Part V – Conclusion

11 Conclusion

This chapter concludes this research work by presenting its main contributions and furthermore points to future work that could be performed to continue this research project to further extend the results obtained.

11.1 Conclusions

The current nature of the business environment of the software industry pushes the software industry organizations to be open to change, and therefore be more flexible and adaptable. A trending way that may guarantee to *"be open to change"* is demonstrated in the Agile Software Development (ASD) paradigm, which could be expended to the field of Software Product Line Engineering (SPLE). Consequently, an emerging paradigm, "Agile Product Line Engineering (APLE)", has been proposed with the aim to reduce the big upfront design, usually practiced in SPLE, and to make the development processes of the product-lines more flexible and adaptable to the current changing environment. This thesis directly addresses the core objective of *"Proposing an Agile Product Line method"*.

Evidently, there are several reasons that encourage companies and organizations to move toward the adoption of *Agile Product Line (APL) methods*. However, there are several potential challenges and risks associated with the adoption of an APL method. It was concluded that the applicability of agile methods for Application Engineering (AE) seems to be highly feasible. Nevertheless, the applicability of agile methods to Domain Engineering (DE) is still challenging and requires more effort, especially when considering traceability and challenges faced by the Product Line Architecture (PLA).

Reviewing the relevant scientific works and practical case studies in the field of APLE has highlighted the different scenarios of combining agile methods and SPL approaches. Some studies have proposed APL methodologies, in which the SPL is used as the main process; where the integration starts from SPL, and then agile practices are added. Other studies have proposed

APL methodologies in which ASD is considered as the main development process, where the integration starts from the agile method, and SPL practices are added. Moreover, the reviewed research works have acknowledged that the main advantage of the APL paradigm is that SPLE could be applied to volatile markets, where changes cannot be easily predicted. In addition, SPLE could be supported by agile methods to develop large-scale product families. The main issue related to combining ASD with SPL is to identify the right level of flexibility, according to the business objectives. The level of flexibility is, thus, a key element in determining whether the approach combines SPLE and ASD at the strategic level, or adopt SPLE at the strategic level and ASD at the tactical one. Accordingly, the APL approach can be applied to real projects only if adequate guidelines are provided on the activities, people, and work-products involved in the project. A software development methodology can satisfy this need; consequently several attempts have been made to propose practical APLE methodologies.

Through this thesis, it was aimed to define a new APL method that presents agile processes, support mechanism, and requirements engineering (RE) techniques. Where the targeted APL method takes advantage of the strengths of the studied APL methodologies and overcomes their weaknesses. When designing a new APL process, first, the basis of the new process should be selected (i.e. start with SPL or ASD). Second, if the basic process is a SPL one, the type of the SPL architecture should be selected (i.e. COPA, KobrA, FORM, etc.). Third, an agile method or a hybrid of agile methods should be selected for the combination. The proposed APL methodologies consist of two integral parts. The first one is dedicated to requirements engineering and the second one is dedicated for the development process.

In this thesis, a Feature-Oriented Agile Product Line method called “Agile Framework for managing evolving Software Product Lines – AgiFPL method” was proposed. AgiFPL targets the development process part. The proposed process consists of a set of units of work (activities, tasks, steps), guidelines, roles, and work-products that allow the people involved (experts, owners, developers, stakeholders, etc.) with the Software Product Line (SPL) to perform the development processes of Domain Engineering (DE) and Application Engineering (AE) aligned with Scrumban and Scrum practices. AgiFPL has indeed presented a Scrumban-inspired process for the DE tier and a Scrum-inspired process for the AE tier. Scrum was adopted as it is considered the mainstream method within development teams, and its practices fit the AE process by handling feedback, iterativeness, incrementally, and adaptability in a systematic way. Scrumban was adopted, as it ascribes more importance to the requirements engineering part than any other agile methodologies; it establishes a structure of limited number of “Production Lines” within its development process. It is furthermore commonly used for maintenance projects, where it helps to deal with the traceability issue in the DE tier.

For the requirement engineering part, this thesis has proposed an Integrated Requirements Engineering framework for Agile Software Product Lines, which is the part providing the syntax and semantics used for expressing the products of an APL method. The aim is to allow analysts and developers to specify requirements that precisely capture the stakeholder’s needs and intentions, as well as to manage product line variabilities.

AgiFPL method was validated and put into practice through an assessment model (i.e. AgiPL-AM), a Criteria-Based Evaluation, and an industrial case study (i.e. TranslogisTIC

project). The case study has demonstrated the applicability of the proposed Agile Product Line methodology (i.e. AgiFPL) onto the establishment of Agile Product Lines for the "Transport and Logistics" Domain. Precisely, the application of AgiFPL in Domain Engineering (DE) and Application Engineering (AE), onto the development of an Outbound Logistics Collaborative Platform, was presented. By this case study, it was shown that the Scrumban-inspired process for DE and Scrum-inspired process for AE have been highly comprehensive for the different people involved. It was observed that the continuous feedback, appropriate iterativeness, and high level of adaptability have eased the communication between the various established roles, as well as the improvement of productivity of the development teams.

The Criteria-Based Evaluation (CBE) has allowed for the identification of the anatomy of the AgiFPL method, as well as its strengths and weaknesses. The CBE has demonstrated that the AgiFPL method is a well-constructed Agile Product Line methodology and has respected almost all the principles and objectives required from an Agile Product Line methodology. However, some improvements are still required, especially for issues related to traceability, consistency management, large organizations, and support tools. Moreover, the AgiPL-AM model has allowed the evaluation of the agility level of the AgiFPL method. The result of the assessment, using AgiPL-AM, has shown that AgiFPL method is strongly agile and has successfully combined the agile principles with SPLE principles.

11.2 Main Contributions

This dissertation has contributed mainly to the field of APLE. The research project has advocated an Agile Software Product Line method in order to help professionals, especially professionals of SPLE community, as well as the ASD community as a whole. In fact, the core contributions of the research work that belong to this thesis have included:

- A multi-evaluation of the Agile Product Line Methodologies with a strong focus on their phases, optional and essential work-products, development processes, as well as the whole lifecycle;
- An adequate definition for "Agile Software Product Line Engineering" (APLE). A definition that fits the undertaken research project;
- A new Feature-Driven APL method called AgiFPL, which presents iterative and incremental development processes Software Product Lines. It furthermore presents a detailed description of the processes, practices, roles, artifacts, communication channels, etc.;
- A Three-phase approach showing how the AgiFPL approach has to be implemented incrementally. The period of implementation depends on the size or the type of the SPL Company (or organization);
- An Integrated Requirements Engineering framework for Agile Software Product Lines that allows analysts and developers to specify requirements. The RE framework was formalized and a meta-model was introduced;

- An assessment model called AgiPL-AM to assess the current situation regarding the combination of agile practices and activities with Software Product Lines. This assessment is considered as an essential step towards a successful integration of agile methods into Software Product Lines;
- The demonstration of the applicability of AgiFPL and its concepts introduced for the different development processes onto a real case study: the TranslogiTIC project.

11.3 Future Work

Future improvements and contributions could be made on the basis of this dissertation. The contributions of this thesis have presented a qualitative leap in the field of Agile Software Product Line Engineering (APLE). However, these contributions have to be completed, extended, and improved upon at different levels in further work.

- The AgiFPL method takes into account the maintenance activities. However, the established processes need to be completed by more explicit maintenance-oriented practices in future works;
- Automation of several tasks can accelerate the proposed process. It is planned in future work to develop a specific tool that ensures the integration of AgiFPL's phases, activities, tasks and work products, including market strategies and mining of legacy assets. Interesting and useful investigations for the academy and industry would be the building of this tool;
- Integrating relevant DevOps and more metrics to AgiFPL method in order to optimize its performance and to improve its applicability within SPLE and the ASD communities;
- The assessment model AgiPL-AM will be evaluated empirically and a number of members of companies will be involved in the evaluation of the applicability of AgiPL-AM; more precisely, in assessing the level of agility of companies, and in evaluating the overall findings of the assessment model;
- Regarding the Integrated Requirements Engineering framework for APLs, several future works will tackle the aim to develop a procedure to discover inconsistencies in mapping results (i.e., generated goal models and/or generated feature models). Moreover, it will be aimed to extend the validation technique in order to cover other properties, such as safe composition (i.e., for all goal models modeled within the Application Engineering (AE) tier, they have at least one correspondent feature model and vice versa);
- With the aim to perform comparative studies, several new case studies in another projects are currently being planned;
- For most of the future studies, the focus lies on capturing quantitative measurements in order to validate statistically new hypothesis, such as studies and research works that compare two methods.

References

Apel, S., Batory, D., Kästner, C., and Saake, G. (2013). *Feature-oriented Software Product Lines: Concepts and Implementation*, 1st ed., Springer-Verlag, Berlin, Heidelberg.

Abo Zaid, L., Kleinermann, F., and De Troyer, O. (2011). Feature Assembly Framework: towards scalable and reusable feature models. In *Proceedings of the 5th Workshop on Variability Modeling of Software-Intensive Systems (VaMoS '11).* Namur, Belgium, pp. 1-9.

Abrahamsson, P., Salo, O., Ronkainen, J. and Warsta, J. (2002). *Agile Software Development Methods: Review and Analysis*. [online] Available at: https://www.vtt.fi/inf/pdf/publications/2002/P478.pdf [Accessed 11 Sep. 2018].

Abrahamsson, P., Warsta, J., Siponen, M. T., and Ronkainen, J. (2003). New directions on agile methods: a comparative analysis. In *Proceedings of the 25th International Conference on Software Engineering*, Portland, Oregon, USA, pp. 244–254.

Abrahamsson, P.,Warsta, J., Siponen, M., Ronkainen, J. (2003). New directions on agile methods: a comparative analysis. In: *Proceedings of the 25th International Conference on Software Engineering*, Portland, OR, USA, pp. 244–254.

Agile Alliance. (2018). *Manifesto for Agile Software Development*. [online] Available at: https://www.agilealliance.org/agile101/the-agile-manifesto/ [Accessed 05 Aug. 2018].

Agile Business Consortium (2014). *Agile Project Management (AgilePM®) Handbook v2*. DSDM Consortium, Kent, UK.

Ajila, S. A, and Kaba, A. B. (2008). Evolution support mechanisms for software product line process. Journal of Systems and Software, 81(10), pp. 1784-1801.

Almeida, E., Balbino, M., Neiva, D., Dilorenzo, E., Silveira, P., Machado, L., Burgos, T., Burégio, V., and Meir S. (2009). *RiPLE: The RiSE Process for Product Line Engineering*. [pdf] Third Brazilian Symposium on Software Components, SBCARS 2009, Architectures and Reuse, SBC - Brazilian Computer Society, Natal, Brazil. Available at: http://www.dimap.ufrn.br/sbcars2009/files/SBCARS2009-Minicurso02-RIPLE.pdf [Accessed 14 Jan. 2018].

Alves, V., Niu, N., Alves, C., and Valença, G. (2010). Requirements engineering for software product lines: A systematic literature review. Information and Software Technology, 52(8), pp. 806-820.

Ambler, S. (2002). *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*. John Wiley & Sons, Inc., New York, NY, USA.

Ambler, S. W. (2002). *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*. 1st ed., John Wiley & Sons Inc., New York, NY, USA.

Ambler, S. W. (2006). *The Agile Unified Process (AUP)*. [online] Ambysoft. Available at: http://www.ambysoft.com/unifiedprocess/agileUP.html [Accessed 21 Aug. 2018].

Ambler, S. W., and Lines, M. (2012). *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise*. IBM Press, Indianapolis, IN, USA.

Ambler, S.W. (2007). Disciplined Agile Software Development: Definition. [Online] Agile Modeling. Available at: http://www.agilemodeling.com/essays/agileSoftwareDevelopment.htm [Accessed 17 Oct. 2018].

America P., Obbink H., van Ommering R., van der Linden F. (2000). CoPAM: A Component-Oriented Platform Architecting Method Family for Product Family Engineering. In: Donohoe P., *Software Product Lines: Experience and Research Directions,* eds., *The Springer International Series in Engineering and Computer Science*, vol 576. Boston, MA: Springer.

America, P. , Hammer, D. , Ionita, M. T., Obbink, H. and Rommes, E. (2005), Scenario-based decision making for architectural variability in product families. Software Process: Improvement and Practice, 10(2), pp. 171–187.

Anderson, D. and Carmichael, A. (2016). *Essential Kanban Condensed*. 1st ed. Lean Kanban Inc., Washington, Seattle.

Anderson, D. J. (2003). *Agile Management for Software Engineering: Applying the Theory of Constraints for Business Results*. Prentice Hall, Upper Saddle River, NJ, USA.

Anderson, D. J. (2010). *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press, Sequim, WA, USA.

António S., Araújo J.A., Silva C. (2009). Adapting the i* framework for software product lines. In *Proceedings of the ER 2009Workshops on Advances in Conceptual Modeling—Challenging Perspectives*, Springer, Berlin, pp. 286–295.

Apel, S., Lengauer, C., Möller, B., and Kaestner, C. (2010). An algebraic foundation for automatic feature-based program synthesis. *Science of Computer Programming*, 75, pp. 1022-1047.

Arcega, L., Font, J., Haugen, Ø., and Cetina, C. (2016). Achieving Knowledge Evolution in Dynamic Software Product Lines. In *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, Suita, pp. 505-516.

Asadi, M., and Ramsin, R. (2008) MDA-Based Methodologies: An Analytical Survey. In: Schieferdecker I., Hartman A. (eds) Model Driven Architecture – Foundations and Applications. ECMDA-FA 2008. Lecture Notes in Computer Science, vol 5095. Springer, Berlin, Heidelberg, pp. 419-431.

Asadi, M., Bagheri, E., Gašević, D., Hatala, M., and Mohabbati, B. (2011). Goal-driven software product line engineering. In *Proceedings of the 2011 ACM Symposium on Applied Computing* (SAC '11). TaiChung, Taiwan, pp. 691-698.

Asadi, M., Gröner, G., Mohabbati, B., and Gasevic, D. (2016). Goal-oriented modeling and verification of feature-oriented product lines. *Software & Systems Modeling* 15(1), pp. 257-279.

Asikainen, T., Männistö, T., and Soininen Kumbang, T. (2007). A domain ontology for modelling variability in software product families. *Advanced Engineering Informatics*, 21(1), pp. 23–40.

Asikainen, T., Mannisto,T., and Soininen, T. (2006). A unified conceptual foundation for feature modelling. In *Proceedings of the 10th International Software Product Line Conference.* Baltimore, MD, USA, pp. 31–40.

Asikainen, T., Männistö,T., and Soininen, T. (2007). Kumbang: a domain ontology for modelling variability in software product families. Advanced Engineering Informatics, 21(1) pp. 23–40.

Association for Project Management. (2019). Agile project management glossary. [Online] Available at: https://www.apm.org.uk/resources/find-a-resource/agile-project-management/glossary/ [Accessed 27 Jan. 2019].

Atkinson, C. (2002). *Component-based product line engineering with UML*, Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA.

Atkinson, C., Bayer, J., Bunse, C., Kamsties, E., Laitenberger, O., Laqua, R., Muthig, D., Paech, B., Wüst, J., and Zettel, J. (2002). *Component-Based Product Line Engineering with UML*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Atkinson, C., Bayer, J., Muthig, D. (2000). Component-Based Product Line Development: The KobrA Approach. In: Donohoe P. *Software Product Lines, The Springer International Series in Engineering and Computer Science, vol 576,* eds. Boston, MA: Springer, pp. 289-309.

Atkinson, C., Muthig, D. (2002). Component-Based Product-Line Engineering with the UML. In: Gacek C. *Software Reuse: Methods, Techniques, and Tools. ICSR 2002. Lecture Notes in Computer Science, vol 2319*, eds. Berlin, Heidelberg: Springer-Verlag.

Auer, K., and Miller, M. (2001). *Extreme Programming Applied: Playing to Win*. 1st ed., Addison-Wesley Professional, Boston, MA, USA.

Auerbach, B. and McCarthy, R. (2014). Does agile+ lean=effective: An investigative study. *Journal of Computer Science and Information Technology*, 2(2), pp. 73–86.

Babar, M.A., Ihme, T., and Pikkarainen, M. (2009). An industrial case of exploiting product line architectures in agile software development. In *Proceedings of the 13th International Software Product Line Conference*, San Francisco, California, USA, pp. 171-179.

Bachmann F., and Bass, L. (2001). Managing variability in software architectures. In Proceedings of the 2001 symposium on Software reusability (SSR'01). New York, NY, USA, pp. 126-132.

Bachmann F., Goedicke M., Leite J., Nord R., Pohl K., Ramesh B., Vilbig A. (2004). A Meta-model for Representing Variability in Product Family Development. In: van der Linden F.J. (eds) Software Product-Family Engineering. PFE 2003. Lecture Notes in Computer Science, vol 3014. Springer, Berlin, Heidelberg, pp. 66–80.

Bachmann F., Goedicke, M., Leite, J., Nord, R., Pohl, K., Ramesh, B., and Vilbig, A. (2004). A Meta-model for Representing Variability in Product Family Development. In: van der Linden F.J., *Software Product-Family Engineering*, eds. PFE 2003, Lecture Notes in Computer Science, vol 3014. Berlin, Heidelberg: Springer-Verlag, pp. 66-80.

Bachmann, F., and Clements, P. (2005). *Variability in Software Product Lines (Technical report CMU/SEI-2005-TR-01)*. [online] Pittsburgh: Software Engineering Institute, Carnegie Mellon University. Available at: http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=7675 [Accessed 19 Apr. 2018].

Balbino, M., de Almeida, E. S., and de Lemos Meira, S. R. (2011). An Agile Scoping Process for Software Product Lines. In *Proceedings of the 23rd International Conference on Software Engineering & Knowledge Engineering*, Miami Beach, USA, pp. 717–722.

Banijamali A., Dawadi R., Ahmad M.O., Similä J., Oivo M., Liukkunen K. (2017) Empirical Investigation of Scrumban in Global Software Development. In: Hammoudi S., Pires L., Selic B., Desfray P. (eds) *Model-Driven Engineering and Software Development*. MODELSWARD 2016. Communications in Computer and Information Science, vol 692. Springer, Cham, pp. 229-248.

Baskerville, R., Levine, L., Pries-Heje, J., Balasubramaniam, R., and Slaughter, S. (2001). How internet software companies negotiate quality. *IEEE Computer*, 34(5), pp. 51–57.

Batory D., Sarvela JN., Rauschmayer A. (2004) Scaling step-wise refinement. *IEEE Transactions on Software Engineering (TSE)*, 30(6), pp. 355–371.

Batory, D.S., and O'Malley, S.W. (1992). The Design and Implementation of Hierarchical Software Systems with Reusable Components. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 1(4), pp. 355-398.

Bayer, J., Gacek, C., Muthig, D., and Widen, T. (2000). PuLSE-I: Deriving instances from a product line infrastructure. In *Proceedings Seventh IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS 2000)*, Edinburgh, UK, pp. 237-245.

Beck, K., and Andres, C. (2004). *Extreme Programming Explained: Embrace Change*. 2nd ed., Addison-Wesley, Boston, MA, USA.

Behjati, R., Yue, T., Briand, L., and Selic, B. (2013). SimPL: A product-line modeling methodology for families of integrated control systems. Information and Software Technology, 55(3), pp. 607-629.

Bellucci, A., Jacucci, G., Kotkavuori, V., Serim, B., Ahmed, I., and Ylirisku, S. (2015). Extreme Co-design: Prototyping with and by the User for Appropriation of Web-connected Tags. In: Díaz P., Pipek V., Ardito C., Jensen C., Aedo I., Boden A. (eds) End-User Development. IS-EUD 2015. Lecture Notes in Computer Science, vol 9083. Springer, Cham, pp. 109–124.

Blau, B., and Hildenbrand, T. (2011). Product Line Engineering in Large-Scale Lean and Agile Software Product Development Environments - Towards a Hybrid Approach to Decentral Control and Managed Reuse. In Proceedings of the 6th International Conference on Availability, Reliability and Security, Vienna, pp. 404-408.

Boeckle, G., Knauber, P., Pohl, K., and Schmid, K. (2004). *Software-Produktlinien – Methoden, Einführung und Praxis* (in German), dpunkt.verlag, Heidelberg.

Boehm, B. (1976). Software engineering. *IEEE Transactions on Computers*, C-25(12), pp. 1226–1241.

Boehm, B. and Basili, V. R. (2001). Software defect reduction top 10 list. *Computer*, 34(1), pp. 135–137.

Boehm, B. and Turner, R. (2003). *Balancing Agility and Discipline: A Guide for the Perplexed*. Addison-Wesley Professional, Boston, MA, USA.

Boehm, B. W. (1988). A Spiral Model of Software Development and Enhancement. *IEEE Computer*, 21 (5), pp. 61–72.

Bolander, W. J., Clements, P. C. and Krueger, C. (2016). It Takes a Village: Why PLE Technology Solutions Require Ecosystems of PLE Technology Providers. *INCOSE International Symposium*, 26(1), pp. 678-690.

Book, M., Gruhn, V., and Striemer, R. (2016). *Tamed Agility: Pragmatic Contracting and Collaboration in Agile Software Projects*. 1st ed., Springer-Cham, Zurich, Switzerland.

Borba, C., and Silva, C. (2009). A Comparison of Goal-Oriented Approaches to Model Software Product Lines Variability. In: Heuser C.A., Pernul G. (eds) Advances in Conceptual Modeling - Challenging Perspectives. ER 2009. Lecture Notes in Computer Science, vol 5833. Springer, Heidelberg, pp. 244-253

Bosch J. (2000). *Design and use of software architectures: Adopting and evolving a product-line approach*. ACM Press/Addison-Wesley, New York, NY, USA.

Bosch, J. (2000). *Design and Use Software Architectures: Adopting and Evolving a Product-Line Approach.* Addison-Wesley Professional, Boston, MA, USA.

Bosch, J. and Bosch-Sijtsema, P.M. (2011). Introducing agile customer-centered development in a legacy software product line. *Software: Practice and Experience,* 41(8), 871–882. https://doi.org/10.1002/spe.1063

Bourque, P. and Fairley, R. E. (2014). *Guide to the Software Engineering Body of Knowledge (SWEBOK(R)): Version 3.0*. 3rd ed., IEEE Computer Society Press, Los Alamitos, CA, USA.

Bozheva T., Gallo M.E. (2005) Framework of Agile Patterns. In: Richardson I., Abrahamsson P., Messnarz R. (eds) Software Process Improvement. *EuroSPI 2005. Lecture Notes in Computer Science*, vol 3792. Springer, Berlin, Heidelberg, pp. 4-15.

Brechner, E. (2015). *Agile Project Management with Kanban*. Redmond, 1st ed. Microsoft Press, Washington, Seattle.

Brezočnik, L., and Majer, Č. (2016). Comparison of agile methods: Scrum, Kanban, and Scrumban. In: *Proceedings of the 19th International Multiconference Information Society - IS 2016: Collaboration, software and services in information society, Volume C*. Ljubljana, Slovenia, pp. 30-33.

Brownsword, L. and Clements, P. (1996). *A case study in successful product line development*. Technical report, cmu/sei-96-tr-016, Software Engineering Institute, Carnegie Mellon University.

Broy, M. (2013) Domain Modeling and Domain Engineering: Key Tasks in Requirements Engineering. In: Münch J., Schmid K. (eds) Perspectives on the Future of Software Engineering. Springer, Berlin, Heidelberg

Buchan, J. (2014). An Empirical Cognitive Model of the Development of Shared Understanding of Requirements. *Communications in Computer and Information Science*, pp. 165–179.

Buxmann, P., Diefenbach, H., and Hess, T. (2013). *The Software Industry: Economic Principles, Strategies, Perspectives*. Springer-Verlag, Berlin, Heidelberg, Germany.

Carbon R, Lindvall M, Muthig D, Costa P. (2006). Integrating product line engineering and agile methods: Flexible design up-front vs. incremental design. APLE '06: First International Workshop on Agile Product Line Engineering (in conjunction with SPLC), Baltimore, MD, U.S.A. [Online] Available at: https://pdfs.semanticscholar.org/fc54/c61e3bd6a0815a0254bf984ee6ccef05725f.pdf [accessed 14 Oct. 2018].

Carbon, R., Knodel, J., Muthig, D., and Meier, G. (2008). Providing feedback from application to family engineering—The product line planning game at the testo ag. In Proceedings of the 12th International Software Product Line Conference - SPLC '08. IEEE Computer Society: Washington, DC, USA, pp. 180–189.

Chen, K., Zhang, W., Zhao, H., and Mei, H. (2005). An approach to constructing feature models based on requirements clustering. In *Proceedings of the International Conference Requirements Engineering (RE)*. IEEE Computer Society, pp 31–40.

Chen, L., and Babar, A. M. (2011). Asystematic review of evaluation of variability management approaches in software product lines, *Information and Software Technology*, 53(4), pp. 344–362.

Christopher, M. (2011). *Logistics & Supply Chain Management*. Pearson Education Limited, Edinburgh Gate, Harlow, UK.

Classen, A., Heymans, P., and Schobbens, PY. (2008) What's in a Feature: A Requirements Engineering Perspective. In: Fiadeiro J.L., Inverardi P. (eds) Fundamental Approaches to Software Engineering. FASE 2008. Lecture Notes in Computer Science, vol 4961. Springer, Berlin, Heidelberg, pp. 16-30.

Classen, A., Heymans, P., Schobbens, PY. (2008) What's in a Feature: A Requirements Engineering Perspective. In: Fiadeiro J.L., Inverardi P. (eds) Fundamental Approaches to Software Engineering. FASE 2008. Lecture Notes in Computer Science, vol 4961. Springer, Berlin, Heidelberg

Clements, D., and BigLever (2015). Product Line Engineering Comes to the Industrial Mainstream. *INCOSE International Symposium*, 25(1), pp. 1305-1319.

Clements, P., and Northrop, L. (2001). *Software Product Lines: Practices and Patterns*, 1st ed., Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA.

Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., Nord, R., and Stafford, J. A. (2010). Documenting Software Architectures: Views and Beyond, 2nd ed., Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA.

Clements, P., Gregg, S.P., Krueger, C., Lanman, J., Rivera, J., Scharadin, R., Shepherd, J.T., and Winkler, A.J. (2014). Second Generation Product Line Engineering Takes Hold in the DoD. *Cross Talk - The Journal of Defense Software Engineering*, 27(1), pp. 12-18.

Clements, P., Krueger, C. W. (2002). Point/counterpoint: Being proactive pays off/eliminating the adoption barrier. *IEEE Software*, 19(4), pp. 28–31.

Cline, A. (2015). *Agile Development in the Real World: How and Why to Use Agile Methods*. 1st ed. Springer Science+Business Media, New York, NY, USA.

Cockburn A. (2007). *Agile Software Development: The Cooperative Game*. 2nd ed., Addison-Wesley Professional, Boston, MA, USA.

Cockburn, A. (2000). *Writing Effective Use Cases*. 1st ed., Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Cockburn, A. (2002). *Agile Software Development*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Cockburn, A. (2006). *Agile Software Development: The Cooperative Game,* 2nd ed., Addison-Wesley Professional, Boston, MA, USA.

Cockburn, A. (2008). Using both incremental and iterative development. *Crosstalk: The Journal of Defense Software Engineering,*21(5), pp. 27–30.

Codenie, W., González-Deleito, N., Deleu, J., Blagojevic, V., Kuvaja, P., Similä, J. (2009). A model for trading off flexibility and variability in software intensive product development. In Proceedings of Third International Workshop on Variability Modelling of Software-intensive Systems - VaMoS ’09, Seville, Spain, ICB Research Report, vol. 29, Universität Duisburg-Essen, pp. 61–70.

Cohn, M. (2004). *User Stories Applied for Agile Software Development*. 1st ed., Pearson Education Inc., Boston, MA, USA.

Cohn, M. (2005). *Agile Estimating and Planning*. Prentice Hall PTR, Upper Saddle River, NJ, USA.

Cohn, M. (2013). *Succeeding with Agile: Software Development Using Scrum*. 1st ed., Addison-Wesley Professional, Upper Saddle River, NJ, USA.

Cohn, M. (2014). *Agile Estimating and Planning*. 1st ed. Pearson Education Inc., Upper Saddle River, NJ, USA.

Colanzi, T.E., Vergilio, S.R., Gimenes, I.M.S., and Oizumi, W.N. (2014). A search-based approach for software product line design. In: *Proceedings of the 18th International Software Product Line Conference*, Florence, Italy, pp. 237–241 .

CollabNet and VersionOne (2019). *The 13th Annual State of Agile Report*. [online] Available at: https://www.stateofagile.com/#ufh-i-521251909-13th-annual-state-of-agile-report/473508 [Accessed 21 May 2019].

Conboy, K. (2008). A Framework of Method Agility in Information Systems Development. PhD. Galway, Ireland: University of Limerick, p. 306.

Cooper, K. and Franch, X. (2006). APLE - 1st international workshop on agile product line engineering. In: *The 10th International Software Product Line Conference (SPLC 2006)*. IEEE Computer Society, Los Alamitos, Orange County, California, USA, pp. 205–206.

Coplien, J., Hoffman, D., and Weiss, D.M. (1998). Commonality and variability in software engineering, *IEEE Software*, 15(6), pp. 37–45.

Craddock, A. (2013). *Agile Project Management and Scrum (Pocketbook)*. DSDM Consortium, Kent, UK.

Czarnecki, K, and Eisenecker, U. (2000). *Generative programming: Methods, tools, and applications*. ACM Press/Addison-Wesley, New York, NY, USA, p. 34.

Czarnecki, K. and Wasowski, A. (2007). Feature diagrams and logics: there and back again. In *Proceedings of the 11th International Software Product Line Conference - SPLC2007*, Kyoto, Japan, pp.23–34.

Czarnecki, K., and Eisenecker, U. (2000). *Generative programming: Methods, tools, and applications*. 1st ed., ACM Press/Addison-Wesley Publishing Co., Boston, MA, USA, p. 34.

Czarnecki, K., Grünbacher, P., Rabiser, R., Schmid, K., and Wąsowski, A. (2012). Cool features and tough decisions: a comparison of variability modeling approaches. In *Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems (VaMoS '12)*. Leipzig, Germany, pp. 173-182.

Czarnecki, K., Grünbacher, P., Rabiser, R., Schmid, K., Wa˛sowski A. (2012). Cool features and tough decisions: A comparison of variability modeling approaches. In Proceedings of the International Workshop on Variability Modelling of Software-intensive Systems (VaMoS). ACM Press, pp 173–182.

da Silva, I. F. (2012). An agile approach for software product lines scoping. In *Proceedings of the 16th International Software Product Line Conference - Volume 2 (SPLC '12)*, Vol. 2., Salvador, Brazil, pp. 225-228.

da Silva, I. F. (2013). *A Scrum-inspired Process for Software Product Lines Scoping*. PhD. Federal University of Pernambuco.

da Silva, I. F., Neto, P., O'Leary, P., de Almeida, E., and de Lemos Meira, S.R. (2011). Agile Software product lines: a systematic mapping study. *Software: Practice and Experience*, 41(8) 2011, pp. 899–920.

da Silva, I.F., Neto, P.A., O'Leary, P., Almeida, E.S., & Meira, S.R. (2015). Using a multi-method approach to understand Agile software product lines. Information & Software Technology, 57, pp. 527-542.

Dalpiaz, F., Franch, X., Horkoff, J.J.C. (2016). iStar 2.0 Language Guide (v3). [Online] Available at: https://arxiv.org/pdf/1605.07767v3.pdf [Accessed 23 Jan. 2019].

Davis, A. M. (2005). *Just enough requirements management: where software development meets marketing.* Dorset House Publishing, New York, NY, USA.

de Souza, D. S., and Vilain, P. (2013). Selecting Agile Practices for Developing Software Product Lines. In *Proceedings of the 25th International Conference on Software Engineering & Knowledge Engineering*, Boston,USA, pp. 220–225.

de Souza, D. S., and Vilain, P. (2013). Selecting Agile Practices for Developing Software Product Lines. In *Proceedings of the 25th International Conference on Software Engineering & Knowledge Engineering, SEKE2013*, Boston, USA, pp. 220–225.

Demuth, A., Lopez-Herrejon, R.E., and Egyed, A. (2014). Automatic and incremental product op- timization for software product lines. In *Proceedings of the 2014 IEEE 7th International Conference on Software Testing, Verification and Validation (ICST 2014)*, IEEE, pp. 31–40

Denzin, N. K. and Lincoln, Y. S. (2017). The sage handbook of qualitative research. 5th ed., Sage Publications Inc., Thousand Oaks, CA, USA.

Derby, E., and Larsen, D. (2006). *Agile Retrospectives: Making Good Teams Great*. The Pragmatic Bookshelf, Raleigh, NC, USA.

Dhungana, D., Grünbacher, P., Rabiser, R., and Neumayer, T. (2010). Structuring the modeling space and supporting evolution in software product line engineering. *Journal of Systems and Software*, 83(7), pp. 1108–1122

Díaz Fernández, J. (2012). *Agile Construction and Evolution of Product-Line Architectures*. PhD. Universidad Politécnica de Madrid (UPM).

Díaz Fernández, J., Pérez Benedí, J. , Yagüe Panadero, A., and Garbajosa Sopeña, J. (2011). Tailoring the Scrum Development Process to Address Agile Product Line Engineering. In *Proceedings of Jornadas deIngeniería del Software y base de Datos - (JISBD 2011)*, España, A Coruña, pp. 457-469.

Díaz, J., Pérez, J., Alarcón, P. P., and Garbajosa, J. (2011). Agile product line engineering—A systematic literature review. *Software: Practice and Experience*, 41(8), pp. 921–941.

Díaz, J., Pérez, J., and Garbajosa, J. (2014). Agile product-line architecting in practice: A case study in smart grids. *Information and Software Technology*, 56(7), pp. 727-748.

Díaz, J., Pérez, J., and Garbajosa, J. (2014). Agile product-line architecting in practice: A case study in smart grids. *Information and Software Technology*, 56(7), pp. 727–748.

Dijkstra, E. W. (1972). Chapter I: Notes on structured programming. In: O. J. Dahl, E. W. Dijkstra, and C. A. R. Hoare, *Structured programmin*, eds. London: Academic Press Ltd., pp. 1–82.

Dingsøyr, T., Dybå, T., and Moe, N.B. (2010). Agile Software Development: An Introduction and Overview. In: Dingsøyr, T., Dybå, T., and Moe, N. eds. *Agile Software Development*. Berlin, Heidelberg: Springer-Verlag, pp. 1–13.

Dorfman, M., and Thayer, R. H., (1997). Software Requirements Engineering. *IEEE Computer Society Press*, Los Alamitos (CA), pp. 7-22.

dos Santos Jr., A. F., Lucena Jr., V. F. (2010). SCRUMPL - Software Product Line Engineering with Scrum. In *Proceedings of ENASE 2010 - Conference on Evaluation of Novel Approaches to Software Engineering*, Athens, Greece, pp. 239-244.

DSDM Consortium (2014). *The DSDM Agile Project Framework*. DSDM Consortium, Kent, UK.

Dybå, T., and Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information & Software Technology*, 50(9-10), pp. 833-859.

Eklund U., Bosch J. (2012) Applying Agile Development in Mass-Produced Embedded Systems. In: Wohlin C. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2012. Lecture Notes in Business Information Processing, vol 111. Springer, Berlin, Heidelberg, pp. 31-46.

Elshandidy, H. and Mazen, H. (2013). Agile and Traditional Requirements Engineering: A Survey. *International Journal of Scientific and Engineering Research*, 4(9), pp. 473-482.

Engels G., and Sauer S. (2010). A Meta-Method for Defining Software Engineering Methods. In Engels G., Lewerentz C., Schäfer W., Schürr A., Westfechtel B. (eds) Graph Transformations and Model-Driven Engineering. Lecture Notes in Computer Science, vol 5765, Springer, Berlin, Heidelberg, pp. 411-440.

Farahani, F. F. and Ramsin, R. (2014). Methodologies for Agile Product Line Engineering: A Survey and Evaluation. In *Proceedings of the 13th International Conference on New Trends in Software Methodologies, Tools and Techniques, SoMeT_14*. Amsterdam: IOS Press BV, pp. 545-564.

Farahani, F.F., and Ramsin, R. (2017). GMAP : A Generic Methodology for Agile Product Line Engineering. In Proceedings of the 12th International Conference on Software Engineering Advances, ICSEA2017, pp. 157-166.

Feng, K., Lempert, M., Tang, Y., Tian, K., Cooper, K., and Franch, X. (2007). Developing a survey to collect expertise in agile product line requirements engineering. In Agile 2007 Conference, International Research-in-Progress Workshop on Agile Software Engineering (RWASE '07), Washington DC, USA, pp. 1–4.

Ferreira, F., Borba, P., Soares, G., and Gheyi, R. (2012). Making software product line evolution saferIn *Proceedings of the 6th Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS 2012),* IEEE, pp. 21–30

Flores, R., Krueger, C.W., and Clements, P.C. (2012). Mega-scale product line engineering at General Motors. In *Proceedings of the 16th International Software Product Line Conference - Volume 1 (SPLC'12)*. Salvador, Brazil, pp. 259-268.

Font, J., Arcega, L., Haugen, Ø., and Cetina, C. (2017). Leveraging variability modeling to address metamodel revisions in model-based software product lines. *Computer Languages, Systems and Structures*, 48, pp. 20–38

Foreman, J. (1996). Product Line Based Software Development: Significant Results, Future Challenges. In *Proccedings of the 6th Software Technology Conference (STC'96)*. Salt Lake City, UT, USA.

Fowler, M. (2010). *Domain-Specific Languages*. 1st ed., Addison-Wesley Professional, Boston, MA, USA.

Frankel, D. S. (2003). *Model Driven Architecture: Applying MDA to Enterprise Computing*, 1st ed., Wiley Publishing Inc., Indianapolis, Indiana, USA.

Fraunhofer-Institut für Experimentelles Software Engineering, IESE. (2001). *Component-Based Software Engineering: The KobrA Approach*. [online] Fraunhofer-Publica. Available at: http://publica.fraunhofer.de/documents/N-7783.html [Accessed 09 Aug. 2018].

Gambill, P. (2013). Scrumban: A different way to be Agile. [Blog] Deloitte Digital. Available at: https://www.deloittedigital.com/us/en/blog-list/2013/scrumban--a-different-way-to-be-agile.html [Accessed 16 Jan. 2018].

Ghanam, Y., and Maurer, F. (2008a). An Iterative Model for Agile Product Line Engineering. In *Proceedings of the 12th International Software Product Line Conference Doctoral Symposium*, Limerick, Ireland, pp. 377–384.

Ghanam, Y., and Maurer, F. (2009). Extreme Product Line Engineering: Managing Variability and Traceability via Executable Specifications. In *Proceedings of the Agile Conference 2009*, Chicago, (USA), pp. 41-48.

Ghanam, Y., and Maurer, F. (2010) Extreme Product Line Engineering – Refactoring for Variability: A Test-Driven Approach. In: Sillitti A., Martin A., Wang X., Whitworth E. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2010. Lecture Notes in Business Information Processing, vol 48. Springer, Berlin, Heidelberg, pp. 43-57.

Ghanam, Y., Andreychuk, D., and Maurer, F. (2010). Reactive Variability Management in Agile Software Development, in *Proceedings of Agile Conference - (AGILE 2010)*, Orlando, Florida, USA, pp. 27–34.

Ghanam, Y., Andreychuk, D., and Maurer, F. (2010). Reactive Variability Management in Agile Software Development, in Proceedings of Agile Conference - (AGILE 2010), Orlando, Florida, USA, pp. 27–34.

Ghanam, Y., Cooper, K., Abrahamsson, P., and Maurer, F. (2009b). A report on the the first XP workshop on agile product line engineering. XP'09, vol. 31. Springer, Berlin, pp. 215–216.

Ghanam, Y., Park, S., and Maurer, F. (2008b). A Test-Driven Approach to Establishing & Managing Agile Product Lines. In *Proceedings of the 5th Software Product Line Testing Workshop (SPLiT 2008) in conjunction with SPLC 2008*, Limerick, Ireland.

Gibbs, R. D. (2006). *Project Management with the IBM®Rational Unified Process®: Lessons From The Trenches*. IBM Press, Indianapolis, IN, USA.

Gilb, T. (2003), 6.2.4 Evolutionary Project Management (Evo): How to Manage Project Benefits and Costs. *INCOSE International Symposium*, 13(1), pp. 364-373.

Goldman, S., Nagel, R., Preiss, K., and Iacocca, L. (1995). Agile competitors and virtual organizations : strategies for enriching the customer. Van Nostrand Reinhold, New York, NY, USA.

Gomaa, H. (2004). *Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures*, vol. 8, Addison-Wesley Professional, Boston, MA, USA.

Gomaa, H. (2005). *Designing software product lines with UML: from use cases to pattern-based software architectures*, Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA.

Gomaa, H. (2011). *Software Modeling and Design: Uml, Use Cases, Patterns, and Software Architectures*, 1st ed., Cambridge University Press, New York, NY, USA.

Gomaa, H., and Gianturco, M. (2002). Domain Modeling for World Wide Web Based Software Product Lines with UML. In *Gacek C. (eds) Software Reuse: Methods, Techniques, and Tools. ICSR 2002. Lecture Notes in Computer Science, vol 2319*. Springer, Berlin, Heidelberg, pp. 78-92.

Gomaa, H., and Hussein, M. (2004). Software reconfiguration patterns for dynamic evolution of software architectures. In: *Proceedings of the 4th Working IEEE/IFIP Conference on Software Architecture*, Oslo, Norway, pp. 79–88.

Gregg, S. P., Scharadin, R. and Clements, P. C. (2016), The Best of Both Worlds: Agile Development Meets Product Line Engineering at Lockheed Martin. In *26th Annual INCOSE International Symposium*, 26, pp. 951-965.

Haidar, H., Kolp, M. and Wautelet, Y. (2017). Agile Product Line Engineering: The AgiFPL Method.In Proceedings of the 12th International Conference on Software Technologies - Volume 1: ICSOFT, Madrid, Spain, pp. 275-285.

Haidar, H., Kolp, M. and Wautelet, Y. (2018). Formalizing Agile Software Product Lines with a RE Metamodel.In Proceedings of the 13th International Conference on Software Technologies - Volume 1: ICSOFT, Porto, Portugal, pp. 90-101.

Haidar, H., Kolp, M., and Wautelet, Y. (2019) An Integrated Requirements Engineering Framework for Agile Software Product Lines. In: van Sinderen M., Maciaszek L. (eds) Software Technologies. ICSOFT 2018. Communications in Computer and Information Science, vol 1077. Springer, Cham, pp. 124-149.

Hall, T., Beecham, S., and Rainer, A. (2002). Requirements problems in twelve software companies: An empirical analysis. *IEE Proceedings Software*, 149(5), pp. 153–160.

Hanssen G. K. (2007). Agile software product line engineering. DT8100 Essay. Availble at: https://pdfs.semanticscholar.org/5c4f/b2185c78570a3347d4a7614f03f95685603b.pdf [accessed 13 Oct. 2018].

Hanssen G. K. (2010). Opening up software product line engineering. In Proceedings of the 2010 ICSE Workshop on Product Line Approaches in Software Engineering - PLEASE ’10. ACM, New York, NY, USA, pp. 1–7.

Hanssen, G. K. (2011). Agile software product line engineering: enabling factors. *Software: Practice and Experience*, 41(8), pp. 883–897.

Hanssen, G. K., and Fægri, T. E. (2008). Process fusion: An industrial case study on agile software product line engineering. *Journal of Systems and Software*, 81(6), pp. 843–854.

Hanssen, G. K., and Fægri, T. E. (2008). Process fusion: An industrial case study on agile software product line engineering. Journal of Systems and Software, 81(6), pp. 843–854.

Hanssen, G., Stålhane, T., and Myklebust, T. (2018). SafeScrum® – Agile Development of Safety-Critical Software. Springer, Cham, Switzerland.

Haugen, Ø., Wasowski, A., & Czarnecki, K. (2013). CVL: common variability language. In *Proceedings of the 17th International Software Product Line Conference (SPLC '13)*. Tokyo, Japan, pp. 277-277.

Hazzan, O. and Dubinsky, Y. (2008). *Agile Software Engineering*. Springer-Verlag, London, UK.

Heng, S. (2017). *Impact of unified user-story-based modeling on agile methods: aspects on requirements, design and life cycle management*. PhD. Louvain Research Institute in Management and Organizations (LouRIM), UCLouvain.

Hevner A. R., March, S., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS Quarterly 2004*, 28(1), pp. 75-105.

Highsmith, J. (2002). *Agile Software Development Ecosystems*. Addison-Wesley Professional, Boston, MA, USA.

Highsmith, J. (2009). *Agile Project Management: Creating Innovative Products*. 2nd ed. Addison-Wesley Professional, Boston, MA, USA.

Highsmith, J. A. (2013). *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*. [ebook] Boston: Addison-Wesley Professional. Available at: https://www.oreilly.com/library/view/adaptive-software-development/9780133489491/ [Accessed 20 Aug. 2018].

Highsmith, J., and Cockburn, A. (2001). Agile Software Development: The Business of Innovation, *Computer*. 34(9), pp. 120-122.

Hohl, P., Ghofrani, J., Münch, J., Stupperich, M., Schneider, K. (2017b). Searching for common ground: Existing literature on automotive agile software product lines. In Proceedings of the 2017 International Conference on Software and System Process (ICSSP 2017), pp. 70-79.

Hohl, P., Münch, J., Schneider, K., and Stupperich, M. (2016). Forces that Prevent Agile Adoption in the Automotive Domain. In *Proceedings of PROFES'16*, Vol. 10027. Springer International Publishing, pp. 468–476.

Hohl, P., Münch, J., Schneider, K., and Stupperich, M. (2017). Real-Life Challenges on Agile Software Product Lines in Automotive. In *Proceedings of PROFES'17*, Springer International Publishing, pp.28–36.

Hohl, P., Stupperich, M., Münch, J. and Schneider, K. (2018). An Assessment Model to Foster the Adoption of Agile Software Product Lines in the Automotive Domain. In *2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, Stuttgart, Germany, pp. 1-9.

Horcas, J.M., Cortiñas, A., Fuentes, L., & Luaces, M.R. (2018). Integrating the common variability language with multilanguage annotations for web engineering. In *Proceedings of the 22nd International Systems and Software Product Line Conference*, vol. 1, Gothenburg, Sweden, pp. 196-207.

Hunt, A., and Thomas, D. (2000). *The Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley Professional, Boston, MA, USA.

Hunt, J. (2006). *Agile Software Construction*. Springer-Verlag, London, UK.

IEEE (2007). Draft Recommended Practice for the Customer-Supplier Relationship in Agile Software Projects. P1648/D5.

IEEE, (2007). ISO/IEC Standard for Systems and Software Engineering - Recommended Practice for Architectural Description of Software-Intensive Systems. *ISO/IEC 42010 IEEE Std 1471-2000*, 1st ed., pp.c1-24.

ISO/IEC (2013). ISO/IEC 15504-6:2013 - Information technology - Process Assessment Model, ISO copyright office, Geneva, Switzerland.

Jahn, M., Rabiser, R., Grünbacher, P., Loberbauer, M., Wolfinger, R., and Mossenbock, H. (2012). Supporting model maintenance in component-based product lines. In P*roceedings of the Joint Working IEEE/IFIP Conference on Software Architecture (WICSA) & European Conference on Software Architecture (ECSA)*, IEEE, pp. 21–30

Jaqueira, A., Lucena, M., Alencar, F. M. R., Castro, J., and Aranha, E. (2013). Using i* Models to Enrich User Stories. In Proceedings of the 6th International i* Workshop 2013, CEUR-WS.org, Valencia, Spain, pp. 55–60.

Jasmine, K. S., and Vasantha, R. (2010). A new capability maturity model for reuse based software development process. *International Journal of Engineering and Technology*, 2(1), pp. 112-116.

John, D., McGregor, J.D., and Monteith, Y. (2012). A Three Viewpoint Model of Software Ecosystems. In *Proceedings of the 16th IASTED International Conference on Software Engineering Applications*, Las Vegas, NV, USA, 970-023.

Johnson, E. (2016). *Why Switch from Scrum to Scrumban*. [Blog] Intland Software. Available at: https://content.intland.com/blog/agile/scrum/why-switch-from-scrum-to-scrumban [Accessed 12 Jan. 2019].

Kakarontzas, G., Stamelos, I., Katsaros, P. (2008). Product line variability with elastic components and test-driven development. In Proceedings of the 2008 International Conference on Computational Intelligence for Modelling Control and Automation - CIMCA '08. IEEE Computer Society: Washington, DC, USA. pp. 146–151.

KäKölä, T., and Dueñas, J. C. (2006). *Software Product Lines: Research Issues in Engineering and Management,* Springer-Verlag, Berlin, Heidelberg.

kanbanize.com (2019). *Kanban: Lead Time vs Cycle Time - Details Explained*. [Online] Businessmap Ltd. Available at: https://kanbanize.com/kanban-resources/kanban-software/kanban-lead-cycle-time/ [Accessed 07 Apr. 2019].

Kane D. (2003). Introducing agile development into bioinformatics: An experience report. In Proceedings of the Agile Development Conference, ADC 2003, Utah, U.S.A., pp. 132–139.

Kang K. C., Sugumaran V., and Park S. (2010). *Applied Software Product Line Engineering,* 1st ed., Auerbach Publications, Boston, MA, USA.

Kang, K., Cohen, SG., Hess, J. A., Novak, WE., Peterson, A. S. (1990). *Feature-oriented domain analysis (FODA) feasibility study*. Technical Report CMU/SEI-90-TR-21, Carnegie-Mellon University, SEI.

Kang, K.C., Kim, S., Lee, J., Kim, K., Kim, G. J., Shin, E. (1998). FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures. *Annals of Software Engineering*, 5, pp. 143-168.

Kang, K.C., Lee, J., and Donohoe, P. (2002). Feature-oriented product line engineering, *IEEE Software*, 19(4), pp. 58-65.

Känsälä, K. (2004). Good-enough software process in Nokia. In Proceedings of Product Focused Software Process Improvement - PROFES ’04 - (Lecture Notes in Computer Science, vol. 3009). Springer, Berlin, pp. 424–430.

Karam, M., Dascalu, S., Safa, H., Santina, R., Koteich, Z. (2008). A product-line architecture for web service-based visual composition of web applications. *Journal of Systems and Software*, 81(6) pp. 855–867.

Karvonen, T., Rodriguez, P., Kuvaja, P., Mikkonen, K., and Oivo, M. (2012). Adapting the lean enterprise self-assessment tool for the software development domain. In *38th EUROMICROConference on Software Engineering and Advanced Applications (SEAA).* IEEE Press, pp. 266-273.

Kautz, K. (2010). Participatory Design Activities and Agile Software Development. In IFIP WG8.2/8.6 International Working Conference, pp.303–316.

Kenneth, S. R. (2013). *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley Professional, Upper Saddle River, NJ, USA.

Kim, S., Lee, H., Kwon, Y., Yu, M., and Jo, H. (2016). Our Journey to Becoming Agile: Experiences with Agile Transformation in Samsung Electronics. In Proc. Of Asia-Pacific Software Engineering Conference. IEEE, 377–380.

Kircher, M., and Hofman, P. (2012). Combining systematic reuse with Agile development: experience report. In *Proceedings of the 16th International Software Product Line Conference*, Salvador, Brazil, pp. 215-219.

Kircher, M., Schwanninger, C., Groher, I. (2006). Transitioning to a software product family approach—Challenges and best practices. In Proceedings of the 10th International on Software Product Line Conference - SPLC ’06. IEEE Computer Society. Washington, DC, U.S.A., pp. 163–171.

Kitchenham, B. (2004). Procedures for Performing Systematic Reviews in software engineering–a systematic literature review - Technical report, TR/SE-0401. [Online] Keele: Keele University, Available at: http://www.inf.ufsc.br/~aldo.vw/kitchenham.pdf [Accessed 21 Jan. 2018].

Kitchenham, B., Brereton, O.P., Budgen, D., Turner, M., Bailey, J., and Linkman, S. (2009). Systematic literature reviews in software engineering–a systematic literature review. Information and software technology 51(1), pp. 7–15.

Klünder, J., Hohl, P., and Schneider, K. (2018). Becoming Agile While Preserving Software Product Lines: An Agile Transformation Model For Large Companies. In *Proceedings of the 2018 International Conference on Software and System Process (ICSSP ’18)*, Gothenburg, Sweden, pp. 1-10.

Kolp, M., Do, T., and Faulkner, S. (2005). Introspecting Agent-Oriented Design Patterns. Unpublished. In *S. K. Chang (Ed.), Handbook of Software Engineering and Knowledge Engineering: Recent Advances: Recent Advances,* Vol. 3, World Scientific publishing, pp. 151-177).

Krueger, C. W. (2002). Easing the transition to software mass customization. In: Proceedings of the International Workshop on Software Product-Family Engineering (PFE) Lecture Notes in Computer Science, vol. 2290. Springer, pp 282–293.

Krueger, C., and Clements, P. (2014). Second generation systems and software product line engineering. In *Proceedings of the 18th International Software Product Line Conference - Volume 1* (SPLC '14). Florence, Italy, p. 358.

Krueger, C., and Clements, P. (2017). Enterprise Feature Ontology for Feature-based Product Line Engineering and Operations. In *Proceedings of the 21st International Systems and Software Product Line Conference - Volume A* (SPLC '17), Sevilla, Spain, pp. 227-236.

Krueger, C.W. (2001). Easing the transition to software mass customization. In *Proceedings of the 4th International Workshop on Software Product-Family Engineering (PFE 2001)*, Bilbao, Spain, pp. 282–293.

Kuhrmann, M., Tiessler, M. (2014). Crafting a Method Engineering Metamodel – Approach, Methods, Results. TU München, Garching, Germany.

Kurmann R. (2006). Agile SPL–SCM Agile software product line configuration and release management. First International Workshop on Agile Product Line Engineering (in conjunction with SPLC) - (APLE ’06). Available at: http://www.lsi.upc.edu/events/aple/Kurmann.pdf [accessed 14 Oct. 2018].

Laanti M., Similä J., and Abrahamsson P. (2013) Definitions of Agile Software Development and Agility. In: McCaffery F., O’Connor R.V., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2013. Communications in Computer and Information Science, vol 364. Springer, Berlin, Heidelberg, pp. 247-258.

Ladas, C. (2009). *Scrumban - Essays on Kanban Systems for Lean Software Development*. 1st ed. Modus Cooperandi Press, Seattle, WA, USA.

Ladas, C. (2009). *Scrumban - Essays on Kanban Systems for Lean Software Development*. 1st ed. Modus Cooperandi Press, Seattle, WA, USA.

Lanman, J., Becker, B., and Samper, W. (2009). Joint Service Partnership: Extending the Live Training Transformation Product Line. In *Proceedings of Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC)*, Orlando, FL, USA, Paper No. 9083, pp. 1-11.

Laplante, P. A. (2013). *Requirements Engineering for Software and Systems*. 2nd ed., CRC Press, Boca Raton, FL, USA.

Larman, C. (2003). *Agile & Iterative Development. A Manager's Guide*. Addison-Wesley Professional, Boston, MA, USA.

Larman, C. and Basili, V. R. (2003). Iterative and Incremental Development: A Brief History. *IEEE Computer*, 36(6), pp. 47-56.

Larman, C. and Vodde, B. (2008). Scaling Lean and Agile Development. Thinking and Organizational Tools for Large-Scale Scrum. Addison-Wesley, Westford, Massachusetts, USA.

Lee, J., and Hwang, S. (2016). Variability change management using the orthogonal variability model-Based traceability. *Wireless Personal Communications*, 89 (3), pp. 729–745

Lee, S., Ko, I. , Kang, S., and Lee, D. (2010). A Usability-Pattern-Based Requirements-Analysis Method to Bridge the Gap between User Tasks and Application Features. In *2010 IEEE 34th Annual Computer Software and Applications Conference*, Seoul, pp. 317-326.

Leffingwell, D. (2007) Scaling Software Agility: Best Practices for Large Enterprises, Addison-Wesley Professional, Boston, MA, USA.

Leffingwell, D. (2011). Agile software requirements: lean requirements practices for teams, programs, and the enterprise, Addison-Wesley Professional, Boston, MA, USA..

Leffingwell, D. (2018). *SAFe 4.5 Reference Guide: Scaled Agile Framework for Lean Enterprises*. 2nd ed., Addison-Wesley Professional, Boston, MA, USA.

Leitner A., and Kreiner C. (2011) Software Product Lines – An Agile Success Factor?. In: O'Connor R.V., Pries-Heje J., Messnarz R. (eds) Systems, Software and Service Process Improvement. EuroSPI 2011. Communications in Computer and Information Science, vol 172. Springer, Berlin, Heidelberg, pp. 203-214.

Liaskos, S., Lapouchnian, A., Yu, Y., Yu, E., and Mylopoulos, J. (2006). On goal-based variability acquisition and analysis. *In Proceedings of the Requirements Engineering, 14th IEEE International Conference*, pp. 79–88, (2006).

Liskin, O., Schneider, K., Fagerholm, F., and Münch, J. (2014). Understanding the role of requirements artifacts in kanban. In Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering, Hyderabad, India, pp. 56-63.

Lucia, A. D. and Qusef, A. (2010). Requirements Engineering in Agile Software Development. *Journal of Emerging Technologies in Web Intelligence*, 2(3), pp. 212–220.

Lyytinen, K., and Rose, G.M. (2006). Information System Development Agility as Organizational Learning. *European Journal of Information Systems*, 15, pp. 183–199.

Maguire, M. (2013). Using Human Factors Standards to Support User Experience and Agile Design. In: Stephanidis C., Antona M. (eds) Universal Access in Human-Computer Interaction. Design Methods, Tools, and Interaction Techniques for eInclusion. UAHCI 2013. Lecture Notes in Computer Science, vol 8009. Springer, Berlin, Heidelberg, pp. 185-194.

Mahnic, V. (2014). Improving Software Development through Combination of Scrum and Kanban. In *Proceedings of the 8th WSEAS International Conference on Computer Engineering and Applications (CEA '14)*, WSEAS Press, Tenerife, Spain, pp. 281-288.

ManifestoAgile (2001). *Manifesto for Agile Software Development*. [online] Available at: https://agilemanifesto.org/ [Accessed 31 Mar. 2018].

Mariani, T., Colanzi, T. E., and Vergilio, S. R. (2016). Preserving architectural styles in the search based design of software product line architectures. Journal of Systems and Software, 115 (C), pp. 157-173.

Martin, J. (1991). *Rapid application development*. Macmillan Pub. Co., New York, NY, USA.

Martinez J., Diaz J., Perez J., and Garbajosa J. (2009) Software Product Line Engineering Approach for Enhancing Agile Methodologies. In: Abrahamsson P., Marchesi M., Maurer F. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2009. Lecture Notes in Business Information Processing, vol 31. Springer, Berlin, Heidelberg, pp. 247-248.

Martini, A., Pareto, L., and Bosch, J. (2013). Communication factors for speed and reuse in large-scale agile software development. In Proc. of the SPLC'13, 42. https://doi.org/10.1145/2491627.2491642

Matinlassi, M. (2004). Comparison of Software Product Line Architecture Design Methods: COPA, FAST, FORM, KobrA and QADA. In *Proceedings of the 26th International Conference on Software Engineering* (ICSE '04). IEEE Computer Society, Washington, DC, USA, pp. 127-136.

Matinlassi, M. Niemelä, E. (2002). Quality-driven architecture design method. ICSSEA´2002, *International Conference of Software and Systems Engineering and their Applications*, Paris, FR.

Matinlassi, M., Niemelä, E., and Dobrica, L. (2002). *Quality-driven architecture design and quality analysis method A revolutionary initiation approach to a product line architecture*. VTT, Technical Research Centre of Finland, Espoo, VTT PUBLICATIONS 456,.

McGregor J. D. (2008b). Agile software product lines, deconstructed. Journal of Object Technology, 7(8), pp. 7–19.

McGregor J. D. (2008c). Mix and match. Journal of Object Technology, 7(6), pp. 7–13.

McGregor, J. (2008a). Agile Product Line Engineering—A working session. SPLC '08. IEEE Computer Society. Silver Spring, MD, p. 364.

McGregor, J.D., Jarrad, S., Northrop, L.M., and Pohl, K. (2002). Initiating Software Product Lines. *IEEE Software*, 19(4), pp.24–27.

Memmel, T., Gundelsweiler, F., and Reiterer, H. (2007). Agile human-centered software engineering. In Proceedings of the 21st British HCI Group Annual Conference on People and Computers, BCS HCI, volume 1, pp. 167-175.

Messeger, S., Stapleton, J., and Hicks, A. (2014). *Agile Programme Management (AgilePgM®) Handbook*. 1st ed., DSDM Consortium, Kent, UK.

Misevičiūtė, D. (2016). *Scrum vs. Kanban vs. Scrumban – What's the difference?* [Blog] Eylean Blog. Available at: http://www.eylean.com/blog/2016/08/scrum-vs-kanban-vs-scrumban-whats-the-difference/?utm_source=blog&utm_campaign=16d35&utm_medium=blog [Accessed 05 Dec. 2018].

mm1 Consulting & Management Partnerschaftsgesellschaft, (2016). *mm1 [Poster] Scrumban* (hybrid of Scrum and Kanban). [Online] mm1.com, Available at: https://mm1.com/en/about-us/newsroom/publications/poster-scrumban/ [Accessed 13 Jan. 2019].

Mohan, K., Ramesh, B., and Sugumaran, V. (2010). Integrating Software Product Line Engineering and Agile Development. IEEE Computer Society, 27, pp. 48-55.

Moher, D., Liberati, A., Tetzlaff, J., Altman, D.G., and The PRISMA Group (2009). Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement. *PLoS Medicine* 6(7): e1000097. [Online] Public Library of Science (PLOS), Available at: https://doi.org/10.1371/journal.pmed.1000097 [Accessed 05 Feb. 2019].

Monden, Y. (1993). *Toyota production system: an integrated approach to just-in-time*. Industrial Engineering and Management Press, Norcross, Ga, USA.

Moran, A. (2014). *Agile Charting: A Practical Guide*. 1st ed., Springer-Cham, Zurich, Switzerland.

Moran, A. (2015). *Managing Agile: Strategy, Implementation, Organization and People*. 1st ed., Springer-Cham, Zurich, Switzerland.

Mouratidis, H., Kolp, M., Faulkner, S., Giorgini, P. (2005). A secure architectural description language for agent systems. In *Proceedings of the 4th international joint conference on Autonomous agents and multiagent systems*, Utrecht, Netherlands, pp. 578-585.

Muller, G. (2004). CAFCR: A Multi-view Method for Embedded Systems Architecting; Balancing Genericity and Specificity. PhD thesis, Technische Universiteit Delft.

Mywestford.com (2015). Inbound and Outbound Logistics Management Process. [Online] Available at: https://mywestford.com/blog/outbound-and-inbound-logistics-in-supply-chain-management/ [Accessed 17 May. 2019].

Nakagawa, E.Y., Oliveira Antonino, P., Becker, M. (2011). Reference architecture and product line architecture: a subtle but critical difference. In: Crnkovic, I., Gruhn, V., Book, M. *ECSA 2011*, eds. LNCS, vol. 6903, Berlin, Heidelberg: Springer-Verlag, pp. 207–211.

Näkki, P., Koskela, K., and Pikkarainen, M. (2011). Practical model for user-driven innovation in agile software development. In *Proceeding of the 17th International Conference on Concurrent Enterprising*, Aachen, pp. 1-8.

Navarrete, F., Botella, P., Franch, X. (2006). An approach to reconcile the agile and CMMI contexts in product line development. In First International Workshop on Agile Product Line Engineering - APLE ’06 - (in conjunction with SPLC), Baltimore, MD, U.S.A. Available at: http://www.lsi.upc.edu/events/aple/NavarreteBotellaFranch.pdf [accessed 16 Oct. 2017].

Nerur, S., and Balijepally, V.G. (2007). Theoretical Reflections on Agile Development Methodologies. Communications of the ACM 50(3), pp. 79-83.

Neves, G.S., & Vilain, P. (2014). Reactive Variability Realization with Test Driven Development and Refactoring. In Proceedings of the 26th International Conference on Software Engineering and Knowledge Engineering (SEKE 2014), Vancouver, Canada, pp. 100-105.

Nieke, M., Engel, G., and Seidl, C. (2017). DarwinSPL: an integrated tool suite for modeling evolving context-aware software product lines. In *Proceedings of the 11th International Workshop on Variability Modelling of Software-intensive Systems (VaMoS ’17)*, ACM, 2017, pp. 92–99,

Nikitina, N., Kajko-Mattsson, M., and Stråle, M. (2012). From scrum to scrumban: a case study of a process transition. In: *Proceedings of the International Conference on Software and System Process (ICSSP '12)*. IEEE Press, Piscataway, NJ, USA, pp. 140-149.

Noor M. A., Grünbacher P., Hoyer C. (2008b). A collaborative method for reuse potential assessment in reengineering-based product line adoption. In 2nd IFIP TC 2 Central and East European Conference on Software Engineering Techniques, CEE-SET 2007 (Lecture Notes

in Computer Science, vol. 5082/2008), Poznan, Poland. Springer, Berlin, Heidelberg, pp. 69–83.

Noor, M. A., Rabiser, R., and Grünbacher, P. (2008). Agile product line planning: A collaborative approach and a case study. *Journal of Systems and Software*, 81(6), pp. 868–882.

Noor, M. A., Rabiser, R., Grünbacher, P. (2006). A collaborative approach for reengineering-based product line scoping. In First International Workshop on Agile Product Line Engineering - APLE'06 - (in conjunction with SPLC), Baltimore, MD, U.S.A. Available at: http://www.lsi.upc.edu/events/aple/NoorRabiserGrunbacher.pdf. [Accessed 13 Oct. 2018].

Northrop, L., and Jones, L. G. (2013). Introduction to software product lines adoption. In *Proceedings of the 17th International Software Product Line Conference (SPLC '13)*. New York, NY, USA, pp. 286-286.

Northrop, L., Clements, P., Bachmann, F., Bergey, J., Chastek, G., Cohen, S., Donohoe, P., Jones, L., Krut, R., Little, R., McGregor, J., and O'Brien, L. (2012). *A Framework for Software Product Line Practice, Version 5.0*. [online] Pittsburgh: Software Engineering Institute (Carnegie Mellon University). Available at: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=495357 [Accessed 17 Apr. 2018].

Nuseibeh, B. and Easterbrook, S. (2000). Requirements Engineering: A Roadmap. In *Proceedings of the Conference on The Future of Software Engineering, ICSE '00*, Limerick, Ireland, pp. 35–46

O'Leary, P., Babar, M. A., Thiel, S., and Richardson, I. (2007b). Towards agile product derivation in software product line engineering. In Proceedings of 4th International Workshop on Rapid Integration of Software Engineering Techniques - RISE '07. Luxembourg, pp. 9–32.

O'Leary, P., McCaffery, F., Thiel, S., and Richardson, I. (2012). An agile process model for product derivation in software product line engineering. Journal of Software: Evolution and Process 24(5), pp. 561–571.

O'Leary, P., Rabiser, R., Richardson, I., and Thiel, S. (2009b). Important issues and key activities in product derivation: Experiences from two independent research projects. In Proceedings of the 13th International Software Product Line Conference - SPLC '09. Carnegie Mellon University, Pittsburgh, PA, USA, pp. 121–130.

O'Leary, P., Thiel, S., Botterweck, G., Richardson, I. (2008). Towards a product derivation process framework. In Proceedings of Third IFIP TC2 Central and East European Conference on Software Engineering Techniques - CEE-SET '08. Brno, Czech Republic, pp. 189–202.

O'Regan, G. (2013). Z Formal Specification Language, in Mathematics in Computing: An Accessible Guide to Historical, Foundational and Application Contexts. Springer, London. pp. 109-122.

Obendorf, H., and Finck, M. (2008). Scenario-based usability engineering techniques in agile development processes. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems (CHI EA '08)*. Florence, Italy, pp. 2159-2166.

O'Leary, P. , McCaffery, F. , Thiel, S. and Richardson, I. (2012), An agile process model for product derivation in software product line engineering. *Journal of Software: Evolution and Process*, 24(5), pp. 561-571.

O'Leary, P., Babar, M.A., Thiel, S., & Richardson, I. (2007a). Product derivation process and agile approaches:exploring the integration potential. In 2nd IFIP CEE Conference on SE Techniques, Poznan, Poland, pp. 166–171.

O'Leary, P., McCaffery, F., Richardson, I., and Thiel, S. (2009a) Towards Agile Product Derivation in Software Product Line Engineering. In EuroSPI Conference 2009, Alcala (Madrid), Spain.

OMG. (2017). Unified Modeling Language (OMG UML) - version 2.5.1. Technical report. [Online] Available at: https://www.omg.org/spec/UML/2.5.1/PDF [Accessed 17 Jan. 2019].

Ovaska, E. (2008). *QADA®*, [online] VTT - QADA®. Available at: http://virtual.vtt.fi/virtual/proj1/projects/qada/index.htm [Accessed 10 Aug. 2018].

Oxford University Press (OUP), and Dictionary.com (n.d.). Definition of agile in English. In *Lexico.com*, [Online] Available at: https://www.lexico.com/en/definition/agile [Accessed 17 Oct. 2018].

Ozcan-Top, O., and Demirörs, O. (2013). Assessment of Agile Maturity Models: A Multiple Case Study. In *Woronowicz T., Rout T., O'Connor R.V., Dorling A. (eds) Software Process Improvement and Capability Determination. SPICE 2013. Communications in Computer and Information Science*, vol 349. Springer, Berlin, Heidelberg, pp. 130–141.

Paetsch, F., Eberlein, A., and Maurer, F. (2003). Requirements Engineering and Agile Software Development. In Proceedings of the Twelfth International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE '03, IEEE Computer Society, p. 308.

Pahuja, S. (2018). *What is Scrumban?* [Online] Shore Labs, Available at: https://kanbantool.com/kanban-library/scrumban/what-is-scrumban [Accessed 13/12/2018].

Paige, R. F., Wang, X., Stephenson, Z. R., and Brooke, P. J. (2006). Towards an Agile Process for Building Software Product Lines. In Extreme Programming and Agile Processes in SE, pp. 198-199.

Palmer, S. R., and Felsing, J. M. (2002). *A Practical Guide to Feature-Driven Development*. 1st ed., Prentice Hall PTR, Upper Saddle River, NJ, USA.

Park, J., Moon, M., and Yeom, K. (2011). Variability modeling to develop flexible service-oriented applications. Journal of Systems Science and Systems Engineering, 20(2), pp. 193-216.

Parnas, D. L. (1976). On the Design and Development of Program Families. *IEEE Transactions on Software Engineering*, 2(1), pp. 1-9.

Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64(August 2015), pp. 1-18.

Philips Research, Embedded Systems Institute, and Vestfold University College. (1999). *The Gaudí Project*. [online] Gaudí Systems Architecting. Available at: https://www.gaudisite.nl/GaudiProject.html [Accessed 13 Sep. 2018].

Pleuss, A., Botterweck, G., Dhungana, D., Polzer, A., and Kowalewski, S. (2012). Model-driven sup- port for product line evolution on feature level. *Journal of Systems and Software*, 85 (10), pp. 2261–2274.

Pohl, K. (2010). *Requirements engineering – fundamentals, principles, and techniques*. Springer-Verlag, Berlin.

Pohl, K. and Rupp, C. (2011). Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB Compliant. 1st ed., Rocky Nook, Santa Barbara, CA, USA.

Pohl, K., Böckle, G., van der Linden, F.J. (2005). *Software Product Line Engineering: Foundations, Principles and Techniques*, Springer-Verlag, Berlin, Heidelberg.

Pohl, K., Brandenburg, M., and Gülich, A. (2001). Scenario-based Change Integration in Product Family Development. In *Proceedings of the 2nd ICSE Workshop on Software Product Lines – Economics, Architectures, and Implications*, Toronto, Canada, Fraunhofer IESE, pp. 43-48.

Poppendieck, M., Poppendieck, T. (2003). *Lean Software Development: An Agile Tookit*. Addison-Wesley, New York, NY, USA.

Porter, M. E. (1985). *The Competitive Advantage: Creating and Sustaining Superior Performance*. Free Press, New York, NY, USA.

Productlineengineering.com, (2016). *OVERVIEW*. [online] Available at: http://www.productlineengineering.com/overview/what-is-ple.html [Accessed 20 Apr. 2018].

Qumer, A., and Henderson-Sellers, B. (2008). A framework to support the evaluation, adoption and improvement of agile methods in practice. *Journal of Systems and Software*, 81(11), pp. 1899–1919.

Raatikainen, M., Rautiainen, K., Myllärniemi, V., and Männistö, T. (2008). Integrating product family modeling with development management in agile methods. In Proceedings of the 1st international workshop on Software development governance (SDG '08). ACM, New York, NY, USA, pp. 17-20.

Ramsin, R., and Paige, R.F. (2008). Process-centered review of object oriented software development methodologies. *ACM Computing Surveys*, 40(1), pp. 3:1-3:89.

Reddy, A. (2015). *What is Scrumban & Why Should I Care?* [Online] Pearson Education, Informit. Available at: http://www.informit.com/articles/article.aspx?p=2431512 [Accessed 12 Jan. 2019].

Reddy, A. (2016). *The Scrumban [r]evolution: getting the most out of Agile, Scrum, and Lean Kanban*. 1st ed. Addison-Wesley Professional, New York, NY, USA.

Reinhartz-Berger, I., Figl, K., and Haugen, Ø. (2013). Comprehending feature models expressed in CVL. In *Proceeding of the 17th International Conference Model-Driven Languages and Systems (MODELS)*, Valencia, Spain, pp. 501-517.

Rivera, J., Samper, W., and Clinger, B. (2008). Live Training Transformation (LT2) product line applied standards for reusable integrated and interoperable solutions. In *Proccedings of MILCOM 2008 - 2008 IEEE Military Communications Conference*, San Diego, CA, pp. 1-8.

Rodriguez, L. (2018). The role of transport and logistics in promoting e-commerce in developing countries. [Blog] UNCTAD Transport and Trade Facilitation Newsletter N°77 - First Quarter 2018. Available at: https://unctad.org/en/pages/newsdetails.aspx?OriginalVersionID=1705 [Accessed 17 May. 2019].

Rodriguez, P., Partanen, J., Kuvaja, P., Oivo, M. (2014). Combining lean thinking and agile methods for software development: a case study of a finnish provider of wireless embedded systems detailed. In *47th Hawaii International Conference on System Sciences (HICSS 2014).* IEEE Press, pp. 4770–4779.

Runeson, P., Host, M., Rainer, A., and Regnell, B. (2012). *Case Study Research in Software Engineering: Guidelines and Examples,* 1st ed., Wiley Publishing, Hoboken, NJ, USA.

Schloßer A., Schnitzler J., Sentis T., and Richenhagen J. (2016) Agile processes in automotive industry – Efficiency and quality in software development. In: Bargende M., Reuss HC., Wiedemann J. (eds) 16. Internationales Stuttgarter Symposium. Proceedings. Springer, Wiesbaden, pp. 489-503.

Schön, E.-M., Thomaschewski, J., and Escalona, M. J. (2017). Agile Requirements Engineering: A systematic literature review. Computer Standards & Interfaces, 49, pp. 79-91.

Schuh, P. (2004). *Integrating Agile Development in the Real World*. Charles River Media Inc., Rockland, MA, USA

Schwaber, K. (2006). Scrum is Hard and Disruptive. [online] Available at https://www.verheulconsultants.nl/ScrumIsHardandDisruptive.pdf [Accessed 27 Aug. 2018].

Schwaber, K. and Beedle, M. (2001). *Agile Software Development with Scrum*. 1st ed. Pearson, Upper Saddle River, NJ, USA.

Schwaber, K. and Sutherland, J. (2016). *The Definitive Guide to Scrum: The Rules of the Game*. [Online] Scrum Inc. Available at: http://www.scrumguides.org/ [Accessed 04 Sep. 2018].

Schweigert T., Vohwinkel D., Korsaa M., Nevalainen R., Biro M. (2013) Agile Maturity Model: A Synopsis as a First Step to Synthesis. In: McCaffery F., O'Connor R.V., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2013. Communications in Computer and Information Science, vol 364. Springer, Berlin, Heidelberg

Seidl, C., Schaefer, I., and Aßmann, U. (2014). Capturing variability in space and time with hyper feature models. In *Proceedings of the 8th International Workshop on Variability Modelling of Software-intensive Systems (VaMoS '14)*, Sophia Antipolis, France, pp.6:1–6:8.

Sepúlveda, S., Cravero, A., and Cachero, C. (2016). Requirements modeling languages for software product lines: A systematic literature review. *Information and Software Technology*, 69(1), pp. 16–36.

Shaw, M. (2002). What makes good research in software engineering? *International Journal on Software Tools for Technology Transfer (STTT)*, 4(1), pp. 1-7.

Sheldon, F., Kavi, K., Tausworthe, R., Yu, J., Brettschneider, R., and Everett, W. (1992). Reliability measurement: from theory to practice. IEEE Software, vol. 9, no. 4, pp. 13 –20.

Shen, M., Yang, W., Rong G., and Shao, D. (2012). Applying agile methods to embedded software development: A systematic review. In Proceegings of the 2nd International Workshop on Software Engineering for Embedded Systems (SEES), Zurich, pp. 30-36.

Shore Labs, (2017). Scrumban: the best of Kanban and Scrum. [Blog] Kanban Tool. Available at: http://kanbantool.com/scrumban-kanban-and-scrum [Accessed 14/11/2018].

Sidky, A. and Arthur, J. (2006). Agile adoption process framework - indicators document. [Online] arXiv:cs/0612092, Available at: https://arxiv.org/ftp/cs/papers/0612/0612092.pdf [Accessed 21 Jun. 2019].

Sidky, A. and Arthur, J. (2006). Agile adoption process framework - indicators document. [Online] arXiv:cs/0612092, Available at: https://arxiv.org/ftp/cs/papers/0612/0612092.pdf [Accessed 21 Jun. 2019].

Sidky, A., Arthur, J. and Bohner, S. (2007). A disciplined approach to adopting agile practices: the agile adoption framework. *Innovations in Systems and Software Engineering,* 3(203), pp. 203–216.

Sidky, A.S. (2007) A Structured Approach to Adopting Agile Practices: The Agile Adoption Framework. PhD., Virginia Polytechnic Institute

Sinnema M., Deelstra S., Nijhuis J., Bosch J. (2004). COVAMOF: A Framework for Modeling Variability in Software Product Families. In: Nord R.L. (eds) Software Product Lines. SPLC 2004. Lecture Notes in Computer Science, vol 3154. Springer, Berlin, Heidelberg, pp. 197-213.

Sinnema, M., and Deelstra, S. (2008). Industrial validation of COVAMOF. Journal of Systems and Software, 81(4), pp. 584–600.

Smartsheet.com (n.d.). *Scrumban: Choosing the Middle Ground Between Scrum and Kanban*. [Online] Available at: https://www.smartsheet.com/scrumban-choosing-middle-ground-between-scrum-and-kanban [Accessed 27 Jan. 2019].

Software Productivity Consortium Services Corporation (1993). *Reuse-driven software processes*. Technical report, TR SPC-92019-CMC, Version 02.00.03, Software Productivity Consortium Services Corporation.

Sommerville, I. (2010). Software Engineering. 9th ed., Addison-Wesley Professional, Boston, MA, USA.

Sommerville, I. and Sawyer, P. (1997). *Requirements engineering: a good practice guide*. John Wiley & Sons, 1st edition.

Spinellis, D. (2001). Notable design patterns for domain specific languages. *Journal of Systems and Software*, 56(1), pp. 91–99,

Spivey, J. M. (1988). *Understanding Z: A Specification Language and Its Formal Semantics*. Cambridge University Press, New York, NY, USA.

Stapleton, J. (1997). *DSDM, Dynamic Systems Development Method: The Method in Practice*. Addison-Wesley Professional, Boston, MA, USA.

Stoiber, R. (2012). *A New Approach to Product Line Engineering in Model-Based Requirements Engineering*. PhD thesis, Department of Informatics, University of Zurich.

Stoica, M., Ghilic-Micu, B., Mircea, M., & Uscatu, C. (2016). Analyzing agile development - from waterfall style to scrumban. *Informatica Economica, 20*(4), pp. 5-14.

Subramaniam, V., and Hunt, A. (2005). *Practices of an Agile Developer – Working in the Real World*. The Pragmatic Bookshelf, Dallas, TX, USA.

Suresh, A. (2018). Scrum and Kanban – How Scrumban can help? [Online] Practical Logix, Available at: https://www.practicallogix.com/scrum-and-kanban-how-scrumban-can-help/ [Accessed 09 Jan. 2019].

Sutherland, J. (2010). *Iterative vs. Incremental Development*. [Blog] Scrum Inc. Blog. Available at: https://www.scruminc.com/iterative-vs-incremental-development/ [Accessed 03 Sep. 2018].

Sutherland, J. and Schwaber, K. (2011). *The Scrum Papers: Nut, Bolts, and Origins of an Agile Framework*. [online] Available at: https://www.researchgate.net/publication/242437392_The_Scrum_Papers_Nuts_Bolts_and_Origins_of_an_Agile_Process [Accessed 04 Sep. 2018].

Taborda L. (2004). The release matrix for component-based software systems. In Proceedings Component-based Software Engineering - CBSE ’04 - (Lecture Notes in Computer Science, vol. 3054). Springer, Berlin, pp. 100–113.

Takahira, R. Y., Laraia, L. R., Dias, F. A., Yu, A. S., Nascimento, P. T. S., and Camargo, A. S. (2014). Scrum and Embedded Software development for the automotive industry. In Proceedings of PICMET '14 Conference: Portland International Center for Management of Engineering and Technology; Infrastructure and Service Integration, Kanazawa, pp. 2664-2672.

Takeuchi, H. and Nonaka, I. (1986). The new new product development game. *Harvard business review*, 64(1), pp. 137–146.

Takeuchi, H., and Nonaka, I. (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press, Oxford, UK.

Taylor, R. N., Medvidovic, N., and Dashofy, E. M. (2009). *Software Architecture: Foundations, Theory, and Practice,* John Wiley & Sons, Hoboken, NJ, USA.

TEAM, CMMI Product. (2010). CMMI® for Development, Version 1.3, Improving Processes for Developing Better Products and Services. no. CMU/SEI-2010-TR-033. Software Engineering Institute. [Online] Available at: https://resources.sei.cmu.edu/asset_files/TechnicalReport/2010_005_001_15287.pdf [Accessed 05 Apr. 2019].

Thiel S., and Hein A. (2002). Systematic Integration of Variability into Product Line Architecture Design. In *Proceedings of the Second International Conference on Software Product Lines (SPLC 2)*, Chastek, G.J., volume 2379 of Lecture Notes in Computer Science, eds. Berlin, Heidelberg: Springer-Verlag, pp. 130-153.

Tian K., and Cooper K. (2006). Agile and Software Product Line Methods: Are They So Different? [Online] Available at: http://www.cs.upc.edu/events/aple/TianCooper.pdf [accessed 16 Oct. 2017].

Tian, K. (2014). Adding More Agility to Software Product Line Methods: A Feasibility Study on Its Customization Using Agile Practices. *International Journal of Knowledge and Systems Science*, pp. 17-34.

Tourret, R. (2006). Using agile methods to develop software product lines. Master's Thesis, Computer Science—University of York, York, England.

Trinidad, P., Benavides, D., Durán, A., Ruiz-Cortés, A., Toro, M. (2008). Automated error analysis for the agilization of feature modeling. Journal of Systems and Software, 81(6), pp. 883–896.

Trinidad, P., Benavides, D., Ruiz-Cortés, A., Segura, S. (2006). Explanations for agile feature models. APLE '06: First International Workshop on Agile Product Line Engineering (in conjunction with SPLC), Baltimore, MD, U.S.A. Available at: http://www.lsi.upc.edu/events/aple/TrinidadEtAl.pdf. [Accessed 13 Oct. 2018].

Tsui, F., Karam, O. and Bernal, B. (2013). *Essentials Of Software Engineering*. 3rd ed., Jones and Bartlett Publishers, Inc., New York, NY, USA.

Turetken, O., Stojanov, I., and Trienekens, J. J. M. (2017). Assessing the adoption level of scaled agile development: a maturity model for Scaled Agile Framework. Journal of Software: Evolution and Process, 29(6), e1796.

Tzeremes, V. (2016). *End user software product line support for smart spaces*. PhD thesis, Departement of Computer Science, George Mason University.

van der Linden, F. (2002). Software product families in Europe: the Esaps & Cafe projects. *IEEE Software*, 19(4), pp. 41-49.

van der Linden, F., Schmid, K., and Rommes, E. (2007). *Software Product lines in Action: The Best Industrial Practice in Product Line Engineering*, 1est ed., Springer-Verlag, Berlin, Heidelberg.

van Lamsweerde, A. (2009). Requirements Engineering: From System Goals to UML Models to Software Specifications. 1st ed. Wiley Publishing Inc., Indianapolis, Indiana, USA.

van Lamsweerde, A., and Letier, E. (2004). From Object Orientation to Goal Orientation: A Paradigm Shift for Requirements Engineering. In: Wirsing M., Knapp A., Balsamo S. (eds) *Radical Innovations of Software and Systems Engineering in the Future*. RISSEF 2002. Lecture Notes in Computer Science, vol. 2941. Springer, Berlin, Heidelberg, pp. 325-340.

van Ommering, R. (2002). Building product populations with software components. In *Proceedings of the 24th International Conference on Software Engineering (ICSE '02)*. Orlando, FL, USA, pp. 255-265.

VDA QMC Working Group 13 / Automotive SIG. (2017). Automotive spice process assessment / reference model - Version 3.1. [Online] Available at: http://www.automotivespice.com/fileadmin/software-download/AutomotiveSPICE_PAM_31.pdf [Accessed 05 Apr. 2019].

Wanderley, F., Silva, A., Araujo, J., and Silveira, D. S. (2014). SnapMind: A framework to support consistency and validation of model-based requirements in agile development. In *2014 IEEE 4th International Model-Driven Requirements Engineering Workshop (MoDRE)*, Karlskrona, pp. 47-56.

Wang X., Zhao L., Wang Y., Sun J. (2014) The Role of Requirements Engineering Practices in Agile Development: An Empirical Study. In: Zowghi D., Jin Z. (eds) Requirements Engineering. Communications in Computer and Information Science, vol 432. Springer, Berlin, Heidelberg

Wang, X. (2005). *Towards an Agile Method for Building Software Product Lines*, M.Sc. Thesis, University of York, UK.

Waters, D. (2003). *Logistics: An Introduction to Supply Chain Management*. Palgrave Macmillan, New York, NY, USA.

Wautelet, Y., Achbany, Y., Lange, J. C., and Kolp, M. (2009). A process for developing adaptable and open service systems: Application in supply chain management. In *International Conference on Enterprise Information Systems* (pp. 564-576). Springer, Berlin, Heidelberg.

Wautelet, Y. (2012). Representing, modeling and engineering a collaborative supply chain management platform. *International Journal of Information Systems and Supply Chain Management*, 5(3), pp. 1–19.

Wautelet, Y., Heng, S., Kiv, S., Kolp, M. (2017). User-story driven development of multi-agent systems: A process fragment for agile methods. *Computer Languages, Systems & Structures,* 50, pp. 159–176.

Wautelet, Y., Heng, S., Kolp, M., and Mirbel, I. (2014). Unifying and Extending User Story Models. In: Jarke M. et al. (eds) Advanced Information Systems Engineering. CAiSE 2014. LNCS, vol 8484, Springer, Cham, pp. 211-225.

Wautelet, Y., Kolp, M., and Penserini, L. (2018). Service-Driven Iterative Software Project Management with I-Tropos. *Journal of Universal Computer Science*, 24(7), pp. 975-1011.

Weiss, D. M. and Lai, C. T. R. (1999). *Software product-line engineering: a family-based software development process*, 1st ed., Addison-Wesley, Boston, MA, USA.

Weyns, D., Michalik, B., Helleboogh, A., and Boucke, N. (2011). An architectural approach to sup- port online updates of software product lines. In *Proceedings of the 9th Working IEEE/IFIP Conference on Software Architecture (WICSA 2011)*, IEEE, pp. 204–213,

Wiegers, K. (2005). More about software requirements: thorny issues and practical advice. Microsoft Press, Redmond, Washington, D.C., USA.

Wiegers, K. and Beatty, J. (2013). Software Requirements. 3rd ed., Microsoft Press, Redmond, Washington, D.C., USA.

Wohlin, C., Höst, M., and Henningsson, K. (2003). Empirical Research Methods in Software Engineering. In: Conradi R., Wang A.I. (eds) *Empirical Methods and Studies in Software Engineering*. Lecture Notes in Computer Science, vol 2765. Springer, Berlin, Heidelberg, pp. 7-23.

Young, B., and Clements, P. (2017). Model Based Engineering and Product Line Engineering: Combining Two Powerful Approaches at Raytheon. *INCOSE International Symposium*, 27(1), pp. 518-532.

Yourdon, E. (1989). Modern Structured Analysis, volume 191. Yourdon Press Englewood Cliffs, NJ, USA.

Yu, E., Giorgini, P., Maiden, N., and Mylopoulos, J. (2011). *Social Modeling for Requirements Engineering* (eds.). MIT, Cambridge, MA.

Yu, Y., do Prado Leite, J.C.S., Lapouchnian, A., and Mylopoulos, J. (2008). Configuring features with stakeholder goals. In P*roceedings of the 2008 ACM Symposium on Applied Computing*, pp. 645–649.

Zave, P. (2003). An experiment in feature engineering. In: McIver A., Morgan C. (eds) Programming Methodology. Monographs in Computer Science. Springer, New York, NY, pp. 353-377.

Zhang, X., and Jarzabek, S. (2004). XVCL: a mechanism for handling variants in software product lines. Science of Computer Programming, 53(3), pp. 381–407.

Zhang, X., Haugen, Ø., and Moller-Pedersen, B. (2011). Model Comparison to Synthesize a Model-Driven Software Product Line. In *2011 15th International Software Product Line Conference*, Munich, pp. 90-99.

Zhou, Y. (2010). A Requirement Container Model Coupling with Goal and Feature. In *2010 International Conference on Computational Intelligence and Software Engineering*, Wuhan, pp. 1-4.

Appendix A

AgiFPL: Roles, Meetings, and Artifacts

Domain Engineering Roles

Business at domain experts

"Business Experts" play a key role in aligning business goals, prioritizing major features, and selecting business goals and market strategies Doing So, leads to the definition of the Domain Scope *Domain Experts* come from different backgrounds and they represent the users, clients, sponsors, business analysts, or any mix of these. A Domain Expert uses deep knowledge of the business to explain to the developers in various levels of details the tasks that the system must perform Business and Domain Experts represent the interests of the top management of the "Software Vendor".

Domain Sensei

"Domain Sensei" role is basically a role of supporting of the agile principles and practices for the other roles. Domain Sensei has to be present in all DE meetings that are "Planning 1","Planning 2","Daily","Review", "Retrospective", and "Andon". The Domain Sensei should has a deep knowledge about agile values and principles, as well as, the agile practices of the DE process In fact, the Domain Sensei removes obstacles of the team, guides the project members focused on the Domain Design goal According to AgiFPL, the Domain Sensei must know the value for each agile practices and activities related to DE Meetings and Production Flows.

Domain Development Team

"Domain Development Team" is a group of developers with complementary skills that perform all the defined stories and task The Domain Development Team is a cross functional group of people responsible for delivering potentially shippableincrements of reusable artifacts (at the end of every production cycle). This team perform the selection of tasks for the different product lines during the "Planning 1" and "Planning 2". The team members also pay attention to the commonalities and variabilities that have to be specified.

Domain Stakeholders

"Domain Stakeholders" are the people involved that enable the project They are only directly involved in the process during the reviews Aside from that, they may solely influence the team by discussing their needs with the product owner. Typically, the main stakeholders are external consultants, managers, customers and users related to the Domain of the Software Product Lines.

Application Engineering Roles

App i Owner

"App i Owner" is the person responsible for maintaining the "App i backlog" by representing thinterests of the stakeholders, ensuring the value of the work the development team does.

Line i Scrum Master

"Line i Scrum Master" is the person responsible for the product line i process, making sure it is used correctly and maximizing its benefits In fact, the role of the "Line i Scrum Master" is involved in all tasks that are balanced with Scrum practices. This role is a role of supporting agile principles and practices for the other roles of the "Line i".

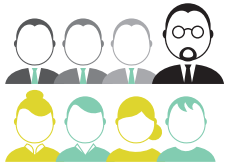
Line i Development Team

"Line i Development Team" is a cross functional group of people responsible for delivering potentially shippable increments of the App i at the end of every sprint The members of the development team have complementary skills that perform the sprint tasks.

App i Stakeholders

"App i Stakeholders" role represents the people involved that enable the project related to the "Product i". They are only directly involved in the process during the reviews Aside from that, they may solely influence the team by discussing their needs with the "App i Owner" Typically, the main stakeholders are managers, customers and users that are related to the "Product i".

Domain Engineering Meetings



Planning 1

Planning 1: The "WHAT": Whenever the product owner pulls new user stories into the selected backlog. The "Domain Analysts" holds it to select the next user stories to work on, explaining the user stories of the Features Backlog and answering open questions. After this analysis, the development team should understand the requirements Therefore, the team is able to estimate the complexity of each user story.

Planning 2

Planning 2: The "HOW" Whenever team members pull new user stories into the production flow Here, the team discusses solutions for new user stories in the SIP backlog and creates tasks for each user story accordingly.

Daily

Daily: (15 min max) A short, time boxed meeting, taking place every day at the same time Every team member answers three questions:
i. What have I done since yesterday**?**
ii. What am I planning to do today**?**
iii. What are my impediments**?**

Review

Review: (Whenever the team ships an increment) the team uses this meeting to present and review the work it has completed since the last delivery. Usually, it also includes a demonstration of the features created in the last increment.

Retrospective

Retrospective: (After any review) The "Domain Sensei" holds the retrospective to reflect on the past production cycle in order to ensure continuous process improvements. The sensei always asks two questions in the retrospective:
i. What went well during the last cycle**?**
ii. What should improve in the next cycle**?**

Andon

Andon: (Whenever a problem occurs) the "Domain Sensei" organizes an Andon meeting whenever problems in the production flow occur For example, a story is over the expected cycle time, or a task is frequently re-assigned and not yet solved Both the development Team and the product owner take place in this meeting and work on a solution to solve the open issue.

Application Engineering Meetings

Sprint Planning 1

Sprint Planning 1: (60 min per sprint week) is held to select the work to be done for the next sprint (the "WHAT"). The "App i Owner" explains the stories of the App backlog to the team and answers their question. After the planning, the team should have understood the requirements and its commits the scope for the sprint.

Sprint Planning 2

Sprint Planning 2: (60 min per sprint week) the designing phase for the selected backlog (the "HOW"). The team discusses a solution for the selected stories and creates according task for each story.

Daily

Daily: (about 15 min) short, time boxed meeting, every day at the same time Every team member answers three questions:
i. What have I done since yesterday**?**
ii. What am I planning to do today**?**
iii. What are my impediments**?**

Sprint Review

Review: (up to 60 min per sprint week) used to present and review the work that was completed and not completed during a sprint. It should include a demonstration of the realized product increment.

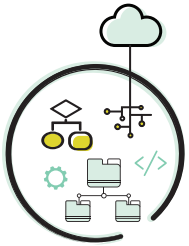
Sprint Retrospective

Sprint Retrospective: (about 45 min per sprint week) a reflection on the past sprint used to make continuous process improvements. Two main questions are asked in the sprint retrospective:
i. What went well during the sprint**?**
ii. What could be improved in the next sprint**?**

Backlog Refinement

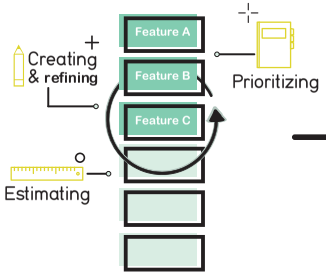
Backlog Refinement: (max 60 min) used to introduce and estimate new backlog items and to refine existing estimations as well as acceptance criteria It is also used to break large stories into smaller ones.

Domain Engineering Artifacts (1)

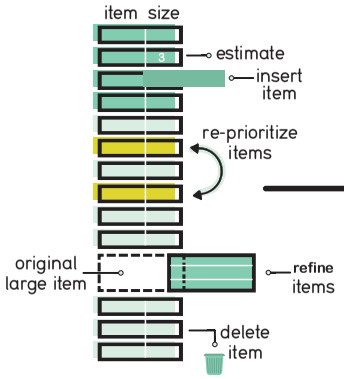


Domain Requirement Engineering (DRE): DRE aims to understand the problem space by studying the settings of the organizations related to the domain By studying a set of organizations, the emphasis is put on understanding the motivations and rationales that underline the domain requirements In fact, at this stage the reusable artifacts to be are described within its operational environment, along with relevant functions and qualities.

Domain Design (DD): At this stage, the scope of the domain and the architectures have to be determined, i.e., decide which the product line should cover products and consequently, which features are relevant and should be implemented as reusable artifacts.



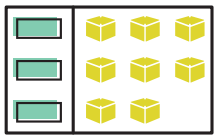
Features Backlog (FB): an ordered list of features that the team maintains for a product. At this stage, one should document requirements in 'user story' format Anyone can edit the backlog, but "Domain Analysts" are ultimately responsible for ordering the user stories Stories in the features backlog contain rough estimates of both business value and development effort.



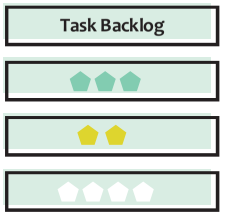
Selected backlog (SB): a list of work the development team must address next. It has a defined capacity limit. As soon as capacity is available, it is filled up with user stories or features from the top of the product backlog.

Feature: is a characteristic or end-user-visible behavior of a software system. Features are used in AgiFPL to specify and communicate commonalities and differences of the products between stakeholders, and to guide structure, reuse, and variation across all the phases of the software life cycle.

Domain Engineering Artifacts (2)



Story in Progress (SIP) Backlog: a list of user stories, which the development team currently addresses. Team members pull user stories from the selected backlog when there are no more remaining tasks in the task backlog.



Task backlog (TB): a table structured along the phases that are necessary for completing the project, e.g. design, development, and test. The development team breaks the user stories or features from the SIP backlog down into single tasks. Once a task has finished one phase, a team member from the consecutive phase eventually pulls the task to process it further.

User Story (US): a description of a certain feature or behavior, written strictly from the Domain Experts' point of view. Usually, the "Domain Experts" writes the user stories.

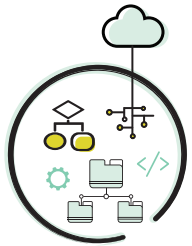
Task: a unit of work, which should be feasible within one working day or less. To implement a user story, you must accomplish all associated tasks.

Work in Progress (WIP) Limit: Limits the number of stories and tasks in each productive steps of the production flow and thus impedes work overload.

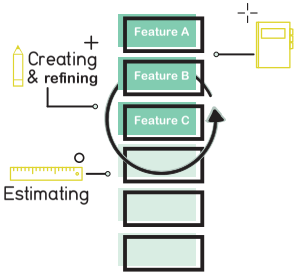
Parking lot: for tasks, which the team cannot finish due to external dependencies. For example, another team has to review a document Placing a task in the parking lot prevents the team from deadlocks, where unfinished tasks block production lines.

Impediment backlog: a list maintained by the sensei including all current impediments.

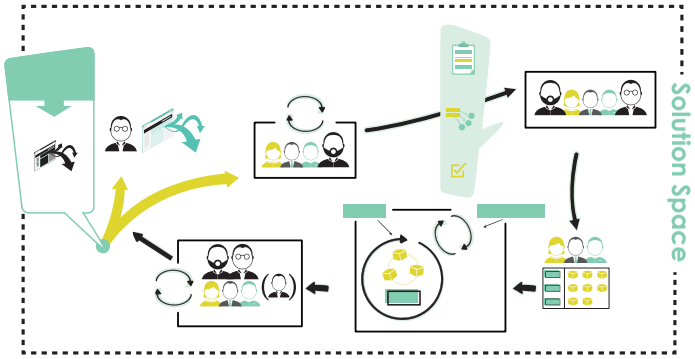
Application Engineering Artifacts (1)



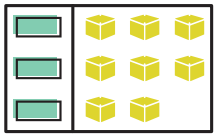
App Requirement Engineering (ARE): where the system-to-be (App to be) is described within its operational environment, along with relevant functions and qualities.



App Backlog: an ordered list of requirements that the team maintains for an application In AgiFPL approach, one should document requirements in "User Story" format. Anyone can edit the backlog, but the product owner is ultimately responsible for ordering the user stories. Stories in the App Backlog contain rough estimates of both business value and development effort.

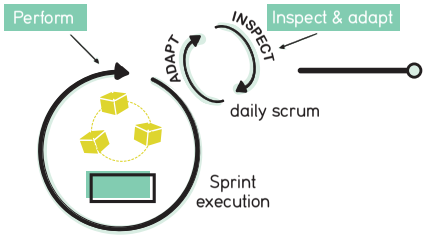


Sprints: work is performed in iterations or cycles of up to a calendar month called sprints. Sprints are time boxed so they always have a fixed start and end date, and generally, they should all be of the same duration. A new sprint immediately follows the completion of the previous sprint.



Sprint Backlog: a list of work the development team must address during the next sprint. The list is created by selecting user stories from the top of the product backlog until the development team feels it has enough work to fill the sprint, keeping in mind the velocity of its previous sprints. The stories or features are broken down into tasks by the development team. Often an accompanying task board is used to see and change the state of the tasks of the current sprint, like "To Do","In Progress" and "Done".

Application Engineering Artifacts (2)



Sprint Execution: Once the team finishes sprint planning and agrees on the content of the next sprint, the development team, guided by the Line i Scrum Master 's coaching, performs all of the task level work necessary to get the features done . "Done" means there is a high degree of confidence that all of the work necessary for producing good quality features has been completed.

User story: a description of a certain product/application feature or behavior, written strictly from the user's point of view. Usually, the "App i Owner" writes the user stories.

Task: a unit of work, which should be feasible within one working day or less. To implement a user story, you must accomplish all associated tasks.

Impediment Backlog of Line i: a list of current impediments maintained by the Line i Scrum Master.

DoD

Definition of Done (DOD): a checklist of activities required to declare the implementation of a story to be completed. The definition is determined at the beginning of the project but the team can change it at any time.

LOUVAIN SCHOOL OF MANAGEMENT
Doctoral Thesis

An Agile Feature-Driven Framework for Managing Evolving Software Product Lines

The AgiFPL Method

Hassan HAIDAR

Agile Product Line Engineering is a paradigm that has emerged as a solution for responding to the need for managing changes in requirements, reducing time-to-market, promoting product quality, and decreasing development costs in software organizations. Agile Product Line (APL) approaches are the results of combining agile methods with Software Product Lines (SPL). The main goal of this thesis is to propose an efficient APL method. In general, software development methodologies consist of two integral parts. The first one is dedicated for requirements engineering and the second one is dedicated for the development process. In this thesis, we have proposed a Feature-Oriented Agile Product Line method called "Agile Framework for managing evolving Software Product Lines – AgiFPL method". AgiFPL has been defined, designed and implemented after studying and analyzing the existent APL methodologies with the aim to take advantage of the strengths of the studied methodologies and to overcome their weaknesses. AgiFPL was proposed to address the development process part. For the requirement engineering part, we have proposed an Integrated Requirements Engineering framework for Agile Software Product Lines, which is the part that provides the syntax and semantics used for expressing the products of an APL method. The aim is to allow analysts and developers to specify requirements that precisely capture the stakeholder's needs and intentions as well as to manage product line variabilities.

Hassan HAIDAR *was born in Tyr, Lebanon. He holds a bachelor degree in Business Engineering (Université Saint-Louis – Bruxelles, 2011) and a master in Business Engineering (UCLouvain, 2013). His research primarily resides in the area of Software Engineering. His focus is mainly on Knowledge and Experience on Agile adoption, Information System Analysis and Design, Data and Information Management, Software Project Management, Business Process and Requirements Modeling and Agent-Oriented/Knowledge Systems.*

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
www.uclouvain.be