Applied Computer Science, vol. 13, no. 3, pp. 32–40
doi: 10.23743/acs-2017-19

Submitted: 2017-07-11
Revised: 2017-09-08
Accepted: 2017-09-10

MDE, DSL, learnability

Kamil ŻYŁA[*]

SIMPLIFIED GRAPHICAL DOMAIN-SPECIFIC LANGUAGES FOR THE MOBILE DOMAIN – PERSPECTIVES OF LEARNABILITY BY NONTECHNICAL USERS

Abstract

Increasing number of technologically advanced mobile devices causes the need for seeking methods of software development that would involve persons without or with highly limited programming skills. They could participate as domain experts or individual creators of personal applications. Methods based on models might be the right answer, thus the author conducted workshops and surveys concerning perspectives of graphical modeling languages for the mobile domain. Research revealed that nontechnical users declared high learnability of simplified ones as well as the majority of them correctly read models in such languages.

1. INTRODUCTION

Development as well as expansion of mobile devices is very dynamic. Not only design and hardware is changing, but also available functionality. For example, mobile phones evolved from simple devices for texting an making voice calls to multithread context-aware microcomputers continuously utilizing Internet connection and processing a lot of multimedia content. Also, other devices like smart watches, TV sets and multimedia centers in cars, started to provide some of their functionality.

Forrester (Taylor, 2015) predicted that by the end of 2016, 4.8 billion people globally will use a mobile phone, and 46% of population will use a smartphone.

[*] Lublin University of Technology, Electrical Engineering and Computer Science Faculty, Institute of Computer Science, 38A Nadbystrzycka St., 20-618 Lublin, +48 81 538 43 49, k.zyla@pollub.pl

According to Gartner (Meulen & Forni, 2016), global sales of smartphones to end users totaled 373 mln units in the 3rd quarter of 2016. Both sources predict growth of mobile services market. These create demand for new methods of mobile software development, that takes into account also nontechnical users. It is gaining importance as the number of such people surrounded by technically advanced devices is still growing. Authors of scientific publications started mentioning about involving nontechnical persons as domain experts or creators of simple personal as well as business applications (Angeles, 2017; Kawasaki, 2016; Viedma, 2010; Kapitsaki, Kateros & Pappas, 2015; Mohamad el al., 2011).

Author assumes in this paper that nontechnical person is a person which is unable (or with significant difficulties) to create software utilizing classical programming languages and methods. Those difficulties might be the result of intellectual deficiencies or lack of proper education.

Another notion that needs explanation is learnability. It could be understood as "...a quality of products and interfaces that allows users to quickly become familiar with them and able to make good use of all their features and capabilities" ("Definition – learnability", 2017). This is coherent with e.g. ISO 9241-11 standard defining learnability as "time of learning" (Abran, Khelifi, Suryn & Seffah, 2003).

Graphical modeling languages are the tools within model-driven engineering (MDE) that could help to involve nontechnical people into software development process. They could be used to create platform-independent models that could be later transformed into running applications by code generation (Brambilla, Cabot & Wimmer, 2012; Kesik & Żyła, 2011). Despite some doubts concerning claimed benefits of using MDE tools (Hutchinson, Rouncefield & Whittle, 2011; Mohagheghi & Dehlen, 2008) and the small number of fully-mature tools, experience gained at the Lublin University of Technology (Żyła & Kesik, 2012; Kesik, Żyła & Nowakowski, 2014; Żyła, 2013, 2015; Rieger, 2017) show that MDE tools might play an important role for people without programming skills.

Modeling languages for the mobile domain, chosen for the purpose of research described in this paper, could be divided into 3 categories. The 1st category concerns languages that are similar to 3G programming languages, although they provide simplified syntax – some irrelevant technical details are hidden. Nevertheless they still operate on the level of instructions and require some programming skills. The 2nd group concerns general purpose graphical modeling languages oriented on defining interactions among objects. They require at least basic object-oriented programming skills, as they operate on the level of objects and invocations of methods. The last category concerns simplified graphical domain-specific languages based on high-level components performing complex actions (activities) and flows among them.

2. AIM OF THE STUDY

The number of mobile devices in common people surroundings is constantly increasing, which makes the need for programming or managing (orchestrating) them by nontechnical persons harder to avoid. The same trend concerns also technical aspects of intelligent houses, smart cities, etc. As the result, the problem of choosing a proper tool (language) suitable for nontechnical users emerges. Thus it is particularly important to recognize, what type of solution is less problematic for such target group.

Analysis of tools available on the market revealed that simplified graphical domain-specific languages might be the solution to at least some of above-mentioned problems. The question is what is the opinion and preferences of nontechnical persons.

With such problem statement in mind, the following research questions were formulated:

1. What kind of modeling language is easier to learn and use by nontechnical persons?
2. Whether simplified graphical modeling languages are suitable for creating personal as well as simple business applications?

3. MATERIALS AND METHODS

In order to answer the formulated research questions, short workshops on modeling software for mobile devices were conducted. English was the working language. They were designed for persons without previous experience in developing software for mobile devices or an IT background (like computer science studies or in a related field; training in software engineering, modeling languages, programming languages, etc.). After the workshop, participants were asked to voluntarily fill in an anonymous survey.

A single workshop lasted a few hours, and each person could participate in only one workshop. The main workshop topics were: mobile systems for reporting life-threatening situations (what those systems are, why they are needed, how they work and how they should work) and graphical modeling of mobile reporting systems (why modeling could be useful, what tools are available and how to use them, how to read models, how to communicate using models). Participants were also working with exemplary models depicting functionality related to processing the report.

Typical deficiencies of nontechnical persons are inability to handle too many technical details and ease of discouragement, which was also mentioned in (Żyła, 2015). Due to the specificity of the respondents, survey needed simplifications concerning technology choices as well as the number and character of questions.

After analyzing a short list of modeling languages available for the mobile domain, three of them were chosen. Each of them represented a distinguishable class (group) of solutions described in the introduction. 1st group was represented by Unified Modeling Language (UML) as an industrially recognized tool (Arisholm, Briand, Hove & Labiche, 2006); 2nd group by App Inventor (AI) as a tool well recognized by the community (Kowalczyk, Turczynski & Żyła, 2016; "MIT App Inventor", 2017); and 3rd group by AergiaML as a tool designed for nontechnical persons that tries to be an answer for disadvantages of former tools (Żyła, 2015). Models made in those solutions were equivalent – they held the same information, sufficient to fully describe the behavior of an application. Such level of details allows for the code generation resulting in platform-specific executable file with the application or its source code.

The anonymous survey was divided into the following parts:

1. Personal background:
   Age, country of origin, information regarding studies, experience in creating and modeling mobile applications.
2. Tasks:
   T1: Mark from 1 to 6 how easy it is for you to learn and use UML.
   T2: Mark from 1 to 6 how easy it is for you to learn and use App Inventor.
   T3: Mark from 1 to 6 how easy it is for you to learn and use AergiaML.
   T4: What is depicted by the presented model made in UML?
   T5: What is depicted by the presented model made in App Inventor?
   T6: What is depicted by the presented model made in AergiaML?
3. Questions:
   Q1: Do you think that AergiaML allows you to focus on the idea of application and you are not distracted by too many technology-specific details?
   Q2: Do you think that you would be able to learn AergiaML in a degree that allows to create applications fulfilling your everyday needs?

The following examples depict the level of complexity of models in the survey tasks T4–T6: acquiring location of a mobile device before submitting a report, collecting data from the form before submitting a report, managing photos in a gallery.

4. RESULTS AND DISCUSSION

A group of 67 English-speaking persons participated in the workshops. 29 of them filled in an anonymous survey. Among respondents were students and graduates in medicine, administration, sociology, foreign relations, policy making, management and environmental engineering. They originated from more than 12 countries, including Poland, Czech Republic, Germany, Portugal, Russia, Ukraine, USA, Canada, Republic of Korea and Indonesia. None of them had ever created software on his/her own or learned how to use UML, App Inventor, or any other solution. Their age structure is presented in Figure 1.

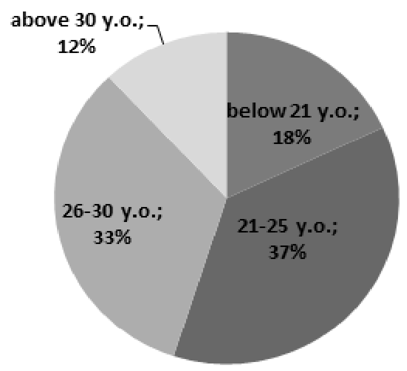
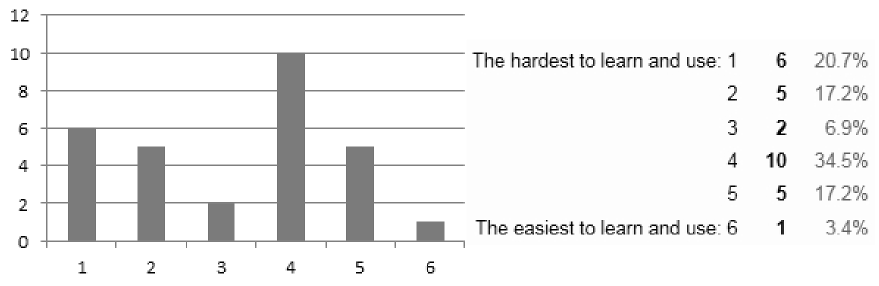


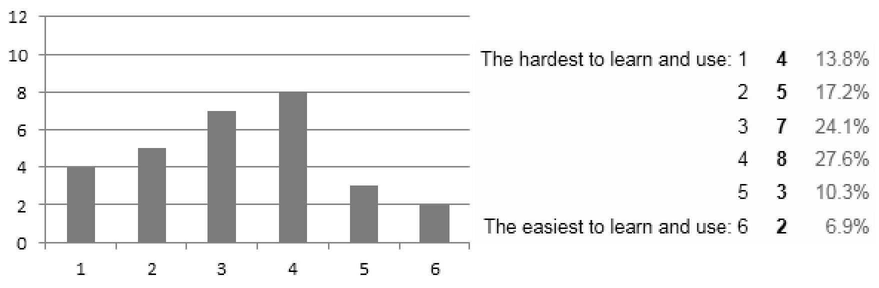
Fig. 1. Age structure of the survey respondents

Respondents, by solving tasks T1–T3, indicated AergiaML as a language that is easiest to learn and use – the average mark being 4.86. UML received the average mark being 3.24, and App Inventor the average mark being 3.21. Distribution of marks is presented by Figures 2–4 (the higher the mark, the better).



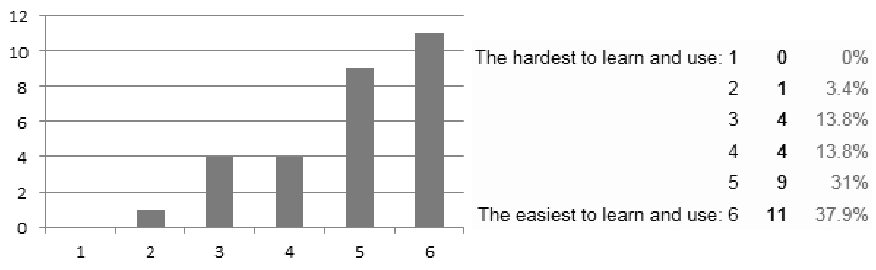
The hardest to learn and use: 1	6	20.7%
2	5	17.2%
3	2	6.9%
4	10	34.5%
5	5	17.2%
The easiest to learn and use: 6	1	3.4%

Fig. 2. Distribution of marks – App Inventor



The hardest to learn and use: 1	4	13.8%
2	5	17.2%
3	7	24.1%
4	8	27.6%
5	3	10.3%
The easiest to learn and use: 6	2	6.9%

Fig. 3. Distribution of marks – UML



The hardest to learn and use: 1	0	0%
2	1	3.4%
3	4	13.8%
4	4	13.8%
5	9	31%
The easiest to learn and use: 6	11	37.9%

Fig. 4. Distribution of marks – AergiaML

The Wilcoxon signed rank test shows that there is no statistically significant difference between declared ease of learn and use of UML and App Inventor (two-sided test p-value of 0.5055; the hypothesis was that the score of UML would be higher than App Inventor). However, there is a statistically significant difference between UML or App Inventor and AergiaML, in favor of the latter (p-value of 0.0007 and 0.0004; the hypothesis was that the score of AergiaML would be higher than any of the two other). Due to multiple testing, the standard significance level of 5% was corrected, using Bonferroni, to 0.016. Calculations were performed using R environment.

When it comes to recognizing what is depicted by the model (tasks T4–T6), the McNemar test shows that the percentage of correct responses for App Inventor and AergiaML was statistically significantly greater than for UML (p-values of 0.0154 and 0.0003; the hypothesis was that the percentage of correct responses for App Inventor or AergiaML would be greater than for UML). However, the percentage of correct responses for App Inventor was not statistically significantly greater than for AergiaML (p-value of 0.1445; the hypothesis was that the percentage of correct responses for App Inventor would be greater than for AergiaML). Due to multiple testing, the standard significance level of 5% was corrected, using Bonferroni, to 0.016. Calculations were performed using R environment (Żyła, 2015).

Respondents also answered questions Q1 and Q2. 79% of them declared that AergiaML allowed them to focus on the idea of application and they were not distracted by too many technology-specific details. In case of Q2, 59% of respondents declared that they would be able to learn AergiaML in a degree that allows to create applications fulfilling their everyday needs. High rate of unsure persons, might be the result of no prior experience concerning software development. The distribution of answers is presented in Figures 5 and 6.

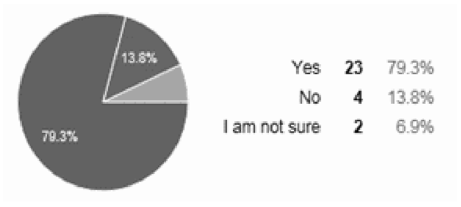


Fig. 5. Distribution of responses to question Q1

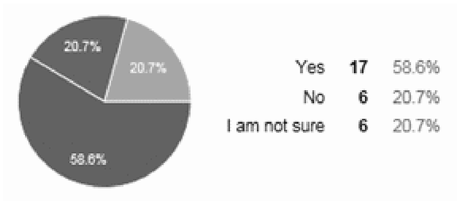


Fig. 6. Distribution of responses to question Q2

5. CONCLUSIONS

In order to determine perspectives of learnability of simplified graphical domain-specific languages for the mobile domain by nontechnical users, a series of workshops were conducted. Highly diversified (origin and education) group of 29 participants without prior knowledge in creating software, after participating in workshops, filled in anonymous surveys.

Analysis of the surveys revealed that nontechnical persons indicated simplified graphical domain-specific languages as a category of languages that is the easiest to learn and use. It was confirmed by highest (although statistically not significant) rate of correct answers concerning functionality of models. Moreover respondents indicated that such languages are suitable for creating personal and simple business application, as they allow to focus on the idea of an application and they could be learned to an extent that allows to individual creation of applications.

Obtained results allows to suggest that it is worth to investigate languages that are based on components that perform complex activities (e.g. saving something to a database, getting access to a GPS sensor), where the role of a user is to show how those components interact by connecting them in chains of complex actions, executed in a particular order. Such approach might be highly useful when involving nontechnical people in a process of software development.

REFERENCES

Abran, A., Khelifi, A., Suryn, W., & Seffah, A. (2003). Usability meanings and interpretations in ISO standards. *Software Quality Journal, 4*(11), 325–338. doi: 10.1023/A:1025869312943

Angeles, S. (2017, February 27). How to make an app: Choosing the right app maker for you. *Business News Daily*. Retrieved from http://www.businessnewsdaily.com/8617-making-an-app.html

Arisholm, E., Briand, L. C., Hove, S. E., & Labiche, Y. (2006). The impact of UML documenttation on software maintenance: an experimental evaluation. *IEEE Transactions on Software Engineering, 32*(6), 365–381.

Brambilla, M., Cabot, J., & Wimmer, M. (2012). *Model-driven software engineering in practice*. Morgan & Claypool Publishers.

Definition – learnability. (n.d.). Retrieved July 2, 2017, from TechTarget, WhatIs.com? website http://whatis.techtarget.com/definition/learnability

Hutchinson, J., Rouncefield, M., & Whittle, J. (2011). Model-driven engineering practices in industry. In *ICSE '11 Proceedings of the 33rd International Conference on Software Engineering, 21-28 May 2011* (pp. 633–642). New York, USA: ACM.

Kapitsaki, G. M., Kateros, D. A., & Pappas, C. (2015). Enabling the deployment of mobile services for end-users: the SMS approach. *Service Oriented Computing and Applications, 9*(1), 21–40.

Kawasaki, B. (2016, May 20). App development should be democratized. *DZone/Mobile Zone*. Retrieved from https://dzone.com/articles/app-development-should-be-democratized

Kesik, J., & Żyła, K. (2011). *Współczesne technologie informatyczne. Technologie MDE w projektowaniu aplikacji internetowych.* Lublin: Politechnika Lubelska.

Kesik, J., Żyła, K., & Nowakowski, K. (2014). Domain-specific languages as tools for teaching 3D graphics. In *Proceedings of the 2nd International Conference on Model-Driven Engineering and Software Development (MODELSWARD '2014), 7-9 January 2014* (pp. 498–503). Lisbon, Portugal: IEEE.

Kowalczyk, R., Turczynski, L., & Żyła, K. (2016). Comparison of App Inventor 2 and Java in creating personal applications for Android on example of a notepad. *Advances in Science and Technology Research Journal, 10*(31), 247–254. doi: https://doi.org/10.12913/22998624/64058

Meulen, R., & Forni, A. A. (2016, November 17). Gartner says Chinese smartphone vendors were only vendors in the global top five to increase sales in the third quarter of 2016. *Gartner*. Retrieved from http://www.gartner.com/newsroom/id/3516317

MIT App Inventor. (n.d.). Retrieved July 3, 2017, from MIT, App Inventor website http://appinventor.mit.edu/explore

Mohagheghi, P., & Dehlen, V. (2008). Where is the proof? - A review of experiences from applying MDE in industry. *Lecture Notes in Computer Science, 5095*, 432–443.

Mohamad, S. N. H., Patel, A., Tew, Y., Latih, R., & Qassim, Q. (2011). Principles and dynamics of block-based programming approach. In *2011 IEEE Symposium on Computers & Informatics, ISCI 2011, Kuala Lumpur, Malaysia, March 20-23, 2011* (pp. 340–345). Kuala Lumpur, Malaysia: IEEE. doi: 10.1109/ISCI.2011.5958938

Rieger, C. (2017). Business apps with MAML: a model-driven approach to process-oriented mobile app development. In *SAC '17 Proceedings of the Symposium on Applied Computing, Marrakech, Morocco, April 4-6, 2017* (pp. 1599–1606). New York, NY: ACM. doi: https://doi.org/10.1145/3019612.3019746

Taylor, H. (2015, November 9). How mobile will transform business in 2016: Forrester. *CNBC.* Retrieved from http://www.cnbc.com/2015/11/09/forrester-mobile-predictions-for-2016.html

Viedma, C. (2010). *Mobile web mashups. The long tail of mobile applications. Master 's thesis.* School of ICT, Stockholm, Sweden.

Żyła, K. (2013). Economic aspects of user-oriented modeling for mobile devices. *Actual Problems of Economics, 4*(142), 334–340.

Żyła, K. (2015). Perspectives of simplified graphical domain-specific languages as communication tools in developing mobile systems for reporting life-threatening situations. *Studies in Logic, Grammar and Rhetoric, 43*(56), 161-175. doi: https://doi.org/10.1515/slgr-2015-0048

Żyła, K., & Kesik, J. (2012). Podsumowanie i kierunki badań nad MDE na Politechnice Lubelskiej. In M. Miłosz & W. Wójcik (Eds.), *Kompetentny absolwent informatyki 2012* (pp. 135–152). Lublin: Polskie Towarzystwo Informatyczne.