7:e Utvecklingskonferensen för Sveriges ingenjörsutbildningar, Luleå tekniska universitet,
27 november – 28 november 2019

Methods for Enhancement of a Master of Engineering Programme

H. Forsberg, K. Lundqvist, and K. Forsberg

***Abstract*—This paper describes methods we used to improve our Master of Engineering programme in Dependable Aerospace Systems together with the industry. The target audience is mainly programme coordinators/managers who are in the process to develop their programmes for future demands. The two main questions we address are: Q1 – How do we ensure a good progression within a programme to ensure the industry's current and future needs in engineering skills? and Q2 – How do we ensure students become acquainted with research during their studies? The results indicate that our suggested method to analyse programme progression through subject abilities supports developer of engineering programmes and that our approach to undergraduate research opportunities is a way forward to introduce students to research early.**

***Index Terms*—Engineering education, engineering methods, undergraduate research opportunities, unified engineering**

I. Introduction

When developing (enhancing or changing) a new Master of Engineering programme, i.e. a "civilingenjörsprogram", it is mandatory to show evidence of how students will fulfil 16 abilities after completing their studies. These abilities are set up by the Swedish Higher Education Authority (UKÄ). They are divided into three areas; knowledge and comprehension, skills and capabilities, and judgement and attitude. Making sure the students fulfil these 16 abilities is a very good reference for the industry. But when it comes to subjects specific for each programme, abilities are typically not defined. Instead, the degree objectives usually require a certain amount of credits in basic and advanced courses within the main subjects. We address question Q1 with a suggested method based on subject *abilities* and *skills* within a programme. We introduced this idea in [1] but never formalised it. The method assumes that all courses have been or will be developed based on a structure, where each course's learning outcomes are explicit and can be analysed through the Structure of the Observed Learning Outcome (SOLO) taxonomy [2]. The SOLO taxonomy classifies the learning outcomes in terms of verb complexity, which enables analysis of the difficultness of work the students have to perform to fulfil the learning outcomes in a specific course. The SOLO taxonomy divides the verbs in five different complexity levels. See Table 1. The lower complexity levels (2-3) are associated with surface learning while the highest levels (4-5) are associated with deep learning. Level 1 is omitted for higher education. The SOLO taxonomy requires explicit learning objectives.

Table 1. Structure of the Observed Learning Outcome (SOLO), levels and examples of their associated verbs. Source [3]

Quantitative (surface learning)		Qualitative (deep learning)	
SOLO2	SOLO3	SOLO4	SOLO5
Uni-structural	Multi-structural	Relational	Extended abstract
Paraphrase	Combine	Analyse	Theorize
Define	Classify	Compare	Generalize
Identify	Structure	Contrast	Hypothesize
Count	Describe	Integrate	Predict
Name	Enumerate	Relate	Judge
...	...	...	...

← Hierarchy of verbs →

To allow students to become acquainted with research during their studies (question Q2 above), we developed our own variant of undergraduate research opportunities (UROP) and named it SFP (studentforskningsprogrammet). SFP was a direct response to the industry's requests for research driven engineers but also to make sure we have suitable future PhD candidates. Some ideas were derived from Merkel [4].

The rest of the paper is organized as follows: In section II we introduce Methods for Enhancement of a Programme and in Section III we discuss our implementation of Undergraduate Research Opportunities. In Section IV we reveal results and conclude the paper.

II. Methods for Enhancement of a Programme

This section covers four different methods *A*. to *D*. that we used when we developed new advanced courses and enhanced the progression within our engineering programme.

A. Balancing Course In-Depth Levels

It might be a trivial task to do but important. We recommend doing an in-depth level analysis as the first step before any program progression analysis is performed. A well-balanced program should have courses in each main subject from first cycle with only upper-secondary level entry requirements (G1N) all the way up to second cycle with second-cycle courses as entry requirements (A1F) (except for the subject containing the master thesis which should go even deeper (A2E)). See Table 2 for our programme's course in-depth levels for each subject. The three formal main subjects are aeronautical engineering, computer science and electronics.

Table 2. Course in-depth levels for each subject. HP = credits

Subject	G1N	G1F	G2F	A1N	A1F	A2E
Aeronautical Engineering	20HP	15HP	15HP	5HP	30HP	30HP
Computer Science	12.5HP	15HP	-	30HP	22.5HP	
Electronics	7.5HP	17.5HP	7.5HP	7.5HP	7.5HP	
Mathematics	15HP	15HP				
Physics		15HP				
Product and Process Development			10HP			
Other Subjects	2.5HP					
Sum	57.5HP	77.5HP	32.5HP	42.5HP	60HP	30HP
Dependability	17.5HP	7.5HP	25HP	12.5HP	52.5HP	30HP
Entry Level	167.5HP of which is 50HP dependability			---	---	---
Advanced Level	---	---	---	132.5HP of which is 95HP dependability		
Total Dependability	145HP					

For these three subjects, the courses are evenly spread over all in-depth levels except for a hole in G2F for computer science. This hole is informally filled by one of the aeronautical engineering courses at the same level which includes a large portion of software engineering. Note that Mälardalen University does not have a formal subject in dependability or systems, even if the students get a degree in *Dependable Systems.* Table 2 also shows that the subject dependability covers all in-depth levels with emphasize on the advanced levels.

B. *Analysing progression using SOLO average values*

Inspired by Brabrand and Dahl [5] and their introduction of SOLO average for analysis of competence progression, we did some analysis of our programme in a similar manner.

By checking each verb's associated SOLO level in a course syllabus, it is possible to measure the SOLO average level for that course. To give an example. One of our year one courses, Programming (DVA117), contains the following verbs and associated SOLO levels; *solve* (2), *use* (2), *perform* (2) and *explain* (4). The SOLO average is then: (2+2+2+4) / 4 = 2.5. By averaging all courses for each year, a yearly SOLO average can be achieved. These values should have a positive trend. Table 3 shows our programme's yearly SOLO average values. These values are indeed increasing except for the last year. The reason for the last year's lower value compared to year 4 is that the syllabus for the master thesis course contains some SOLO level 2 and 3 verbs together with the fact that there are only three courses in year 5 compared to eight in year 4.

Table 3. SOLO average values for each year in the Master of Engineering Programme in Dependable Systems

Year	1	2	3	4	5
SOLO Average	3.17	3.25	3.48	3.86	3.69

It is also possible to study different subjects and their progression in SOLO average values for courses and respective in-depth levels. To give an example. The subject Electronics at Mälardalen University starts with a basic course teaching discrete components. The students shall typically *apply, describe, analyse* and *reason* about the electronics. The same verbs apply for more advanced courses. The difference is instead in the complexity of the electronics the students learn about. The more advanced courses the more complex electronics. Table 4 shows the SOLO average values for electronics courses in our programme. The variations between each in-depth level is relatively small.

Table 4. SOLO average values for respective in-depth level for the subject electronics

Electronics In-Depth Level	G1N	G1F	G2F	A1N	A1F
SOLO Average	3.47	3.45	3.5	3.33	3.57

For dependability, however, the variations in SOLO average levels increase steeply from first to second cycle courses. See Table 5. The only dependability course to be found in level G1F is a new course, *Requirements Engineering*, which is not fully developed yet, thus the *X* in SOLO average level. The expected value is however close to 3.4.

Table 5. SOLO average values for respective in-depth level for dependability

Dependability In-Depth Level	G1N	G1F	G2F	A1N	A1F
SOLO Average	3.32	*X*	3.40	3.93	3.92

Our conclusion regarding the use of SOLO average analysis is that it is not so useful when analysing progression in engineering programmes.

C. *Ensuring Programme Progression Through Subject Abilities*

Our next analysis approach for ensuring programme progression is based on evidencing abilities for a specific subject. The idea is inspired by the way Master of Engineering programmes (civilingenjörsprogram) are developed in Sweden. When developing, enhancing or changing a "civilingenjörsprogram", it is mandatory to show evidence of how students will fulfil 16 abilities after completing their studies. It doesn't matter if the students are from industrial economics or robotics, the same principles apply. By using the same evidencing principle but for specific topics instead of a whole programme, we suggest using the following method:

1) *Suggested Method*

Our method works as follows:

a) Create a matrix, with abilities (skills) that a student should have within a specific subject as rows, see Table 6.
b) Arrange the abilities from lower to higher complexity (1 to *m*).
c) Identify all courses from first to second levels that have learning outcomes helping the students to achieve the abilities within a subject. Add these courses as columns in the same matrix (1 to *n*).
d) Identify the SOLO levels for each relevant learning outcome and colour the item in the table with corresponding colours. See Table 6.

Table 6. Required abilities and courses enabling those abilities for a subject within a programme. The colours in the matrix corresponds to each course's relevant learning outcome's SOLO level

Ability – Course ⇒ ⇓	Course 1	Course 2	…	Course *n*
Ability 1	SOLO2	SOLO3		
…		SOLO3	SOLO4	
Ability *m*				SOLO5

For a well-balanced programme, each defined ability should be covered by one or several courses. Basic abilities should preferably be covered by basic courses and complex abilities should be covered by advanced courses with higher SOLO-levels, i.e. the matrix should be coloured from green (top-left) to red (bottom-right). (SOLO level 1 is assumed not relevant for higher educations.) If several courses cover the same ability at the same SOLO level, these courses should be studied deeper, for removal of eventual overlaps. Once this matrix is developed the following process should be performed when developing a new course (replacing an old) or changing a course within the programme:

i. The course developer matches the new learning outcomes with the abilities identified for the subject.
ii. If the course's learning outcomes (or content) cannot match some identified abilities that the course is supposed to cover (gaps in the matrix), the learning outcomes have to be changed or changes in other courses have to be suggested.

We applied the suggested method to our newly developed programme after year two when some courses on advanced level still had to be developed. In the section below we show how we evaluated our method.

2) *Method Evaluation*

When we evaluated our method for the main subject dependability for our programme we defined abilities and skills from terminology and methods from several sources, e.g. Laprie's dependability tree [6] and CS-25, certification requirements for large aircraft [7] and then discussed these with our industry partners working with highly dependable systems. The most complex ability is to be able to theorize about shortcomings with current methods and tools for future systems. We identified 13 abilities and 12 courses relevant for achieving those abilities, see Table 7. Notice that this table includes 14 courses. At the time for evaluation we had not developed the courses *Design of fault tolerant systems*, FLA432, and *Introduction to computers and software engineering*, DVA113. While the former was planned from the beginning of the programme the latter was not. When developing FLA432, we noticed a lack of an ability, i.e. to analyse and maintain sufficient degree of dependability for advanced embedded systems, see Table 7. We therefore made sure this ability was covered by at least one learning outcome when we developed the syllabus. In addition to this, we knew from previous teaching in the course *Development of avionics systems*, FLA309, that this course included too much topics such that the students never got sufficient deep learning, even if the verbs all suggests deeper learning. We therefore changed the program to include two additional courses, i.e. *Introduction to computers and software engineering*, DVA113 and *Requirements Engineering*, DVAXXX. While the former is included in table the latter is not simply because it is not developed yet. Once these two courses are implemented, we need to change FLA309 to cover fewer abilities such that the students can focus on the right topics. The matrix below will help with this development. The results clearly indicate the method as a support for programme developers.

D. *Reducing Complex Dependencies Between Courses*

When we started to develop the programme from the beginning, we used two prominent ideas for year one of the programme: 1) *Unified Engineering* (inspired by MIT) and 2) no mathematical courses until year 2. See [1] for additional information and reasons behind these ideas.

Table 7. Identified abilities and skills for the subject dependability and courses covering these abilities and skills. Green colour = SOLO level 2, Yellow colour = SOLO level 3, Orange colour = SOLO level 4 and Red colour = SOLO level 5

Courses → / Abilities/skills ↓	Introduction to Dependable Aerospace Systems - FLA 107	Introduction to Computers and Software Engineering - DVA113	Human Factors - FLA103	Development of avionics systems - FLA309	Total Quality Management and Maintenance Technology - PPU309	Robust Electronics - ELA304	Autonomous Vehicles - FLA433	Safety Critical Systems Engineering - DVA437	Embedded Systems II - DVA482	Programming of Reliable Embedded Systems - DVA452	Design of autonomous systems - DVA472	Design of fault tolerant systems - FLA432	Project Course in Dependable Systems - FLA400	Thesis for the Degree of Master of Science in Engineering - Dependable Systems - FLA500
Fundamental elements - terminology dependability	2				3			3						
Fundamental elements - basic knowledge in different kinds of faults and errors, fault modes, symmetrical and assymetrical faults, and component failures	2		2		3									
Fundamental elements - elementary methods for identification and classification of faults, risks and fault-tree analysis					3			3						
Fundamental elements - basic analysis methods within dependability (processes and methods for design and analysis methods for dependable systems		4		4	3			3						
Understand how the fundamental elements can be applied in different design examples (from examples, understand and explain how different fundamental elements can be used)				4		3		3		4		4	4	3
Reflect upon multiple design solutions for dependability		3		4		4						4		5
Fundamental elements on a higher level - architectures and fault tolerant mechansims for dependable systems. Techniques to create fail-safe and robust designs (environment, safety and availability), related to intended use (HMI, security and environment)		3	2	4		4			4			4		4
Methods for safety evidence (analysis, verification etc.)				4				4			5	5		3
Methods to measure and evaluate safety. Proven reliability (statistics), argumention skills in showing evidence for safety, and verification					3							4	4	4
Methods to ensure life-cycle traceability, from requirements to implementation (process assurance)		4		4	3								4	
Be able to design and verify a system with high dependability requirements						3	3	4	4	4		5		4
Ability to analyse and maintain sufficient degree of dependability (including safety) for advanced embedded systems with high integration of functions (software intense systems, advanced IMA systems)												5		
Ability to theorize about shortcomings in current system safety methods for future systems														5

After two years with students in the programme, we realized that there were so many dependencies[1] between the first mathematical courses and several other courses. A multitude of students were not able to start more than few courses in year 3 if they failed a single course in math in the beginning of year 2. We therefore decided to visualize all dependencies between all courses in the program. The result was a rat's nest. After a complete review of all courses, we changed some specific entry requirements for certain courses and moved around other courses. Among other things, we moved the course *Single Variable Calculus*, MAA149, from year 2 to year 1. Our new dependency diagram is shown in Figure 1. The figure shows fewer and less complicated dependencies than before. Yet, it is possible to see that two courses are critical for the students to complete. Those are *Programming*, DVA113, and *Single Variable Calculus*.

To conclude the work in Chapter II, we suggest using methods *A.*, *C.* and *D.* in specified order when analysing and/or improving an engineering programme.

III. UNDERGRADUATE RESEARCH OPPORTUNITIES

In our granted KKS project (*AVANS - civilingenjörsprogrammet i tillförlitliga flyg- och rymdsystem*), we got funding for developing a closer relationship between the students and ongoing research at the department. It is desirable for the industry to have research driven engineers and it is important for the university to have strong candidates for future PhD student positions. Inspired by MIT's Undergraduate Research Opportunity Program (UROP) and after discussions with our industry partners in the Industry Advisory Group, we created our own version of UROP and called it SFP (Studentforskningsprogrammet).

[1] The dependencies are located in the course syllabuses as specific entry requirements. These requirements can specifically pinpoint other courses as prerequisites but also a certain amount of credits that has to be satisfied before entering a course.

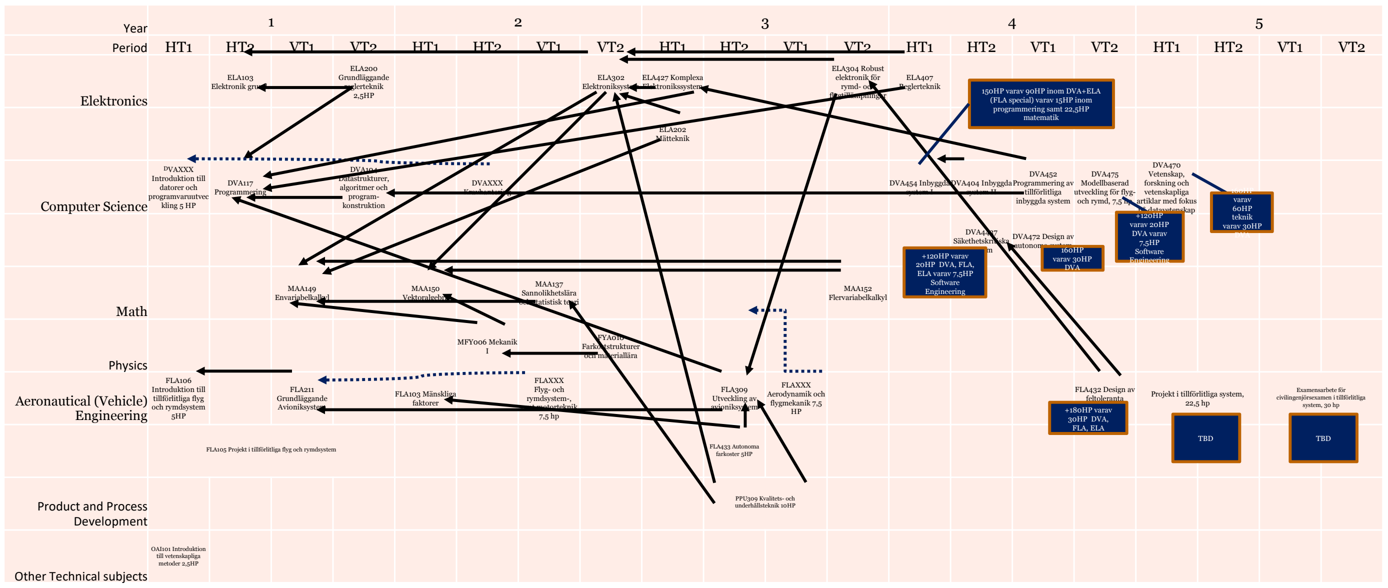
Year
Period
1
2
3
4
5
HT1
HT2
VT1
VT2
Elektronics
Computer Science
Math
Physics
Aeronautical (Vehicle) Engineering
Product and Process Development
Other Technical subjects
ELA103 Elektronik grund
ELA200 Grundläggande reglerteknik 2,5HP
ELA302 Elektroniksystem
ELA427 Komplexa Elektronikssystem
ELA202 Mätteknik
ELA304 Robust elektronik för rymd- och flygtillämpningar
ELA407 Reglerteknik
150HP varav 90HP inom DVA+ELA (FLA special) varav 15HP inom programmering samt 22,5HP matematik
DVAXXX Introduktion till datorer och programvaruutveckling 5 HP
DVA117 Programmering
DVA104 Datastrukturer, algoritmer och program-konstruktion
DVAXXX
DVA454 Inbyggda system I
DVA404 Inbyggda system II
DVA452 Programmering av tillförlitliga inbyggda system
DVA475 Modellbaserad utveckling för flyg- och rymd, 7,5 hp
DVA470 Vetenskap, forskning och vetenskapliga artiklar med fokus på datavetenskap
DVA4437 Säkethetskritiska system
DVA472 Design av autonoma system
+120HP varav 20HP DVA, FLA, ELA varav 7,5HP Software Engineering
160HP varav 30HP DVA
+120HP varav 20HP DVA varav 7,5HP Software Engineering
160HP varav 60HP teknik varav 30HP DVA
MAA149 Envariabelkalkyl
MAA150 Vektoralgebra
MAA137 Sannolikhetslära och statistisk teori
MAA152 Flervariabelkalkyl
MFY006 Mekanik I
FYA010 Farkoststrukturer och materiallära
FLA106 Introduktion till tillförlitliga flyg och rymdsystem 5HP
FLA211 Grundläggande Avioniksystem
FLA103 Mänskliga faktorer
FLAXXX Flyg- och rymdsystem-, och motorteknik 7,5 hp
FLA309 Utveckling av avioniksystem
FLAXXX Aerodynamik och flygmekanik 7,5 HP
FLA432 Design av feltoleranta
+180HP varav 30HP DVA, FLA, ELA
Projekt i tillförlitliga system, 22,5 hp
Examensarbete för civilingenjörsexamen i tillförlitliga system, 30 hp
TBD
TBD
FLA105 Projekt i tillförlitliga flyg och rymdsystem
FLA433 Autonoma farkoster 5HP
PPU309 Kvalitets- och underhållsteknik 10HP
OAI101 Introduktion till vetenskapliga metoder 2,5HP

Figure 1. Intercourse dependencies for the programme

We divided our work in four phases, see Figure 2. In Phase 0, students, researcher and administrative personnel were interviewed. Several requests were taken into account when SFP was formed. The students wanted to understand what research is, work in a real project and not a fictious one, meet companies, and get a certificate to include in their CV in the end of the work. The development of SFP was planned in Phase 0 as well.

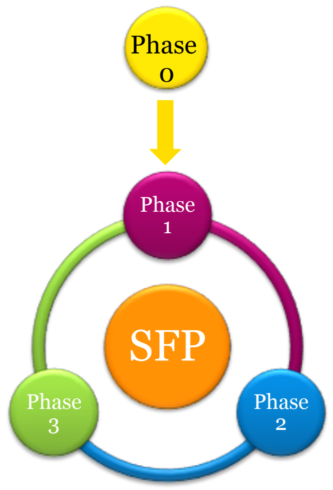

Figure 2.Phases in the SFP, "studentforskningsprogrammet"

In Phase 1, students, research projects and researchers were selected. At first, information was sent out to researchers. Several researchers were interested in having undergraduate students in their projects. Then students were recruited. It was important to consider an approach where the students did not drop behind ordinary courses. Therefore, a maximum limit of 10 SFP-hours work per week was set. In addition, only students with a track record of completing courses were selected for the SFP. Each student that wanted to go through the SFP got a list of interested researchers and their actual projects to select between (seven in our case). They also got the opportunity to visit the researchers, one by one, to get a deeper understanding of what the research project wanted to do. After that, the students applied for and ranked one or several projects. Based on interviews, each researcher then ranked the students who had applied for their project based on technical skills and motivation. Finally, based on both parties ranking, students were paired with researchers. In Phase 2, the students worked with their research tasks. After one month, a follow-up meeting was held. In Phase 3, after completion of the work, another follow-up was held. The developed material for SFP was updated and the students' certificate was handed out. After Phase 3, another new round of SFP could start from Phase 1.

A. UROP Experiences

The first time we implemented SFP, the opportunity was presented for eight candidates in year three of the programme. If a student had taken all courses in the programme at the time we invited them, they should have achieved 165 credits. Only students with more than 157.5 credits were invited. At this time, we had already asked researchers for possible projects. We carefully explained for the researchers the prerequisites including that students may be able to drop whenever they want to. We got plenty of research opportunities for the students. Of the eight students five showed interest and started to work with SFP in parallel with ordinary courses in the 2nd autumn period (HT2). After completion in the middle of spring (between VT1 and VT2) only three students were left. The reasons behind leaving the research projects before the end were never analysed for the two students who left. The students who completed the SFP got a certificate to attach to their curriculum vitae. All three students were very pleased with what they had experienced. After Phase 3, the following documents were deemed reusable for forthcoming years; time plan, documents, certificate template, Power-point presentations for all meetings, and collected opinions and improvements from both students and researchers.

IV. Results and Conclusions

In this paper we went through some methods for enhancing a Master of Engineering programme. We showed the importance of balancing course in-depth levels. We then indicated that the use of SOLO average analysis might not be so useful when analysing engineering programmes (method *B.* in Chapter II). Instead, we developed a method to ensure programme progression through subject abilities. Our results clearly indicate the method to be a support for programme developers. We also presented how to reduce complex dependencies between courses by visualization of intercourse dependencies. Through this procedure it was also possible to see which courses are critical for students to complete early. Finally, we showed our experiences of implementing undergraduate research opportunities in our programme. We conclude that the presented methods *A.*, *C.* and *D.*, in specified order, in Chapter II, provide support for engineering programme developers when new programmes are created, or specific courses will be developed or added to an already existing programme. We also conclude that the way we implemented our undergraduate research programme can help introducing research for students early.

In future work, we will study different methods to retain students throughout a whole programme.

Acknowledgment

This work has been partially supported by funding from the Swedish Knowledge Foundation, within AVANS 2016.

References

[1] Forsberg, H., Lundqvist, K., Ekstrand F., & Otterskog, "Early results and ideas for enhancements of the master of engineering programme in dependable aerospace systems," *The 6th Development Conference for Swedish Engineering (USIU2017)*, Gothenburg, Sweden, Dec 2017.

[2] J.B. Biggs and K.E. Collis, *Evaluating the Quality of Learning: The SOLO Taxonomy* (*Structure of the Observed Learning Outcome*), Academic Press, New York, 1982.

[3] J. B. Biggs, *Teaching for quality learning at university: What the student does*. McGraw-Hill Education (UK), 2011.

[4] C. A. Merkel, "Undergraduate research at six research universities," California Institute of Technology, Pasadena, CA, 2001.

[5] C. Brabrand and B. Dahl, "Using the SOLO taxonomy to analyze competence progression of university science curricula," *Springer*, Higher Education, Vol. 58, Issue 4, Oct. 2009, pp. 531–549.

[6] J-C Laprie, "*Dependability: Basic concepts and terminology*," Dependable Computing and Fault-tolerant Systems, Vol. 5, Springer-Verlag, Vienna, 1992, pp. 3-245.

[7] EASA (European Aviation Safety Agency), Certification specifications for large aeroplanes (CS-25).