level in a tabular form, as shown in Figure 5. Specification is split in four dimensions:

* Dimension 1 specifies *e-competence area*
* Dimension 2 specifies *e-competence title and generic description*
* Dimension 3 specifies *e-competence proficiency levels* (e-1 to e-5), that are related to EQF levels 3 to 8, as shown in Figure 3
* Dimension 4 specifies *required knowledge areas and skills*

Figure 5 Specification of e-competence B.1. Application Development

For each e-competence required for a job profile, as specified in case of Developer in Figure 4, it is necessary to shows appropriate specification, as shown in Figure 5.

## The Body of Knowledge

Specification of knowledge units and skills provided for each e-competence in the previous section is not enough to specify the curriculum for a short cycle program for a profile. The specifies required knowledge and skills are of very high level and need to be specified at lower levels. This is the mission of a Body of Knowledge of a study program. In our case we can use:

* **The Foundation ICT Body of Knowledge**, Version 1, 22 February 2015, a report prepared for the European Commission, DG Internal Market, Industry, Entrepreneurship and SMEs by the Service Contract: *e-Skills: Promotion of ICT Professionalism in Europe | No 290/PP/ENT/CIP/13/C/N01C011* prepared by Capgemini Consulting and Ernst & Young.
* **The Software Engineering Body of Knowledge – SWEBOK 3.0,** specified by the IEEE Computer Society - see P. Bourque and R.E. Fairley, eds., **Guide to the Software Engineering Body of Knowledge, Version 3.0**, IEEE Computer Society, 2014; www.swebok.org.

### The European Foundational ICT Body of Knowledge

The European Foundational ICT Body of Knowledge is the base-level knowledge required to enter the ICT profession and acts as the first point of reference for anyone interested in working in ICT’.

The ultimate objective is to create a recognised and supported Foundational ICT Body of Knowledge that:

* Serves as an entry point to get into ICT for anyone contemplating a career in ICT and entering from other professions or wanting to digitise their current job;
* Facilitates communication between and understanding of ICT professionals in Europe in whatever sector they are active, thereby reducing risks and strengthening ICT professionalism;
* Increases the supply and pool of ICT professionals and enhances the image of ICT.

The definition of an ICT Professional is defined, as someone who should:

* Possess a comprehensive and up-to-date understanding of a relevant body of knowledge;
* Demonstrate on-going commitment to professional development via an appropriate combination of qualifications, certifications, work experience, non-formal and / or informal education;
* Adhere to an agreed code of ethics / conduct and / or applicable regulatory practices; and
* Through competent practice deliver value for stakeholders.

Some of the key challenges for the near future are to:

* Ensure that as many ICT professionals as possible have the necessary relevant knowledge, skills and competence to deliver professional products and service in today’s digital economy;
* Improve the quality of the ICT profession;
* Close the ICT resource and skills gap;
* Enhance growth in digital jobs in Europe;
* Improve general ICT knowledge among professionals in other fields of expertise.

**The Foundational ICT Body of Knowledge**provides the base-level knowledge that ICT professionals require. However, considering the wide range of knowledge in the ICT field, it has to be intended as a “permissive model” where every ICT professional will acquire as much breadth as possible in terms of knowledge

In addition to the dimension of ICT core knowledge defined above, the European Foundational ICT Body of Knowledge consists of a second dimension of complementary base-level knowledge required to enter the ICT profession. This dimension includes cross-cutting knowledge that cannot be considered purely in relation to one ICT knowledge area but can be referred to, at different levels, in relation to all core knowledge areas, i.e.:

* **Legal, ethical, social and professional practices:** including this knowledge in the Foundational ICT Body of Knowledge serves to provide key reference points for everyone interested in the ICT profession, as they are strongly linked to the definition of the ICT profession itself. Legal, ethical, social and professional practices need to be addressed at different levels at different stages of professional development. Thevery nature of professional work means that some knowledge and skills are best developed through experience and that an understanding of complex issues, such as ethics, grows with maturity. Further development will be provided at a full professional level through participation in certification programs.
* **Soft skills:** including soft skills in the Foundational ICT Body of Knowledge provides a concrete contribution to the evolution of the ICT profession. Soft skills integrate the technical skills, providing a sound basis for developing “dual thinker” profiles, which are oriented towards team building, collaboration, negotiation, e-leadership, etc.
* **Emerging / disruptive technologies:** given the fast growth in the disruptive technologies of cloud, mobile, social and big data, which are predicted to constitute 40% of the global market and 98% of growth by 2020, and the expected creation of 4.4 million IT jobs globally to support big data – base- level knowledge should be provided to improve an understanding of these technologies and their impacts on business and society.

The BOK illustrated below (Figure 6) and expanded on in the following sections presents the taxonomy of **the high-level areas of knowledge** that represent the base level that starting ICT professionals should understand. These knowledge areas are then broken down and described in further detail, including with a general definition of the knowledge area, a detailed list foundational knowledge, reference to the e-CF, potential job profiles and examples of specific Bodies of Knowledge, certification and training opportunities.

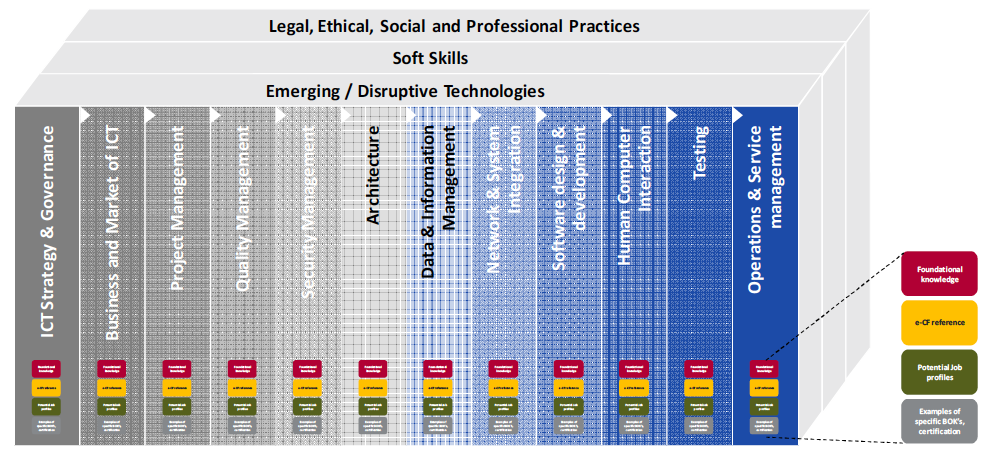


Figure 6 Taxonomy of Foundational ICT Body of Knowledge

This Version 1.0 of **the European Foundational ICT Body of Knowledge** presents the taxonomy of high-level areas of knowledge that represent the base level starting ICT professionals should understand.

The following section presents **12 Knowledge Areas**:

1. ICT Strategy & Governance
2. Business and Market of ICT
3. Project Management
4. Security Management
5. Quality Management
6. Architecture
7. Data and Information Management
8. Network and Systems Integration
9. Software Design and Development
10. Human Computer Interaction
11. Testing
12. Operations and Service Management.

Each **Knowledge Area is further detailed**, including a:

1. Definition of the Knowledge Area;
2. List of items required as foundational knowledge necessary under this Knowledge Area;
3. List of references to the e-Competence Framework (dimension 4: knowledge);
4. List of possible job profiles that require having an understanding of the Knowledge Area;
5. List of examples of specific Bodies of Knowledge, certification and training possibilities.

Figure 7 specifies the content, as an example, knowledge area *Software Design and Development.*

Knowledge Areas specified in the *European Foundational ICT Body of Knowledge* provide broader knowledge then specified for different job profiles specified in the *European e-Competence Framework 3.0.* We can notice it if we compare,specification of knowledge areas given in Figure 5 and Figure 7.

Unfortunately, the ICT Foundation Body of Knowledge does not provide yet lower levels of knowledge and it is not sufficient for a curriculum development. Therefore, additional extensions (sub-topics) of the Bodies of Knowledge are needed.



Figure 7 Software Design and Development Knowledge Area

### The Body of Knowledge of the SCHE program for a job profile

IEEE Computer Society specified two Bodies of Knowledge (BOK) that are relevant for ICT Profile Developer:

1. **Computer Science Curricula 2013** - Curriculum Guidelines for Undergraduate Degree Programs in Computer Science, December 20, 2013, The Joint Task Force on Computing Curricula of Association for Computing Machinery (ACM) and IEEE Computer Society
2. **SWEBOK 3**.0 – Guide to the Software Engineering Body of Knowledge, Editors Pierre Bourque, École de technologie supérieure (ÉTS) and Richard E. (Dick) Fairley, Software and Systems Engineering Associates (S2EA), IEEE Computer Society

Knowledge areas and topics from these two Bodies of Knowledge are to be selected according to specified of Knowledge Areas and e-competences required for ICT Profile Developer specified in previous sections. In order to illustrate this procedure, we will give in Tables 4.1 and 4.2 an example how to specify e-competences and their for proficiency levels in case of two ICT profiles that might be specified in Generation 3:

* **Junior Java Developer**, and
* **Java Developer**

*Table 4.1: Job description related to different e-competences*

|  |  |
| --- | --- |
|  | **JUNIOR JAVA DEVELOPER & JAVA DEVELOPER** |
| **e-competences** | **Job Description with** |
| B.1. Application Development | Interprets the application design to develop a suitable application in accordance with customer needs. Adapts existing solutions by e.g. porting an application to another operating system. Codes, debugs, tests and documents and communicates product development stages. Selects appropriate technical options for development such as reusing, improving or reconfiguration of existing components. Optimises efficiency, cost and quality. Validates results with user representatives, integrates and commissions the overall solution. |
| B.2. Component Integration | Integrates hardware, software or sub system components into an existing or a new system. Complies with established processes and procedures such as, configuration management and package maintenance. Takes into account the compatibility of existing and new modules to ensure system integrity, system interoperability and information security. Verifies and tests system capacity and performance and documentation of successful integration. |
| B.3.Testing | Constructs and executes systematic test procedures for ICT systems or customer usability requirements to establish compliance with design specifications. Ensures that new or revised components or systems perform to expectation. Ensures meeting of internal, external, national and international standards; including health and safety, usability, performance, reliability or compatibility. Produces documents and reports to evidence certification requirements. |
| B.5. Documentation Production | Produces documents describing products, services, components or applications to establish compliance with relevant documentation requirements. Selects appropriate style and media for presentation materials. Creates templates for document-management systems. Ensures that functions and features are documented in an appropriate way. Ensures that existing documents are valid and up to date. |
| C.4. Problem Management | Identifies and resolves the root cause of incidents. Takes a proactive approach to avoidance or identification of root cause of ICT problems. Deploys a knowledge system based on recurrence of common errors. Resolves or escalates incidents. Optimises system or component performance. |

*Table 4.2: e -Competence levels*

|  |  |  |
| --- | --- | --- |
|  | **JUNIOR JAVA DEVELOPER** | **JAVA DEVELOPER** |
| **e- competences** | **Level e-2** | **Level e-3** |
| B.1. Application Development | Systematically develops and validates applications. | Acts creatively to develop applications and to select appropriate technical options. Accounts for others development activities.  Optimizes application development, maintenance and performance by employing design patterns and by reusing proved solutions. |
| B.2. Component Integration | Acts systematically to identify compatibility of software and hardware specifications. Documents all activities during installation and records deviations and remedial activities. | As for Level e-2 |
| B.3.Testing | Organises test programs and builds scripts to stress test potential vulnerabilities. Records and reports outcomes providing analysis of results. | As for Level e-2 |
| B.5. Documentation Production | Determines documentation requirements taking into account the purpose and environment to which it applies. | Adapts the level of detail according to the objective of the documentation and the targeted population. |
| C.4. Problem Management | Identifies and classifies incident types and service interruptions. Records incidents cataloguing them by symptom and resolution. | Exploits specialist knowledge and in-depth understanding of the ICT infrastructure and problem management process to identify failures and resolve with minimum outage. Makes sound decisions in emotionally charged environments on appropriate action required to minimise business impact. Rapidly identifies failing component, selects alternatives such as repair, replace or reconfigure. |

Similarly, we can specify knowledge areas needed related to different e-competences, as shown in Table 4.3 and skills needed related to different e-competences, as shown in Table 4.4

*Table 4.3: Knowledge needed related to different e-competences*

|  |  |
| --- | --- |
|  | **JUNIOR JAVA DEVELOPER & JAVA DEVELOPER** |
| **e-competences** | **KNOWLEDGE: Knows/aware of/ familiar with / familiar with** |
| B.1. Application Development | K1 appropriate software programs/modules  K2 hardware components, tools and hardware architectures  K3 functional & technical designing  K4 state of the art technologies  K5 programming languages  K6 Power consumption models of software and/or hardware  K7 DBMS  K8 operating Systems and software platforms  K9 Integrated development environment (IDE)  K10 rapid application development (RAD)  K11 IPR issues  K12 modeling technology and languages  K13 interface definition languages (IDL)  K14 security |
| B.2. Component Integration | K1 old, existing and new hardware components/software programs/modules  K2 the impact that system integration has on existing system/organisation  K3 interfacing techniques between modules, systems and components  K4 integration testing techniques  K5 development tools (e.g. development environment, management, source code access / revision control)  K6 best practice design techniques |
| B.3.Testing | K1 techniques, infrastructure and tools to be used in the testing process  K2 the lifecycle of a testing process  K3 the different sorts of tests (functional, integration, performance, usability, stress etc.)  K4 national and international standards defining quality criteria for testing  K5 web, cloud and mobile technologies and environmental requirements |
| B.5. Documentation Production | K1 tools for production, editing and distribution of professional documents  K2 tools for multimedia presentation creation  K3 different technical documents required for designing, developing and deploying products, applications and services  K4 version control of documentation production |
| C.4. Problem Management | K1 the organisation’s overall ICT infrastructure and key components  K2 the organisation’s reporting procedures  K3 the organisation’s critical situation escalation procedures  K4 the application and availability of diagnostic tools  K5 the link between system infrastructure elements and impact of failure on related business processes. |

*Table 4.4: Skills needed related to different e-competences*

|  |  |
| --- | --- |
|  | **JUNIOR JAVA DEVELOPER & JAVA DEVELOPER** |
| **e-competences** | **SKILLS: is able to** |
| B.1. Application Development | S1 explain and communicate the design/development to the customer  S2 perform and evaluate test results against product specifications  S3 apply appropriate software and/or hardware architectures  S4 develop user interfaces, business software components and embedded software components  S5 manage and guarantee high levels of cohesion and quality  S6 use data models  S7 perform and evaluate test in the customer or target environment  S8 cooperate with development team and with application designers |
| B.2. Component Integration | S1 measure system performance before, during and after system integration  S2 document and record activities, problems and related repair activities  S3 match customers’ needs with existing products  S4 verify that integrated systems capabilities and efficiency match specifications  S5 secure/back-up data to ensure integrity during system integration |
| B.3.Testing | S1 create and manage a test plan  S2 manage and evaluate the test process  S3 design tests of ICT systems  S4 prepare and conduct tests of ICT systems  S5 report and document tests and results |
| B.5. Documentation Production | S1 observe and deploy effective use of corporate standards for publications  S2 prepare templates for shared publications  S3 organise and control content management workflow  S4 keep publications aligned to the solution during the entire lifecycle |
| C.4. Problem Management | S1 monitor progress of issues throughout lifecycle and communicate effectively  S2 identify potential critical component failures and take action to mitigate effects of failure  S3 conduct risk management audits and act to minimise exposures  S4 allocate appropriate resources to maintenance activities, balancing cost and risk  S5 communicate at all levels to ensure appropriate resources are deployed internally or externally to minimise outages |

In order to distinguish two 3 job profiles, used her for illustration, **Junior Java Developer** and **Java Developer**, we cite description elements of EQF Levels and e-CF Levels in Table 3.5 using the relevant part of the table given in e-CF(2014). It shows EQF level and e-CF levels and their descriptions in case of:

* **Senior Professional/Manager**, corresponding to our *Java Developer* profile
* **Professional**, corresponding to our *Junior Java Developer* job profile.

*Table 4.5: European e-CF and EQF level table*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| EQF Level | EQF Levels descriptions | e-CF Levels | e-CF Levels descriptions | Typical Tasks | Complexity | Autonomy | Behaviour |
| 6 | Advanced knowledge of a field of work or study, involving a critical understanding of theories and principles, advanced skills, demonstrating mastery and innovation in solving complex and unpredictable problems in a specialised field of work or study, management of complex technical or professional activities or projects, taking responsibility for decision-making in unpredictable work or study contexts, for continuing personal and group professional development. | e-3 | **Senior Professional/Manager**  Respected for innovative methods and use of initiative in specific technical or business areas; providing leadership and taking responsibility for team performances and development in unpredictable environments. | Consulting | Structured – unpredictable | Works independently to resolve interactive problems and addresses complex issues. Has a positive effect on team performance. | Planning, making decisions, supervising, building teams, forming people, reviewing performances, finding creative solutions by application of specific technical or business knowledge / skills. |
| 5 | Comprehensive, specialised, factual and theoretical knowledge within a field of work or study and an awareness of the boundaries of that knowledge, expertise in a comprehensive range of cognitive and practical skills in developing creative solutions to abstract problems, management and supervision in contexts where there is unpredictable change, reviewing and developing performance of self and others. | e-2 | **Professional**  Operates with capability and independence in specified boundaries and may supervise others in this environment; conceptual and abstract model building using creative thinking; uses theoretical knowledge and practical skills to solve complex problems within a predictable and sometimes unpredictable context. | Concepts / Basic principles | Structured – unpredictable | Works under general guidance in an environment where unpredictable change occurs. Independently resolves interactive issues which arise from project activities. | Designing, managing, surveying, monitoring, evaluating, improving, finding non standard solutions. |

Unfortunately, the ICT Foundation Body of Knowledge does not provide yet lower levels of knowledge and it is not sufficient for a curriculum development. Therefore, additional extensions (sub-topics) of the Bodies of Knowledge are needed. We used two BOKs: SWEBoK 3.0 (2014) and Computer Science BOK (2013), shown in Figure 8. BMU is using these two BOKs for its BSc programs: **Software Engineering** and **Information Technology**. These BOKs specify required knowledge not only at levels of knowledge areas, but also at lower levels: knowledge units and topics.

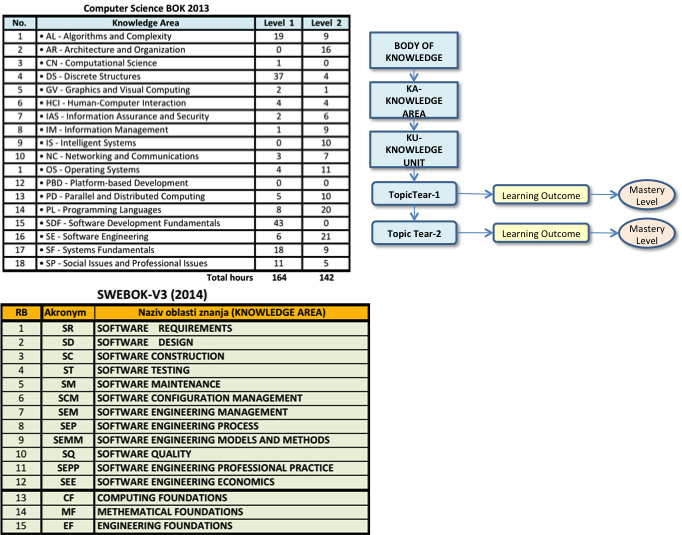


Figure 8 Knowledge areas of SWEBOK 3.0 and Computer Science BOK 2013

Computer Science BOK specifies part of knowledge areas and knowledge units that represent Computing Foundation knowledge area od SWEBOK 3.0.

As shown in Figure, Body of Knowledge (BOK) has structure that specifies several hierarchical levels of knowledge, representing: knowledge areas (KA), knowledge units (KU) and topics (T). A BOK can also specify required learning outcomes. As such, a BOK specifies the knowledge and learning outcomes that a curriculum should provide to its graduates. As it does not provides a very detailed knowledge topics, a developer of a curriculum, and especially, course developers need to specify topics in a more details then specified in a BOK.

# Specification of the short-cycle program for a job profile

## Organizational structure of a short cycle program

In order to develop the required competences of a ICT Profile, such as Developer, a learner must learn all knowledge units (such as topics and sub-topics of a Knowledge Area) specified for the Profile and develop necessary skills. A **course** provides a basic set of knowledge and skills that a student must verify that he or she acquired by passing an exam. To acquire all competences required, a student must complete a number of courses by passing their exams. The granularity of courses my be different and smaller courses are usually preferable, as student can easier complete their assignment specified by their syllabi and pass their exams.

Figure 9 shows a typical organization of a SCHE program, consisting a set of courses, and each course consists with a number of lessons. At BMU, each lesson has a number of learning objects of fine granularity. These are foundation elements that can be easily combined to create different online lessons, of different online courses. Traditional, in-class or Face-to-Face (F2F) courses ate BMU also use online learning materials created witj online learning objects (LOs). Learners of SCHE programs get a Program Certificate if the pass all exams, and Courses Certificates, when they pass the exam of a course.

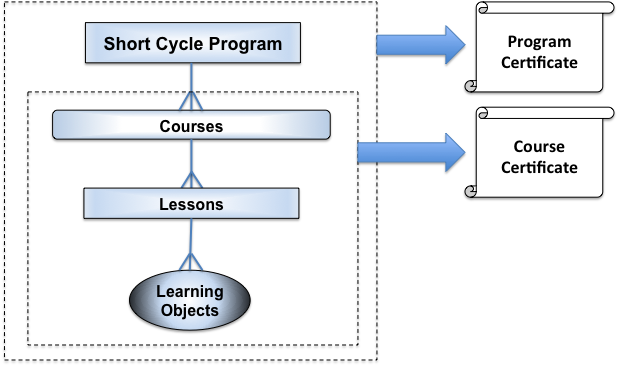


Figure 9 A typical organization of SCHE program

A SCHE program may also have modules. Modules consists of courses that all together, provide a specific set of e-competences. For instance, a SCHE program for a job profile may have two moduls. The first one provides e-competences with lower proficiency level, a the second one – with higher proficiency level. It allows students (learners) to choose a needed le proficiency level and to get an appropriate Module Certificate, as shown in Figure 10. In case of SCHE program for Java Developers, the SCHE program may have two modules:

1. Basic Java module
2. Advanced Java module

Each of them has several courses.

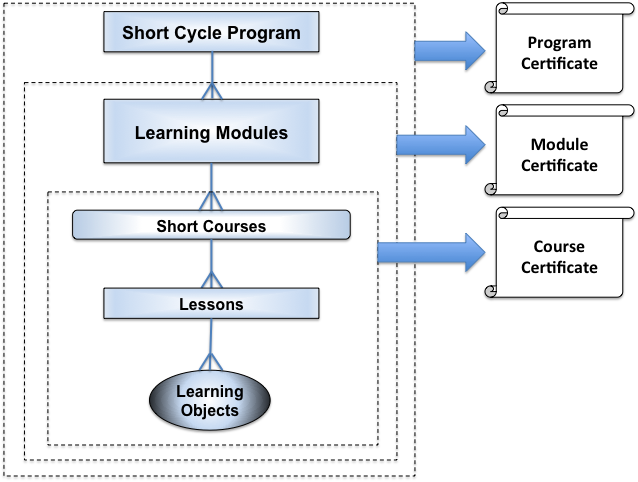


Figure 10 A SCHE program with modules

## Acquiring e-competence

When designing SCHE program, it is of vital importance to specify an appropriate set of e-competences. They can be specified as it is explained in section 3.

## Mapping of BOM into e-competences

Mapping of BOM into e-competences aims to collect knowledge areas, knowledge units and topics needed for an e-competence at a specific e-CF proficiency level. As knowledge units and topics are related to learning objects (LOs), a SCHE developer, together with it SCHE developers, has to search and select needed LO from BMU LO Repository. It consists of all LOS developed for all its courses. If it is necessary, some LO must be modified or developed, as new LOs, if the existing ones are not appropriate.

As the result, it is necessary to create a table showing all topics and LOs needed fro each for each required e-competence.

# Courses of the short-cycle program

## List of courses and their sequence

A SCHE program may be implemented in three stages:

1. **Stage 1: Preparatory Stage** - aiming to prepare trainees for the SCHE program, providing some basic, prerequisite knowledge, or fundaments. Probably a majority of students may have an appropriate fundamental knowledge specified as prerequisite for the SCHE program, but for those that have not this level of knowledge, the Preparatory Stage provides courses they need to have before are ready for courses planned for the SCHE program.
2. **Learning stage** – provides knowledge, skills and competences specified in Section 4 required for the target job profile, distributed to a set of courses. Each course represents a pedagogically complete component of a SCHE program.
3. **On-the-job training stage** – providing students internship, lasting typically one month. If a SCHE program has a course that is project-oriented, without teaching, where students implement learnt knowledge solving a practical problem, solve with a group project, thwn this stage can include also this, “project-oriented” course

All courses in computing are specified as follow: **KIxxx <name of the course>**, for instanc: *KI104 Java2: Object-Oriented Programming.* In case of SCHE Programming in Java,the courses is presented in Figure 11. The program provides 600 hours of active teaching and 60 ESPB, and its duration is 12 months: 9 months of learning, one month for summer holidays, and 2 months for internship. Learning is performed in blocks, i.e. according to the "course-by-course" system. All courses are released using blended learning: 504 online hours (84%), 72 workshops F2F hours (12%) in computer rooms, and 24 hours (4%) of online consultations with the lecturer. A student spends on each course, at least three hours a day. Including Saturdays (or 18 hours per week) using the e-Learning System of BMU. A workshop has 3x45 minutes=135 minutes = 2,25 F2F hours.

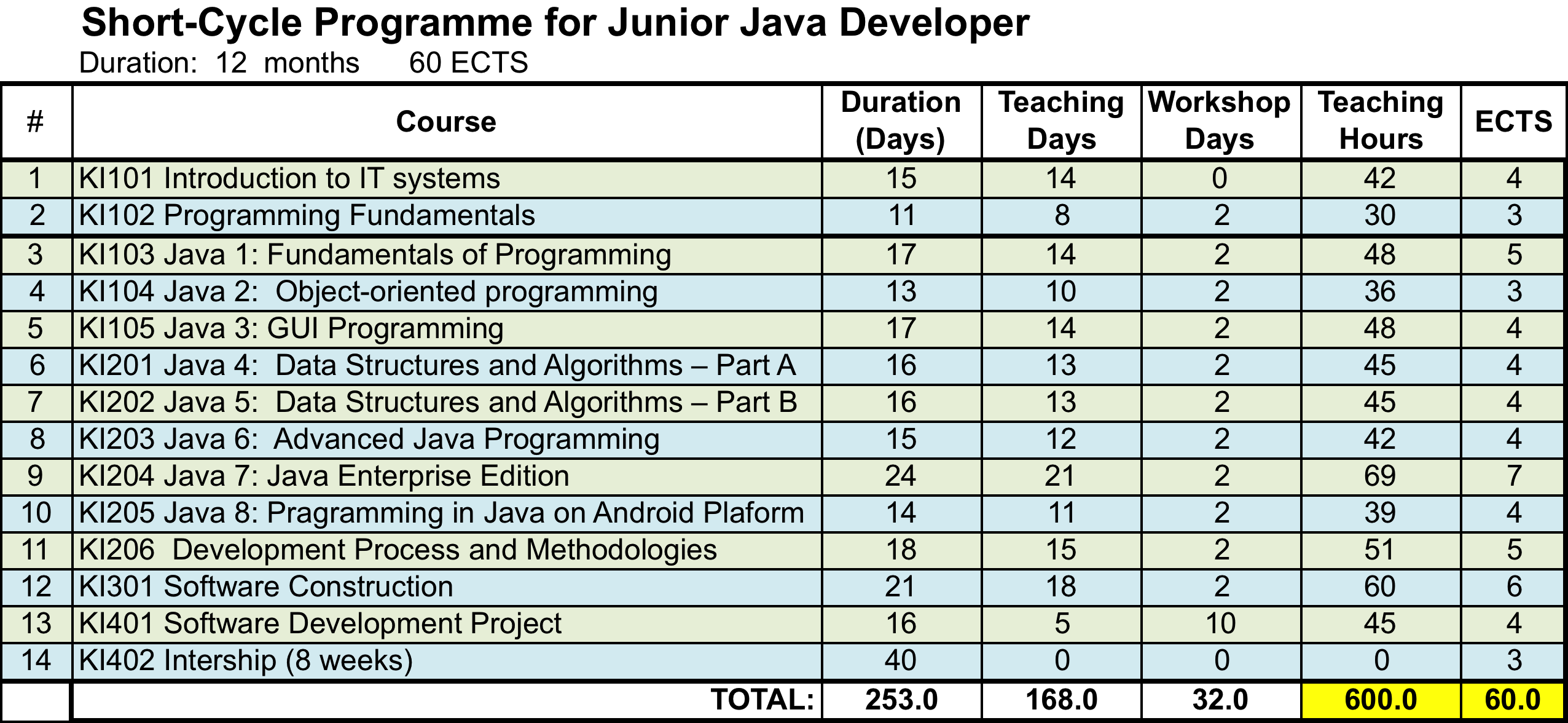


Figure 11 Courses of the SCHE Programming in Java

## Syllabi of courses

For each course a syllabus must be specified, providing all needed information about the course to students. As an example, we provide here the syllabus for KI104 Object-Oriented Programming.

**Syllabus**

**KI104 OBJECT-ORIENTED PROGRAMMING**

**2018/19**

|  |  |
| --- | --- |
| **LECURERS** | |
| Author of the learning material and course designer |  |
| Lecturers |  |
| e-mail addresses of lecturers |  |
| Skype addresses of lecturers |  |
| Skype consultation hours of lecturers |  |
| **Course Data** | |
| Number of ECTS | 3 |
| Number of online learning hours (estimation) | 30 hours |
| Number of workshops and number of F2F hours | 2 workshops, with total 6x45 miinutes = 4,5 hours |
| Expected additional individual learning hours | 30 hours |
| **Assignments** | |
| Number of homework assignmnets | 10 |
| Maximum number of points per homework assigment | 3 |
| Number of tests | 10 |
| Maximum number of points per test | 2 |
| Maximum number of points for the exam | 50 |
| **Maximum number of points** | **100** |
| Duration of the exam | 3 hours |
| Examination format | Theoretical questions and practical problems |
| Tools used during the exam | NetBeans, PowerDesigner, Java JDK |

**References**:

1. Online e-learning material for KI104 Java2: Object-oriented programming, BMU, 2017

**Recommended online tutorials:**

1. <https://docs.oracle.com/javase/tutorial/>
2. <http://www.javatpoint.com/java-tutorial>
3. <https://www.ibm.com/developerworks/learn/java/intro-to-java-course/>

**Course Aims**:

Predmet uvodi studente u osnovne koncepte programiranja iz perspektive objektno-orijentisanog programiranja. Nastavne teme obuhvataju jednostavne tipove podataka, upravljačke strukture, strukture podataka u vidu nizova karaktera i redova, algoritme, kao i upoznavanje sa programskim jezicima. Studenti se upoznaju sa osnovama objektno-orijentisanog programiranja: objektima, klasama, metodima, prenosu parametar, učauravanja, nasleđivanja i polimorfizma. Pored toga, studenti ovladavaju i osnovama sintakse i semantike programskih jezika: promenljive, tipovi, izrazi, dodeljivanje vrednosti, ulazno- izlazne instrukcije, uslovne i iterativne upravljaèke strukture i strukturisane dekompozicija. Izlažu se principi softverskog inženjerstva i razvoja osnovnih veština programiranja u objektno-orijentisanim jezicima.

**Course Description:**

Predmet postavlja osnove za razumevanje i korišćenje objektno – orijentisanih koncepata i principa i korišćenje odgovarajućih razvojnih okruženja. Ispit je bazični i, kao takav, predstavlja polazni osnov za razumevanje naprednih tehnika i koncepata programiranja koji će biti izučavani u predmetima koji slede nakon KI103. Predmet obuhvata širok spektar bazičnih tema kao što je: razumevanje klasa, objekata i nasleđivanja, savladavanje koncepata enkapsulacije i polimorfizma, rad sa programima koji koriste GUI, rad sa grafikom u JAVA programima. Pre toga, predmet se fokusira na opšte koncepte programiranja: tipove, promenljive i konstante, metode, grananja i iteracije.

**Learning outcomes:**

* Studenti rezumeju i koriste OO koncepte i principe: klasu, objekat, nasleđivanje, polimorfizam i enkapsulacija;

**Assesment of Students**:

1. Homework assgnments: 10 x 3 = 30 points maximum
2. Tests: 10 x 2 =20 points maximum
3. Exam: 50 points maximum

A students has to have at leat 50% of points for each of above three categori

**Teaching Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Day** | **Ho-urs** | **Teaching units** | **Topics** | **Objectives – knowledge or skills that the student should receive** |
| **1,2** | **6** | **Classes and objects** | Defining classes for objects  Example: defining classes and creating objects  Constructing objects using constructors  Accessing objects via reference variables  Using classes from the java library  Static variables, constants, and methods  Visibility modifiers  Data field encapsulation  Passing objects to methods  Array of objects  Immutable objects and classes  The scope of variables  The this reference  Programming exercises  Programming assignment | To describe objects and classes, and use classes to model objects  To use UML graphical notation to describe classes and objects  To demonstrate how to define classes and create objects  To create objects using constructors  To access objects via object reference variables  To define a reference variable using a reference type  To access an object’s data and methods using the object member access operator (.)  To define data fields of reference types and assign default values for an object’s data fields  To distinguish between object reference variables and primitive data type variables  To use the Java library classes Date, Random, and Point2D  To distinguish between instance and static variables and methods  To define private data fields with appropriate getter and setter methods  To encapsulate data fields to make classes easy to maintain  To develop methods with object arguments and differentiate between primitive-type arguments and object-type arguments  To store and process objects in arrays  To create immutable objects from immutable classes to protect the contents of objects  To determine the scope of variables in the context of a class  To use the keyword this to refer to the calling object itself |
| **3,4** | **6** | **Object-oriented thinking** | Class abstraction and encapsulation  Thinking in objects  Class relationships  Case study: designing the course class  Case study: designing a class for stacks  Processing primitive data type values as objects  Automatic conversion between primitive types and Wrapper class types  The BigInteger and BigDecimal classes  The String class  The StringBuilder and StringBuffer classes  Programming exercises  Programming assignment | To apply class abstraction to develop software  To explore the differences between the procedural paradigm and object-oriented paradigm  To discover the relationships between classes  To design programs using the object-oriented paradigm  To create objects for primitive values using the wrapper classes (Byte, Short, Integer, Long, Float, Double, Character, and Boolean)  To simplify programming using automatic conversion between primitive types and wrapper class types  To use the BigInteger and BigDecimal classes for computing very large numbers with arbitrary precisions  To use the String class to process immutable strings  To use the StringBuilder and StringBuffer classes to process mutable strings |
| **5,6** | **6** | **Inheritance and Polymorphism** | Superclasses and subclasses,  Superclasses and subclasses methods  Using super keyword  Overriding methods Overriding vs overloading, Polymorphism  Dynamic binding  Casting objects and the instanceof operator.  The Object’s equals method  The ArrayList class  Case study: a custom stack  The protected data and methods  Preventing extending and overriding  Programming exercises  Programming assignment | To define a subclass from a superclass through inheritance  To invoke the superclass’s constructors and methods using the super keyword  To override instance methods in the subclass  To distinguish differences between overriding and overloading  To explore the toString() method in the Object class  To discover polymorphism and dynamic binding  To describe casting and explain why explicit downcasting is necessary  To explore the equals method in the Object class  To store, retrieve, and manipulate objects in an ArrayList  To construct an array list from an array, to sort and shuffle a list, andto obtain max and min element from a list  To implement a Stack class using ArrayList  To enable data and methods in a superclass accessible from subclasses using the protected visibility modifier  To prevent class extending and method overriding using the final |
| 7,8 | 6 | **Exception Handling and Text I/O** | Exception-Handling Overview  Exception types  More on exception handling  The finally clause  When to use exceptions  Rethrowing exceptions  Chained exceptions  Defining custom exception classes  The File class  File input and output  Reading data from the Web  Case study: Web Crawler  Programming exercises  Programming assignment | To get an overview of exceptions and exception handling  To explore the advantages of using exception handling  To distinguish exception types: Error (fatal) vs. Exception (nonfatal)and checked vs. unchecked  To declare exceptions in a method header  To throw exceptions in a method  To write a try-catch block to handle exceptions  To explain how an exception is propagated  To obtain information from an exception object  To develop applications with exception handling  To use the finally clause in a try-catch block  To use exceptions only for unexpected errors  To rethrow exceptions in a catch block  To create chained exceptions  To define custom exception classes  To discover file/directory properties, to delete and rename files/ directories, and to create directories using the File class  To write data to a file using the PrintWriter class  To use try-with-resources to ensure that the resources are closed automatically  To read data from a file using the Scanner class  To understand how data is read using a Scanner  To develop a program that replaces text in a file  To read data from the Web  To develop a Web Crawler |
| **9**  **10** | **6** | **Abstract Classes and Interfaces** | Abstract classes  Case study: the AbstractNumber Class  Case study: Calendar and GregorianCalendar  Interfaces  The Comparable interface  The Cloneable interface  Interfaces vs. abstract classes  Case Study: the Rational class  Class design guidelines  Programming exercises  Programming assignment | To design and use abstract classes  To generalize numeric wrapper classes, BigInteger, and BigDecimal using the abstract Number class  To process a calendar using the Calendar and GregorianCalendar classes  To specify common behavior for objects using interfaces  To define interfaces and define classes that implement interfaces  To define a natural order using the Comparable interface  To make objects cloneable using the Cloneable interface  To explore the similarities and differences among concrete classes, abstract classes, and interfaces  To design the Rational class for processing rational numbers  To design classes that follow the class-design guidelines |
| **11** | **3** | **F2F Project Workshop**  (in BMU computer rooms, optionally -online) | Distribution of projects assignments  Students work on their project tasks with assistance of instructors | To learn how to specify a project  To learn how to organize the project and to break-down tasks  To implement acquired knowledge during the course |
| **12** | **3** | **F2F Project Workshop**  (in BMU computer rooms, optionally -online) | Students work on their project tasks with assistance of instructors | To develop necessary Java programs  To realize all programming tasks of students’ project.  Presentation of the project report |
| **13** | **3** | **Final examination**  (in BMU computer rooms) | Students get examination questions and problems  Exam duration - 3 hours | To evaluate knowledge and skills acquired during the course |

# Pedagogical approach for implementation of curriculum of a SCHE program

Teaching of SCHE programs is not the same as teaching Bachelor of Science programs. Heterogeneity of students is evident, taking into consideration of their ICT knowledge level, age, employment, motivation etc. The SCHE programs may be delivered with courses in the following modes:

1. F2F courses
2. Online courses
3. Blended learning courses

For SCHE Programming i Java, BMU implemented option 3 - blended learning workshops, as shown in Figure 12.



Figure 12 Three components of a course in SCHE Programming in Java

Instead of academic organization of courses (4-5 courses per semester realized in parallel during 15 weeks), it is expected that a SCHE program may be more effective if courses are sequentially thought, as shown in Figure 13. Exams should demonstrated students’ ability to implement what they learnt. If they fail, they will have one additional exam.



Figure 13: Sequential implementation of courses of SCHE Programming in Java

Figure 14 shows the organization of an online lesson. It consists of a number of topics and sub-topics. A topic or sub-topic consists of one or more sections that contain contents in form of multimedia web pages created by mDita Editor developed by BMU.

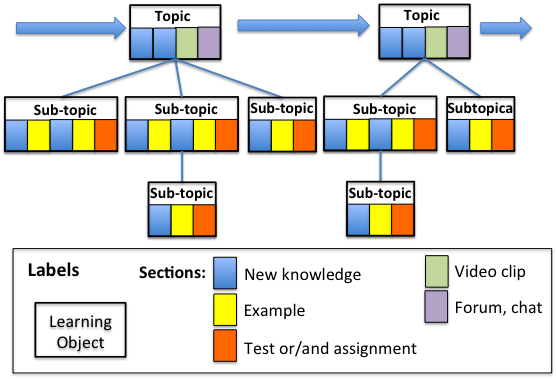


Figure 14 Organization of an online lesson with learning objects, related to topics and sub-topics using sections of different kinds

Figure 15 shows the structure of an online lesson, consisting of learning units and topics. Students learn each topic by using many small sub-steps: Learning-Example-Task. Mapping of learning units into learning objects (LO) and topics into sections, we get an online lesson. An online lessons contains a number of learning objects with one or more sections. Sections may provide now knowledge concepts, examples, assignments, tests, video clips, forums or chats. First order learning objects (or LO) contains topic sections or/and sub/topic sections. Each section is multimedia web page that contains textual information, video and audio clips, listings of Java codes and evaluation sections, such as different kind of tests and assignments. Authors of courses organize online lessons as hierarchy of learning objects related to topics and sub-topics. Online lessons, topics and subtopics are specified according to knowledge units and topics defined in BOM (the Body of Knowledge) of the SCHE Programming in Java. Hours on online lessons are rough estimation of durations of online lessons, but the focus is on lessons’ content, not in their durations.

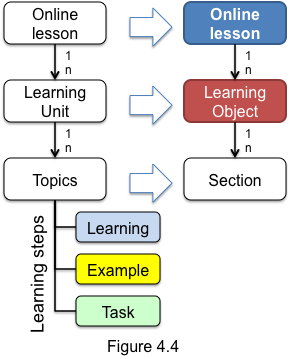


Figure 15

Figure 16 specifies a learning unit and its steps. Authors decide the granularity of their learning objects. A learning units could be implemented with one or more learning objects. Learning sub-steps are usually implemented with sections (web pages).

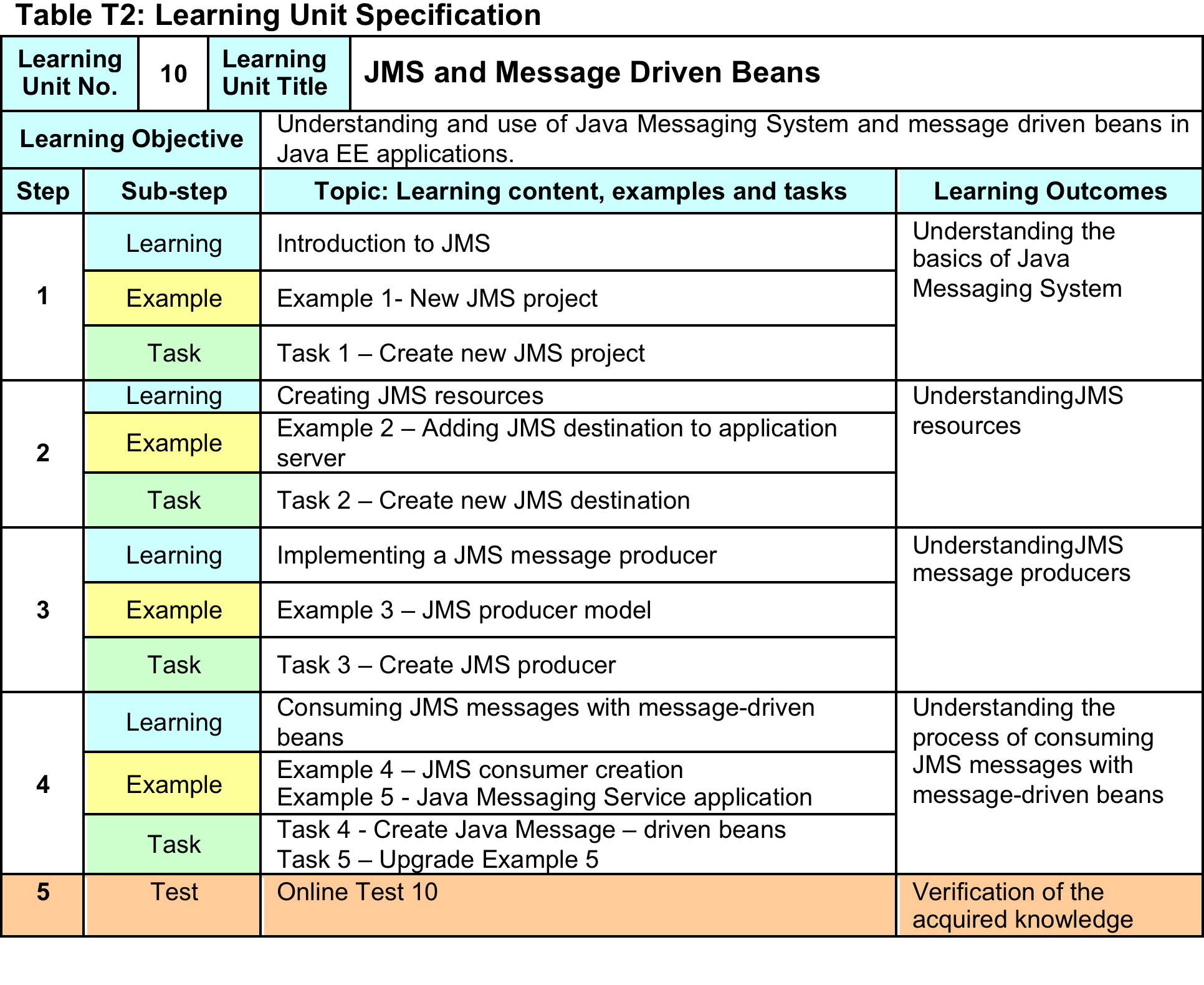


Figure 16 An example of specification of a learning unit

A teaching plan of each course should be created. For each learning unit, a table like the one shown in Figure 16 should be specified.

# Implementation plan of a short-cycle program

After the specification of the curriculum of a SCHE program and syllabi of its courses, as described in previous sections, it is necessary to specify the implementation plan of a SCHE program. It depends of the mode of delivery of a SCHE program (F2F, online or blended learning). In case of SCHE program “Programming with Java” of BMU, a blended learning mode was implemented. Figure 17 shows its implementation plan specifying time periods planned for each course that is taught in sequential order.

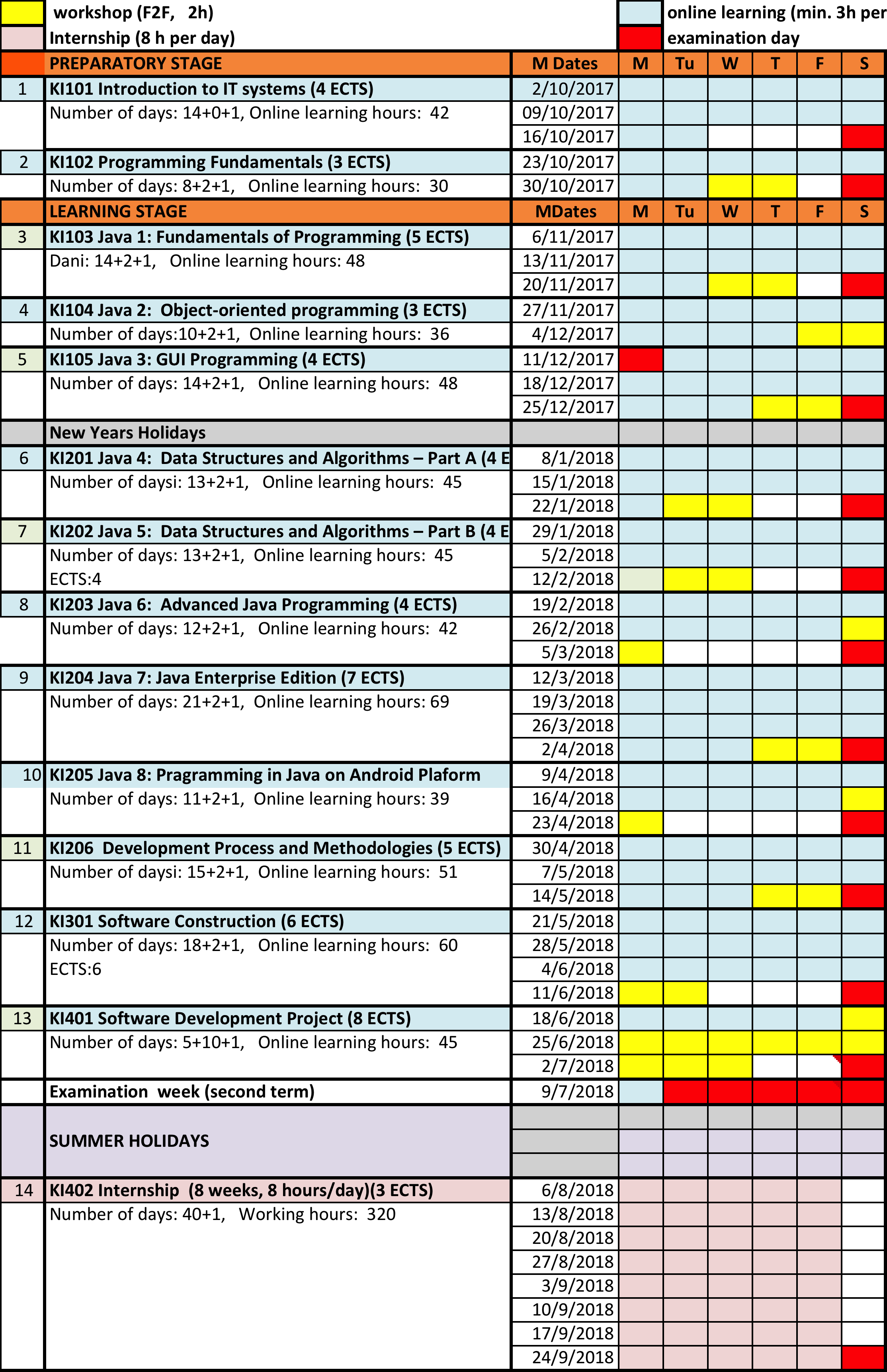


Figure 17: Implementation plan of SCHE program “Programming in Java” of BMU

# The procedure of development and acceptance of a SCHE program

In order to prepare a successful and quality SCHE program, it is necessary to specify a process of development and acceptance of a SCHE program, that includes all key stakeholders. Figure 18 the process implemented at BMU.

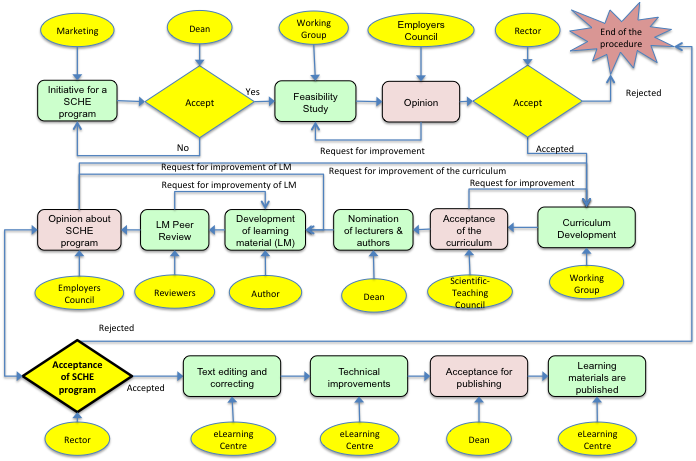


Figure 18: The process of development and acceptance of a SCHE program at BMU

Implementation of such processes, a part of QA, minimizes the risk of failures with selection of organization of SCHE programs.

# The procedure of implementation and analysis of SCHE programs

Once a HEI decides to implement a curriculum of SCHE program, its implementation pan needs to be developed and monitored. Based on its analysis, a SCHE program may be:

* accepted, without any change for next implementation stage;
* modified and improved before the net implementation stage;
* aborted, as it was not successful.

The analysis should include many factors and success indicators, such as quality assessments by employers, but also including financial ones. Figure 19 shows the procedure of implementation and analysis of SCHE programs at BMU.

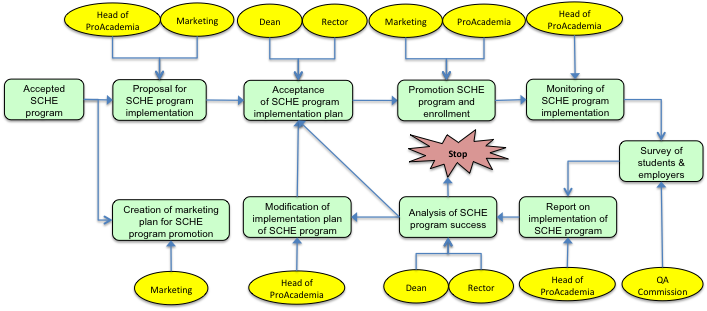


Figure 19: The procedure of implementation and analysis of SCHE programs at BMU

# Quality assurance

In this section it is necessary to specify some organizational measures that may also affect the quality of SCHE program.

## Instructions to lecturers

SCHE programs should provide appropriate learning materials to their students. The have to be different from learning materials used for bachelor studies, as profile of students are different. Learning materials for SCHE programs must have more examples and exercises and less theoretical and historical elements then learning material for bachelor studies. Also, they have to implement the “step-by-step” learning approach described in Section 7.

## Monitoring

Monitoring of implementation of SCHE programs is necessary in order to verify that the all activities described in this document are implemented as required. Special attention should be given to

* **curriculum and syllabi** – appropriate for the target job profile;
* **learning materials** - implement “step-by-step” learning approach and implementation of internal review of developed learning material;
* **learning process** – analysis of synchronous and asynchronous consultations with students, and especially, realization of planned workshops in computer rooms of BMU;
* **realization of homework assignments and tests** - achieved results of students, timing submission of students reports;
* **realization the final course with teamwork–based projects** - results achieved, time of project reports submission;
* **exams** - exam questions and tasks, achieved results of students, duration of exams, timing information (when students passed each exam);
* **internships** – whether selected companies provided adequate jobs to students, whether students submitted required reports, obtained opinions of companies about knowledge and skills of students

Head of ProAcademia is responsible for all these monitoring activities. If necessary, he/she can assign some tasks, when a specific technical knowledge is needed.

## Reports

Results of monitoring activities should be reported to appropriate key BMU managers (rector, dean, general secretary, marketing director, QA director). The head of ProAcademia is responsible to prepare and distribute these reports for each SCHE program, such as:

1. Enrolment report for each group of students;
2. Report on realization of the procedure of preparation and acceptance of new SCHE programs, according to Section 9 (Figure 18);
3. Report on implementation and its analysis, according to the process described in Section 10 (Figure 19);
4. Report on peer review of learning material;
5. Report on success of students on exams and their assignments;
6. Report on realized internships.

## Corrective actions

Head of ProAcademia and key managers of BMU must initiate and perform corrective actions if some of activities specified in this documents are not performing as expected. Corrective actions should be implemented ASAP after monitoring of realization of activities and their analysis. Results of implemented corrective actions should be reported by the head of ProAcademia, and analyzed by BMU management. Based on implemented corrective actions, it is possible to prepare modification of a SCHE program for future groups if students.

## Students’ surveys

Students’ surveys should be realized in different stages of a SCHE program:

* after realization of each course;
* after realization of all courses of a SCHE program and internships of students

Analysis of the survey should identify good and bad elements of a SCHE program and should suggest corrective actions, if they are needed.

## Employers’ surveys

Two surveys of employers should be organized:

1. Survey after internships, and
2. survey after employment of students.

Each of these surveys should be analyzed separately, and reported independently. On order to realize the second survey, it is necessary that ProAcademia has information on employment of students of each SCHE program.

The analysis of the surveys aim to identify the weak and strong aspects of a SCHE program and to recommend possible improvements.

# Conclusions

This documents is one of QA documents as it specify activities that can affect the quality of a SCHE program. By “quality of a SCHE program” we mean achievement of the following goals:

* All students with the certificate of a SCHE program got an appropriate job, i.e. are working according to the target job profile specified for the SCHE program.
* Employers are satisfied with students that completed the SCHE program and got its certificate.
* Most of students, if not all, successfully completed the SCHE program.

# References

1. CEN ICT Skills Workshop „European ICT Professional Profiles in action“, INTERIM REPORT, June 2017
2. CEN, “European e-Competence Framework”, 2014
3. European Commission, DG Internal Market, Industry, Entrepreneurship and SMEs, „The European Foundational ICT Body of Knowledge“, European Union, 2015.
4. The Government of the Republic of Serbia, the Ministry of Education, Science and Technological Development, „Strategy for Education Development in Serbia 2020“, The Ministry of Education, Science and Technological Development of the Republic of Serbia, Belgrade, 2012.
5. The Government of the Republic of Serbia, „Low for Higher Education in Serbia“, Službeni glasnik RS, No. 88/17, Belgrade, 29 September 2017.
6. Association for Computing Machinery (ACM) and IEEE Computer Society (IEEE-CS), „Information Technology Curricula 2017 - IT2017 Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology“, 2017.
7. P. Bourque and R.E. Fairley, eds., ”Guide to the Software Engineering Body of Knowledge“, Version 3.0, IEEE Computer Society, 2014
8. Association for Computing Machinery (ACM) and IEEE Computer Society (IEEE-CS), „EITBOK - Enterprise Information Technology Body of Knowledge“, 2017.