Bibliography

[AGARWAL and SINHA, 2003] AGARWAL, R.; SINHA, A. P.. **Object-oriented modeling with uml: a study of developers' perceptions**. Commun. ACM, 46(9):248–256, September 2003.

[AKSIT *et al.*,, 1991] AKSIT, M.; DIJKSTRA, J. W. ; TRIPATHI, A.. **Atomic delegation: Object-oriented transactions**. IEEE Softw., 8(2):84–92, March 1991.

[AKSIT and BERGAMANS, 1998] AKSIT, M.; BERGMANS, L.. **Examples of reusing synchronization code in aspect-oriented programming using composition-filters**. In: PROCEEDINGS OF THE MGHREBIAN CONFERENCE ON SOFTWARE ENGINEERING AND ARTIFICIAL INTELLIGENCE (MCSEAI), p. 257–272, 1998.

[ALSHAYEB and LI,2005] ALSHAYEB, M.; LI, W.. **An empirical study of system design instability metric and design evolution in an agile software process**. J. Syst. Softw., 74(3):269–274, Feb. 2005.

[ALVES *et al.,* 2006] ALVES, V.; GHEYI, R.; MASSONI, T.; KULESZA, U.; BORBA, P. ; LUCENA, C.. **Refactoring product lines**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON GENERATIVE PROGRAMMING AND COMPONENT ENGINEERING (GPCE), p. 201–210, New York, NY, USA, 2006. ACM.

[ALVES *et al.,* 2007] ALVES, V.; MATOS, P.; COLE, L.; VASCONCELOS, A.; BORBA, P. ; RAMALHO, G.. **Transactions on aspect-oriented software development iv**. chapter Extracting and evolving code in product lines with aspect-oriented programming, p. 117–142. Springer-Verlag, Berlin, Heidelberg, 2007.

[ANDA *et al.,* 2006] ANDA, B.; HANSEN, K.; GULLESEN, I. ; THORSEN, H. K.. **Experiences from introducing uml-based development method in a large organization**. Empirical Softw. Engg., 11(4):555–581, May 2006.

[APEL and BATORY, 2006] APEL, S.; BATORY, D.. **When to use features and aspects?: a case study**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON GENERATIVE PROGRAMMING AND COMPONENT ENGINEERING (GPCE), p. 59–68, New York, NY, USA, 2006. ACM.

[APEL *et al.,* 2008] APEL, S.; LEICH, T. ; SAAKE, G.. **Aspectual feature modules**. IEEE Trans. Softw. Eng., 34(2):162–180, March 2008.

[ARACIC *et al.,* 2006] ARACIC, I.; GASIUNAS, V.; MEZINI, M. ; OSTERMANN, K.. **An overview of caesarj**. In: Rashid, A.; Aksit, M., editors, TRANSACTIONS ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT I (TAOSD), volumen 3880 de **Lecture Notes in Computer Science**, p. 135–173. Springer Berlin / Heidelberg, 2006.

[ARISJOLM and SJOBERG *et al.,* 2004] ARISHOLM, E.; SJOBERG, D.. **Evaluating the effect of a delegated versus centralized control style on the maintainability of object-oriented software**. IEEE Trans. Softw. Eng., 30(8):521–534, Aug. 2004.

[ARISHOLM *et al.,* 2006] ARISHOLM, E.; BRIAND, L.; HOVE, S. E. ; LABICHE, Y.. **The impact of uml documentation on software maintenance: An experimental evaluation**. IEEE Trans. Softw. Eng., 32(6):365–381, June 2006.

[BARTOLOMEI *et al.,* 2006] BARTOLOMEI, T.; GARCIA, A.; SANT'ANNA, C. ; FIGUEIREDO, E.. **Towards a unified coupling framework for measuring aspect-oriented programs**. In: PROCEEDINGS OF THE INTERNATIONAL WORKSHOP ON SOFTWARE QUALITY ASSURANCE (SOQUA), p. 46–53, New York, NY, USA, 2006. ACM.

[BARTSCH and HARRISON, 2006] BARTSCH, M.; HARRISON, R.. **A coupling framework for aspectj**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON EVALUATION AND ASSESSMENT IN SOFTWARE ENGINEERING (EASE), p. 131–131, Swinton, UK, 2006. British Computer Society.

[BASILI *et al.,* 1999] BASILI, V. R.; SHULL, F. ; LANUBILE, F.. **Building knowledge through families of experiments**. IEEE Transactions on Software Engineering, 25:456–473, 1999.

[BERGMANS and AKSIT, 1992] BERGMANS, L.; AKSIT, M.. **Composition filters: extended expressiveness for oopls**. In: WORKSHOP

OBJECT-ORIENTED PROGRAMMING LANGUAGES: THE NEXT GENERATION AT THE PROCEEDINGS OF THE CONFERENCE ON OBJECT-ORIENTED PROGRAMMING, SYSTEMS, LANGUAGES, AND APPLICATIONS (OOPSLA), p. 341–358, 1992.

[BERTRAN *et al.*,2007] MACIA BERTRAN, I.; GARCIA, A. ; VON STAA, A.. **An exploratory study of code smells in evolving aspect-oriented systems**. In: PROCEEDINGS OF THE TENTH INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT, AOSD '11, p. 203–214, New York, NY, USA, 2011. ACM.

[BRACHA and COOK, 1990] BRACHA, G.; COOK, W.. **Mixin-based inheritance**. In: PROCEEDINGS OF THE EUROPEAN CONFERENCE ON OBJECT-ORIENTED PROGRAMMING ON OBJECT-ORIENTED PROGRAMMING SYSTEMS, LANGUAGES, AND APPLICATIONS (ECOOP), p. 303–311, New York, NY, USA, 1990. ACM.

[BRIAND *et al.*,1997] BRIAND, L.; DEVANBU, P. ; MELO, W.. **An investigation into coupling measures for c++**. In: PROCEEDINGS OF THE 19TH INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, ICSE '97, p. 412–421, New York, NY, USA, 1997. ACM.

[BRIAND *et al.*,1999] BRIAND, L. C.; WUEST, J. ; LOUNIS, H.. **Using coupling measurement for impact analysis in object-oriented systems**. In: PROCEEDINGS OF THE IEEE INTERNATIONAL CONFERENCE ON SOFTWARE MAINTENANCE, ICSM '99, p. 475–, Washington, DC, USA, 1999. IEEE Computer Society.

[BRIAND *et al.,* 1999] BRIAND, L.; DALY, J. ; WÜST, J.. **A unified framework for coupling measurement in object-oriented systems**. IEEE Trans. Softw. Eng., 25(1):91–121, January 1999.

[BRIAND, 2003] BRIAND, L. C.. **Software documentation: How much is enough?** In: PROCEEDINGS OF THE EUROPEAN CONFERENCE ON SOFTWARE MAINTENANCE AND REENGINEERING (CSMR), p. 13–15, Washington, DC, USA, 2003. IEEE Computer Society.

[BRIAND *et al.,* 2005] BRIAND, L. C.; LABICHE, Y.; PENTA, M. D. ; YAN-BONDOC, H. D.. **An experimental investigation of formality in uml-based development**. IEEE Trans. Softw. Eng., 31(10):833–849, October 2005.

[BURROWS *et al.,* 2010] BURROWS, R.; FERRARI, F.; LEMOS, O.; GARCIA, A. ; TAIANI, F.. **The impact of coupling on the fault-proneness of aspect-oriented programs: An empirical study**. In: PROCEEDINGS OF THE INTERNATIONAL SYMPOSIUM ON SOFTWARE RELIABILITY ENGINEERING (ISSRE), p. 329–338, Washington, DC, USA, 2010. IEEE Computer Society.

[BURROWS *et al.,* 2011] BURROWS, R.; TAIANDANI, F.; GARCIA, A. ; FERRARI, F.. **Reasoning about faults in aspect-oriented programs: A metrics-based evaluation**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON PROGRAM COMPREHENSION (ICPC), p. 131 –140, Washington, DC, USA, 2011. IEEE Computer Society.

[BUSCHMANN *et al.,* 1996] BUSCHMANN, F.; MEUNIER, R.; ROHNERT, H.; SOMMERLAD, P.; STAL, M. ; STAL, M.. **Pattern-Oriented Software Architecture - A System of Patterns**. John Wiley & Sons, 1 edition edition, 1996.

[CACHO *et al.*,2007] CACHO, N.; SANT'ANNA, C.; FIGUEIREDO, E.; GARCIA, A.; BATISTA, T. ; LUCENA, C.. **Composing design patterns: a scalability study of aspect-oriented programming**. In: PROCEEDINGS OF THE 5TH INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT, AOSD '06, p. 109–121, New York, NY, USA, 2006. ACM.

[CAESARJ, 2012] **Caesarj language**. http://caesarj.org/, 2012. Accessed in June.

[CAMILLERI *et al.,* 2009] CAMILLERI, A.; COULSON, G. ; BLAIR, L.. **Cif: A framework for managing integrity in aspect-oriented composition**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE OBJECTS, MODELS, COMPONENTS, PATTERNS (TOOLS), p. 18–36, 2009.

[CHAVEZ and LUCENA, 2002] CHAVEZ, C.; LUCENA, C.. **A metamodel for aspect-oriented modeling**. In: WORKSHOP ON ASPECT-ORIENTED MODELING WITH UML AT THE INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT (AOSD), 2002.

[CHAVEZ *et al.,* 2005] CHAVEZ, C.; GARCIA, A.; KULESZA, U.; SANT'ANNA, C. ; LUCENA, C.. **Taming heterogeneous aspects with crosscutting interfaces**. In: PROCEEDINGS OF THE BRAZILIAN SYMPOSIUM ON SOFTWARE ENGINEERING (SBES), p. 216–231, 2005.

[CHECKER GAME, 2013]

[CHIDAMBER and KEMERER, 1994] CHIDAMBER, S. R.; KEMERER, C. F..

[CLARKE, 2009] CLARKE, S.. **Composition of Object-Oriented Software Design Models**. PhD thesis, Dublin City University, January 2001.

[CLEMENTS and NORTHROP, 2001] CLEMENTS, P.; NORTHROP, L..

[COMPOSE PROJECT, 2012] **Compose* project**. http://composestar.sourceforge.net/, 2012. Accessed in June.

[CZARNECKI and HELSEN, 2006] CZARNECKI, K.; HELSEN, S.. **Feature-based survey of model transformation approaches**. IBM Syst. J., 45(3):621–645, July 2006.

[DANTAS and GARCIA, 2010] DANTAS, F.; GARCIA, A.. **Stability of product lines with composition filters: An exploratory study**. In: PROCEEDINGS OF THE EMPIRICAL EVALUATION OF SOFTWARE COMPOSITION TECHNIQUES (ESCOT), p. 100–108. Springer-Verlag, 2010.

[DAVID, 2000] SHESKIN, D. J.. **Handbook of Parametric and Nonparametric Statistical Procedures**. O'Reilly & Associates, second edition edition, 2000.

[DAVID, 2000] DIMITRIS KOLOVOS, LOUIS ROSE, A. G.-D. R. P.. **The Epsilon Book**. First edition edition, 2011.

[DI PENTA *et al.,* 2007] PENTA, M. D.; STIREWALT, R. E. ; KRAEMER, E.. **Designing your next empirical study on program comprehension**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON PROGRAM COMPREHENSION (ICPC), p. 281–285, Washington, DC, USA, 2007. IEEE Computer Society.

[DUCASSE *et al.,* 2006] DUCASSE, S.; NIERSTRASZ, O.; SCHÄRLI, N.; WUYTS, R. ; BLACK, A. P.. **Traits: A mechanism for fine-grained reuse**. ACM Trans. Program. Lang. Syst., 28(2):331–388, Mar. 2006.

[DZIDEK *et al.,* 2008] DZIDEK, W.; ARISHOLM, E. ; BRIAND, L. C.. **A realistic empirical evaluation of the costs and benefits of uml in software maintenance**. IEEE Trans. Softw. Eng., 34(3):407–432, May 2008.

[ELISH and RINE,2003] ELISH, M. O.; RINE, D.. **Investigation of metrics for object-oriented design logical stability**. In: PROCEEDINGS OF

THE SEVENTH EUROPEAN CONFERENCE ON SOFTWARE MAINTENANCE AND REENGINEERING, CSMR '03, p. 193–, Washington, DC, USA, 2003.

[ENDRIKAT and HANENBERG, 2011] ENDRIKAT, S.; HANENBERG, S.. **Is aspect-oriented programming a rewarding investment into future code changes? a socio-technical study on development and maintenance time**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON PROGRAM COMPREHENSION (ICPC), p. 51–60, Washington, DC, USA, 2011. IEEE Computer Society.

[ETZKORMN and DELUGACH, 2000] ETZKORN, L.; DELUGACH, H.. **Towards a semantic metrics suite for object-oriented design**. In: PROCEEDINGS OF THE TECHNOLOGY OF OBJECT-ORIENTED LANGUAGES AND SYSTEMS (TOOLS), p. 71–, Washington, DC, USA, 2000. IEEE Computer Society.

[FARIAS *et al.,* 2012] FARIAS, K.; GARCIA, A. ; LUCENA, C.. **Evaluating the impact of aspects on inconsistency detection effort: a controlled experiment**. In: PROCEEDINGS OF THE 15TH INTERNATIONAL CONFERENCE ON MODEL DRIVEN ENGINEERING LANGUAGES AND SYSTEMS (MODELS), p. 219–234, Berlin, Heidelberg, 2012. Springer-Verlag.

[FAYAD,2002] FAYAD, M.. **Accomplishing software stability**. Commun. ACM, 45(1):111–115, Jan. 2002.

[FERRARI *et al.,* 2010] FERRARI, F.; BURROWS, R.; LEMOS, O.; GARCIA, A.; FIGUEIREDO, E.; CACHO, N.; LOPES, F.; TEMUDO, N.; SILVA, L.; SOARES, S.; RASHID, A.; MASIERO, P.; BATISTA, T. ; MALDONADO, J.. **An exploratory study of fault-proneness in evolving aspect-oriented programs**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE) - VOLUME 1, p. 65–74, New York, NY, USA, 2010. ACM.

[FIGUEIREDO *et al.,* 2008a] FIGUEIREDO, E.; CACHO, N.; SANT'ANNA, C.; MONTEIRO, M.; KULESZA, U.; GARCIA, A.; SOARES, S.; FERRARI, F.; KHAN, S.; FILHO, F. C. ; DANTAS, F.. **Evolving software product lines with aspects: an empirical study on design stability**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE), p. 261–270, New York, NY, USA, 2008. ACM.

[FIGUEIREDO *et al.,* 2009] FIGUEIREDO, E.; SILVA, B.; SANT'ANNA, C.; GARCIA, A.; WHITTLE, J. ; NUNES, D.. **Crosscutting patterns and design stability: An exploratory analysis**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON PROGRAM COMPREHENSION (ICPC), p. 138–147, May 2009.

[FILHO *et al.*,2006] FILHO, F. C.; CACHO, N.; FIGUEIREDO, E.; MARANHÃO, R.; GARCIA, A. ; RUBIRA, C. M. F.. **Exceptions and aspects: the devil is in the details**. In: PROCEEDINGS OF THE 14TH ACM SIGSOFT INTERNATIONAL SYMPOSIUM ON FOUNDATIONS OF SOFTWARE ENGINEERING, SIGSOFT '06/FSE-14, p. 152–162, New York, NY, USA, 2006. ACM.

[GARCIA *et al.,* 2005] GARCIA, A.; SANT'ANNA, C.; FIGUEIREDO, E.; KULESZA, U.; LUCENA, C. ; VON STAA, A.. **Modularizing design patterns with aspects: a quantitative study**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT (AOSD), p. 3–14, New York, NY, USA, 2005. ACM.

[GENERO *et al.*,2008] GENERO, M.; CRUZ-LEMUS, J. A.; CAIVANO, D.; ABRAHÃO, S.; INSFRAN, E. ; CARSÍ, J. A.. **Assessing the influence of stereotypes on the comprehension of uml sequence diagrams: A controlled experiment**. In: PROCEEDINGS OF THE 11TH INTERNATIONAL CONFERENCE ON MODEL DRIVEN ENGINEERING LANGUAGES AND SYSTEMS, MoDELS '08, p. 280–294, Berlin, Heidelberg, 2008. Springer-Verlag.

[GREENWOOD *et al.,* 2007] GREENWOOD, P.; BARTOLOMEI, T.; FIGUEIREDO, E.; GARCIA, A.; CACHO, N.; SANTÏ¿½ANNA, C.; BORBA, P.; KULESZA, U. ; RASHID, A.. **On the impact of aspectual decompositions on design stability: An empirical study**. In: PROCEEDINGS OF EUROPEAN CONFERENCE ON OBJECT-ORIENTED PROGRAMMING (ECOOP), LNCS, p. 176–200. Springer-Verlag, 2007.

[GRISWOLD *et al.,* 2006] GRISWOLD, W.; SULLIVAN, K.; SONG, Y.; SHONLE, M.; TEWARI, N.; CAI, Y. ; RAJAN, H.. **Modular software design with crosscutting interfaces**. IEEE Softw., 23(1):51–60, Jan. 2006.

[GURGEL *et al.,* 2010] GURGEL, A.; DANTAS, F. ; GARCIA, A.. **Um estudo de composições de padrões de projeto em caesarj**. In: IV LATIN

AMERICAN WORKSHOP ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT, Salvador, July 2010. SBC.

[HAIHAO *et al.,* 2008] SHEN, H.; ZHANG, S. ; ZHAO, J.. **An empirical study of maintainability in aspect-oriented system evolution using coupling metrics**. In: PROCEEDINGS OF THE INTERNATIONAL SYMPOSIUM ON THEORETICAL ASPECTS OF SOFTWARE ENGINEERING (TASE), p. 233–236, Washington, DC, USA, 2008. IEEE Computer Society.

[HASSOUN and CONSTANTINIDES, 2003] HASSOUN, Y.; CONSTANTINIDES, C. A.. **The development of generic definitions of hyperslice packages in hyper/j**. Electronic Notes in Theoretical Computer Science, 82(5):8–20, 2003.

[HENDERSON-SELLERS, 1995] HENDERSON-SELLERS, B.. **Object-oriented Metrics: Measures of Complexity**. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[HOFFMAN and EUGSTER, 2008] HOFFMAN, K.; EUGSTER, P.. **Towards reusable components with aspects: an empirical study on modularity and obliviousness**. In: PROCEEDINGS OF THE 30TH INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE), p. 91–100, New York, NY, USA, 2008. ACM.

[HOHEENSTEIN, 2006] HOHEENSTEIN, U.. **Improving the performance of database applications with aspect-oriented programming**. In: PROCEEDINGS OF 5TH INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPEMENT, Germmany, March 2006. ACM.

[HOLT *et al.,* 1987] HOLT, R.; BOEHM-DAVIS, D. ; SHULTZ, A.. **Empirical studies of programmers: second workshop**. chapter Mental representations of programs for student and professional programmers, p. 33–46. Ablex Publishing Corp., Norwood, NJ, USA, 1987.

[HÖst *et al.,* 2000] HÖST, M.; REGNELL, B. ; WOHLIN, C.. **Using students as subjects : A comparative study ofstudents and professionals in lead-time impact assessment**. Empirical Softw. Engg., 5(3):201–214, Nov. 2000.

[IBM RSA, 2011] IBM. http://www.ibm.com/developerworks/rational/products/rsa/, 2011.

[JBOSS, 2013] REDHAT. **Jboss community**. http://www.jboss.org/, 2013.

[JHESS GAME, 2013] GAME, J.. **Jhess game**. http://sourceforge.net/projects/jhess/, 2009.

[KASTNER *et al.,* 2007] KASTNER, C.; APEL, S. ; BATORY, D.. **A case study implementing features using aspectj**. In: PROCEEDINGS OF THE INTERNATIONAL SOFTWARE PRODUCT LINE CONFERENCE (SPLC), p. 223–232, Washington, DC, USA, 2007. IEEE Computer Society.

[KATZ and KATZ, 2009] KATZ, E.; KATZ, S.. **Modular verification of strongly invasive aspects: summary**. In: PROCEEDINGS OF THE WORKSHOP ON FOUNDATIONS OF ASPECT-ORIENTED LANGUAGES (FOAL), p. 7–12, New York, NY, USA, 2009. ACM.

[KELLY, 2006] KELLY, D.. **A study of design characteristics in evolving software using stability as a criterion**. IEEE Trans. Softw. Eng., 32(5):315–329, May 2006.

[KICZALES *et al.,* 1997] KICZALES, G.; LAMPING, J.; MENHDHEKAR, A.; MAEDA, C.; LOPES, C.; LOINGTIER, J.-M. ; IRWIN, J.. **Aspect-oriented programming**. In: PROCEEDINGS EUROPEAN CONFERENCE ON OBJECT-ORIENTED PROGRAMMING, volumen 1241, p. 220–242, Berlin, Heidelberg, and New York, June 1997. Springer-Verlag.

[KICZALES *et al.,* 2001] KICZALES, G.; HILSDALE, E.; HUGUNIN, J.; KERSTEN, M.; PALM, J. ; GRISWOLD, W.. **An overview of aspectj**. In: PROCEEDINGS OF THE EUROPEAN CONFERENCE ON OBJECT-ORIENTED PROGRAMMING (ECOOP), p. 327–353, London, UK, 2001. Springer-Verlag.

[KICZALES and MEZINI, 2006] **Aspect-oriented programming and modular reasoning**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE), p. 49–58, New York, NY, USA, 2005. ACM.

[KOMPOSE, 2011] **A generic model composition tool**. http://www.kermeta.org/kompose, 2011.

[KRUEGER, 1992] KRUEGER, C. W.. **Software reuse**. ACM Comput. Surv., 24(2):131–183, June 1992.

[KULESHOV, 2007] KULESHOV, E.. **Using asm framework to implement common bytecode transformation patterns**. In: PROCEEDINGS OF 6TH INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPEMENT, Vancouver, March 2007. ACM.

[KUMAR *et al.,* 2009] KUMAR, A.; KUMAR, R. ; GROVER, P. S.. **Generalized coupling measure for aspect-oriented systems**. SIGSOFT Softw. Eng. Notes, 34(3):1–6, May 2009.

[KUZNIARZ *et al.,* 2004] KUZNIARZ, L.; STARON, M. ; WOHLIN, C.. **An empirical study on using stereotypes to improve understanding of uml models**. In: PROCEEDINGS OF THE IEEE INTERNATIONAL WORKSHOP ON PROGRAM COMPREHENSION (IWPC), p. 14–23, Washington, DC, USA, 2004. IEEE Computer Society.

[MACIA *et al.,* 2011] MACIA BERTRAN, I.; GARCIA, A. ; VON STAA, A.. **An exploratory study of code smells in evolving aspect-oriented systems**. In: PROCEEDINGS OF THE TENTH INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT (AOSD), p. 203–214, New York, NY, USA, 2011. ACM.

[MADSEN *et al.*,1989] MADSEN *ET AL.*, .. **Virtual classes: a powerful mechanism in object-oriented programming**. In: CONFERENCE PROCEEDINGS ON OBJECT-ORIENTED PROGRAMMING SYSTEMS, LANGUAGES AND APPLICATIONS, OOPSLA '89, p. 397–406, New York, NY, USA, 1989. ACM.

[MEZINI and OSTERMANN, 2002] MEZINI, M.; OSTERMANN, K.. **Integrating independent components with on-demand remodularization**. SIGPLAN Not., 37(11):52–67, November 2002.

[MEZINI and OSTERMANN, 2003] MEZINI, M.; OSTERMANN, K.. **Conquering aspects with caesar**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT (AOSD), p. 90–99, New York, NY, USA, 2003. ACM.

[MEZINI and OSTERMANN, 2004] MEZINI, M.; OSTERMANN, K.. **Variability management with feature-oriented programming and aspects**. SIGSOFT Softw. Eng. Notes, 29(6):127–136, October 2004.

[MOHAGHEGHI *et al.,* 2004] MOHAGHEGHI, P.; CONRADI, R.; KILLI, O. M. ; SCHWARZ, H.. **An empirical study of software reuse vs. defect-density and stability**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE), p. 282–292, Washington, DC, USA, 2004. IEEE Computer Society.

[NARAYANAN *et al.,* 2006] SRINIVAS NARAYANAN, KAMAL GOVINDRAJ, B. T.-P. N.. **Use of aop in j2ee application performance monitor-**

ing. In: PROCEEDINGS OF 5TH INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPEMENT, Germmany, March 2006. ACM.

[NIERSTRASZ and MEIJLER, 1995] NIERSTRASZ, O.; MEIJLER, T. D.. **Research directions in software composition**. ACM Comput. Surv., 27(2):262–264, June 1995.

[PIVETA *et al.*,2006] PIVETA, E. K.; HECHT, M.; PIMENTA, M. S. ; PRICE, R. T.. **Detecting bad smells in aspectj**. Journal of Universal Computer Science, 2006.

[PREHOFER, 1997] PREHOFER, C.. **Feature-oriented programming: A fresh look at objects**. In: PROCEEDINGS OF THE EUROPEAN CONFERENCE ON OBJECT-ORIENTED PROGRAMMING (ECOOP), volumen 1241 de **Lecture Notes in Computer Science**, p. 419–443. Springer-Verlag, 1997.

[R-ENVIRONMENT, 2012] **R language and environment**. http://www.r-project.org/, 2012. Accessed in June.

[RAJAN and KEVIN, 2005] RAJAN, H.; SULLIVAN, K.. **Classpects: unifying aspect- and object-oriented language design**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE), p. 59–68, New York, NY, USA, 2005. ACM.

[ROO *et al.*, 2008] DE, A. R.; HENDRIKS, M.; HAVINGA, W.; DURR, P. ; BERGMANS, L.. **Compose*: a language- and platform-independent aspect compiler for composition filters**. In: FIRST INTERNATIONAL WORKSHOP ON ADVANCED SOFTWARE DEVELOPMENT TOOLS AND TECHNIQUES, WASDETT 2008, Cyprus, July 2008. No publisher.

[ROO *et al.*, 2008] DE ROO, A. J.; HENDRIKS, M. F. H.; HAVINGA, W. K.; DURR, P. E. A. ; BERGMANS, L. M. J.. **Compose*: a language- and platform-independent aspect compiler for composition filters**. In: FIRST INTERNATIONAL WORKSHOP ON ADVANCED SOFTWARE DEVELOPMENT TOOLS AND TECHNIQUES (WASDETT), Cyprus, July 2008.

[SANTANNA *et al.*, 2003] SANT'ANNA, C.; GARCIA, A.; CHAVEZ, C.; LUCENA, C. ; VON STAA, A.. **On the reuse and maintenance of**

aspect-oriented software: An assessment framework. In: PROCEEDINGS OF THE BRAZILIAN SYMPOSIUM ON SOFTWARE ENGINEERING (SBES), p. 19–34, 2003.

[SANTANNA *et al.,* 2004] SANT'ANNA, C.; GARCIA, A.; KULESZA, U.; LUCENA, C. ; VON STAA, A.. **Design patterns as aspects: A quantitative assessment**. Journal of the Brazilian Computer Society, 10:42–55, 11 2004.

[SCHAEFER *et al.,* 2010] SCHAEFER, I.; BETTINI, L.; DAMIANI, F. ; TANZARELLA, N.. **Delta-oriented programming of software product lines**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON SOFTWARE PRODUCT LINES (SPLC): GOING BEYOND, p. 77–91, Berlin, Heidelberg, 2010. Springer-Verlag.

[SHEO *et al.*,2008] SHEN, H.; ZHANG, S. ; ZHAO, J.. **An empirical study of maintainability in aspect-oriented system evolution using coupling metrics**. In: PROCEEDINGS OF THE 2008 2ND IFIP/IEEE INTERNATIONAL SYMPOSIUM ON THEORETICAL ASPECTS OF SOFTWARE ENGINEERING, TASE '08, p. 233–236, Washington, DC, USA, 2008. IEEE Computer Society.

[SHOGI GAME, 2013] GAME, S.. **Shogi game**. http://sourceforge.net/projects/simpleshogi/, 2010.

[SPRING, 2013] SPRINGSOURCE. **Spring source community**. www.springsource.org/, 2013.

[SRIVISUT *et al.*,2007] SRIVISUT, K.; MUENCHAISRI, P.. **Bad-smell metrics for aspect-oriented software**. In: COMPUTER AND INFORMATION SCIENCE, 2007. ICIS 2007. 6TH IEEE/ACIS INTERNATIONAL CONFERENCE ON, p. 1060 –1065, july 2007.

[TAUBE-SCHOCK *et al.*,2011] TAUBE-SCHOCK, C.; WALKER, R. J. ; WITTEN, I. H.. **Can we avoid high coupling?** In: PROCEEDINGS OF THE 25TH EUROPEAN CONFERENCE ON OBJECT-ORIENTED PROGRAMMING, ECOOP'11, p. 204–228, Berlin, Heidelberg, 2011. Springer-Verlag.

[TOURWE *et al.,* 2003] TOURWE, T.; BRICHAU, J. ; GYBELS, K.. **On the existence of the aosd-evolution paradox**. In: PROCEEDINGS OF THE SOFTWARE ENGINEERING PROPERTIES OF LANGUAGES FOR ASPECT TECHNOLOGIES (SPLAT), 2003.

[WHITTLE *et al.*, 2009]

[WHITEHEAD,2007] WHITEHEAD, J.. **Collaboration in software engineering: A roadmap**. In: 2007 FUTURE OF SOFTWARE ENGINEERING, FOSE '07, p. 214–225, Washington, DC, USA, 2007. IEEE Computer Society.

[WOHLIN *et al.,* 2000] WOHLIN, C.; RUNESON, P.; HÖST, M.; OHLSSON, M. C.; REGNELL, B. ; WESSLÉN, A.. **Experimentation in software engineering: an introduction**. Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[YAU *et al.*,1980] YAU, S. S.; COLLOFELLO, J. S.. **Some stability measures for software maintenance**. IEEE Trans. Softw. Eng., 6(6):545–552, Nov. 1980.

[YAU and COLLOFELLO, 1985] YAU, S. S.; COLLOFELLO, J. S.. **Design stability measures for software maintenance**. IEEE Trans. Softw. Eng., 11(September):849–856, Sept. 1985.

[YOUNG,2005] YOUNG, T. J.. **Using aspectj to build a software product line for mobile devices. msc dissertation**. In: UNIVERSITY OF BRITISH COLUMBIA, DEPARTMENT OF COMPUTER SCIENCE, p. 1–6, 2005.

[ZHAO, 2004] ZHAO, J.. **Measuring coupling in aspect-oriented systems**. In: INFORMATION PROCESSING SOCIETY OF JAPAN (IPSJ), p. 14–16, 2004.

A

AppendixA - Target Applications

This Appendix presents the three evolving software applications used in this thesis, which are MobileMedia (Section A.1), iBatis (Section A.2) and GameUp (Section A.3).

A.1

MobileMedia

It is a program family that provides support to manage (create, delete, visualize, play, send) different types of media (photo, music and video) on mobile devices. During the SPL development and evolution, the initial core architecture was systematically enriched with mandatory, optional and alternative features. Seven releases of the MobileMedia were analyzed. The core features are: create/delete media (photo, music or video), label media, and view/play media. Some varying features, amongst others, are: transfer photo via SMS, count and sort media, copy media and set favorites. We decided to re-implement with advanced mechanisms all the variabilities, whenever it made sense, those ones that are implemented with aspects in the original AspectJ implementation. Table A.1 summarizes the

Table A.1: Mobile Media Details

System Type - Size	Software Product Line - 5 KLOC
Programming Language	Java (OO), AspectJ (AOP) and CaesarJ (FOP)
Number of versions	07
Number of selected versions	07
Avg. # of Modules	2100
Avg. # of Program Elements	13250
Avg. # of Compositions	18310

A.2

iBatis

iBatis is a Java-based open source framework for data mapping and it uses two main APIs: (i) SQL Maps for reducing JDBC code; and (ii) Data Access Objects (DAO) for abstracting the persistence implementation details. It is composed of more than 60 releases developed incrementally and its development is characterized as a reactive approach. Initially, four releases were chosen and implemented using the AspectJ and CaesarJ language in its essential. The functionalities that were refactored were: concurrency, type mapping, design patterns, error context, exception handling, connection, session, and transaction.

Table A.2: iBatis Details

System Type - Size	Framework - 110 KLOC
Programming Language	Java (OO), AspectJ (AOP) and CaesarJ (FOP)
Number of versions	60
Number of selected versions	04
Avg. # of Modules	6010
Avg. # of Program Elements	18920
Avg. # of Compositions	22301

A.3

GameUP

GameUp is a SPL developed following the reactive approach. It encompasses three open-source board games where each of them is an SPL: Shogi , JHess and Checkers . Checkers is an American checker whereas Shogi and JHess are chess games. All of them provide features to manage various functionalities for customizing the board (e.g. indicating moveable pieces) and the matches between players (e.g. indicating player turns). New feature combinations make possible to generate several additional products, which could not be derived from the individual SPLs. In fact, the SPLs involve more than thirty five features that embrace several categories of features . The SPL variabilities provide an extensive list of integration scenarios.

Table A.3: GameUP Details

System Type - Size	Software Product Line - 3 KLOC
Programming Language	Java (OO), AspectJ (AOP) and CaesarJ (FOP)
Number of versions	06
Number of selected versions	06
Avg. # of Modules	308
Avg. # of Program Elements	1963
Avg. # of Compositions	7250

B

AppendixB - Questionnaire of the Experiment

This appendix presents the questionnaire given to the participants involved in the experiment. Basically, the participants had to determine their level of knowledge and experience in developing and evolving software systems.

- **Date of birth**:dd — mm — aaaa
- **Gender**: () Male () Female

1. Which design(s) technologies are you familiar with?

   a. Aspect-oriented (AO) design
   b. Feature-oriented (FO) design
   c. None of the above

   If you selected option "c", which modelling languages have you already used?

   _____

   _____

2. Which programming language(s) are you familiar with?

   a. AspectJ
   b. AspectJ dialects such as Spring and Jboss AOP
   c. CaesarJ
   d. None of the above

   If you selected option "d", which programming languages have you already worked with?

   _____

   _____

3. If you are familiar with AO and FO designs, please tick which of the following activities you have performed.

a. I took part in discussions about software design
b. I took part in discussions about software design and their evolution
c. I have designed software systems
d. I have designed and evolved software systems
e. I have analysed quality attributes (e.g. reusability and reusability) of software design

4. If you are familiar with design technologies (AOP and FOP), how long have you worked with them?

a. 0 to 6 months
b. 6 months to 1 year
c. 1 to 3 years
d. More than 3 years

5. If you are familiar with programming languages (Question 2), how long have you worked with them?

a. 0 to 6 months
b. 6 months to 1 year
c. 1 to 3 years
d. More than 3 years

6. How do you classify your level of experience in the following topics? It can be "Advanced" (more than 3 years of experience), "Intermediate" (1 to 3 years of experience), "Basic" (6 months to 1 year) or "No experience" (0 to 6 months)

6.1 Feature-Oriented Software Development
a. Advanced
b. Intermediate
c. Basic
d. No experience

6.2 Aspect-Oriented Software Development
a. Advanced
b. Intermediate
c. Basic
d. No experience

6.3 Software Evolution
a. Advanced
b. Intermediate

c. Basic
d. No experience

6.4 UML-based Design
a. Advanced
b. Intermediate
c. Basic
d. No experience

ADDITIONAL COMMENTS:

_____

_____

C

AppendixC - Experiment Details

This appendix presents the description of the program modification tasks of the experiment (Section C.1). All the tasks must be implemented with the AspectJ programming language. The participants had to answer all the tasks within 60 minutes. This experiment is based on elements of a family of board games, called GameUP (Section A). All these games provide functionalities for customizing the board (e.g. indicating moveable pieces) and the matches between players (e.g. indicating player turns). In addition, Section C.2 presents the feedback questionnaire provided by the experiment participants.

C.1

Description of Tasks

1. The GameUP developers must evolve the games with a new functionality, which consists of displaying a new screen before the match starts, where the player can specify whether the game will be played on the network or not. This new functionality must be implemented through a method named `setupGame( )` using AspectJ mechanisms. The existing method `startGame()` is in charge of initializing the game. The level of detail expected in your answer is the same as illustrated in the aspect `BoardStartup`, `GameStatus`, `ControllerBase` and `NickNameDefinition`.

Code C.1: Code of Task 1

```
class MainFrame {
   int status;
   GameController controller;
   void startGame(){
      ...
      Controller.setStatus(status);
   }
   void stopGame(){ //stop the game  }
}
-----------------------------------------
aspect BoardStartup{
   pointcut setGame(): call (void *.startGame());
   void setBoardGame( ) {
    //make the board available to the match
   }
   before( ): setGame(){ setBoardGame( );}
}
-----------------------------------------
class GameController {
   int status;
   void setStatus(int x){ this.status = x;}

}
-----------------------------------------
abstract aspect ControllerBase {
   pointcut setData(): call(* *.set*());

}
-----------------------------------------
aspect GameStatus extends ControllerBase{
   after( ): setData( ){ //Print a message }
}
-----------------------------------------
aspect NickNameDefinition {
   pointcut setId(): call (void *.startGame());
   before( ): setId( ){ //display screen to specify nickname
      }
   ...
}
-----------------------------------------
aspect ExecutionOrder {
   declare precedence: BoardStartup, NickNameDefinition
}
```

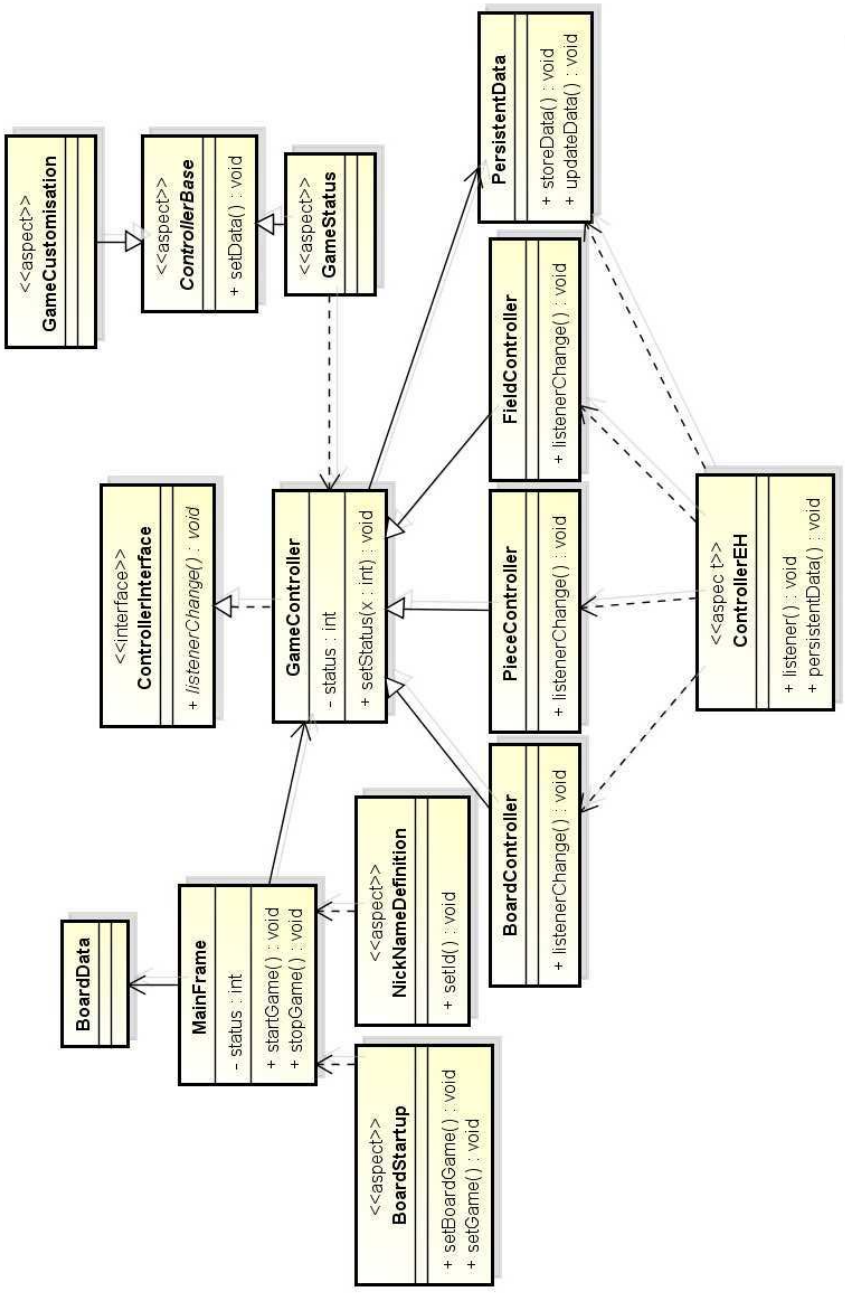

Figure C.1: UML Design for Task 1 - Class Diagram

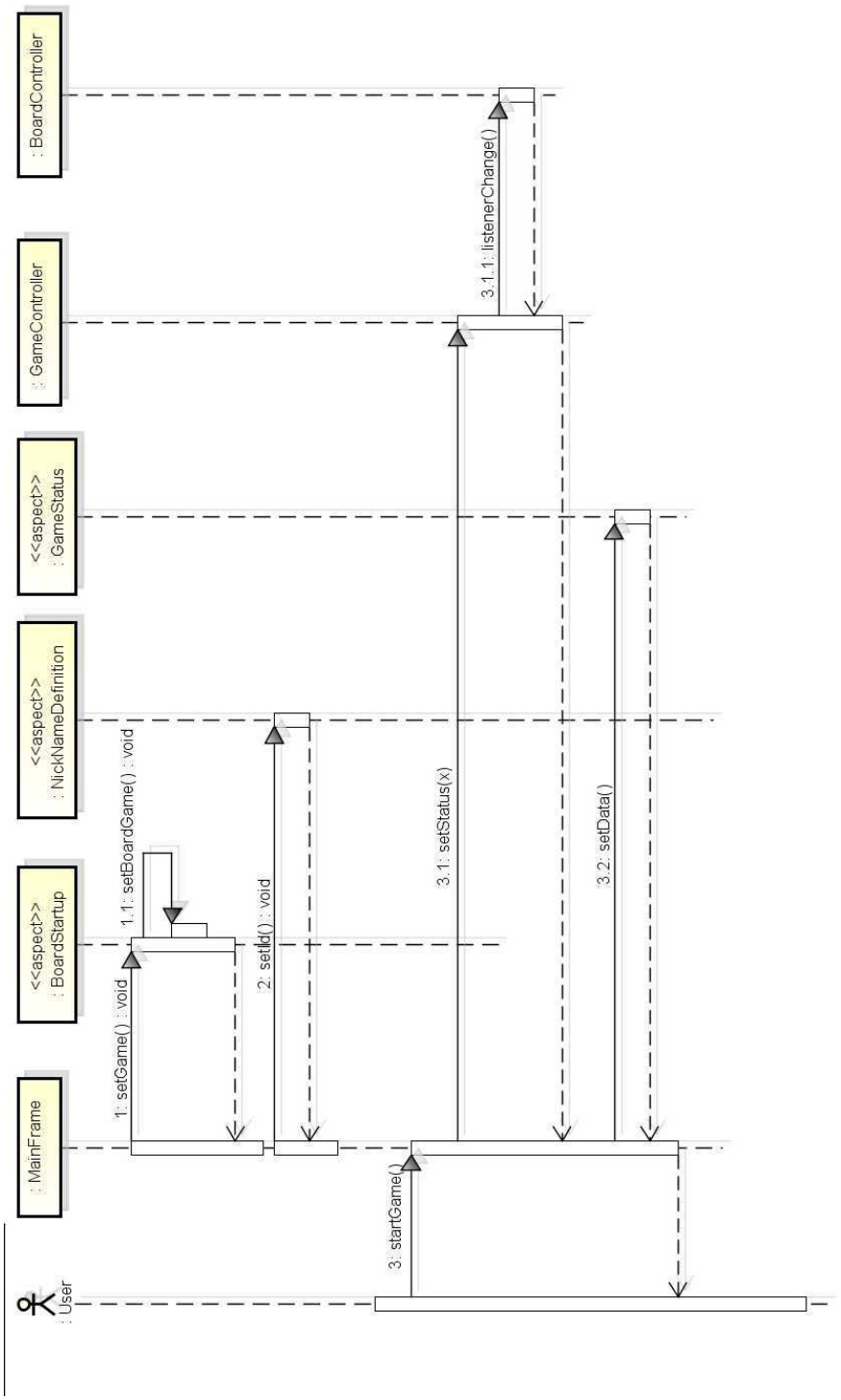

Figure C.2: UML Design for Task 1 - Sequence Diagram

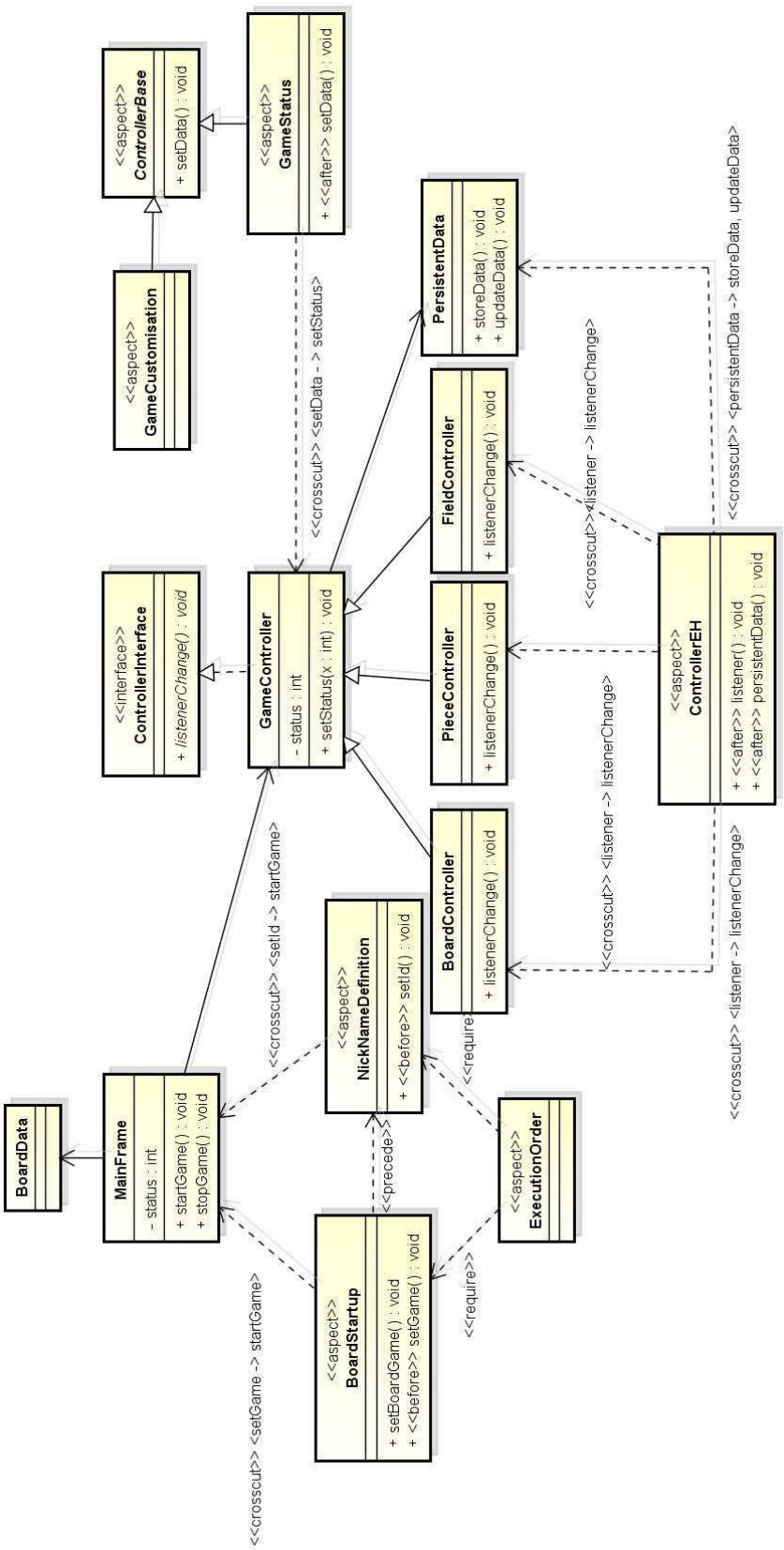

Figure C.3: UML+ Design for Task 1 - Class Diagram

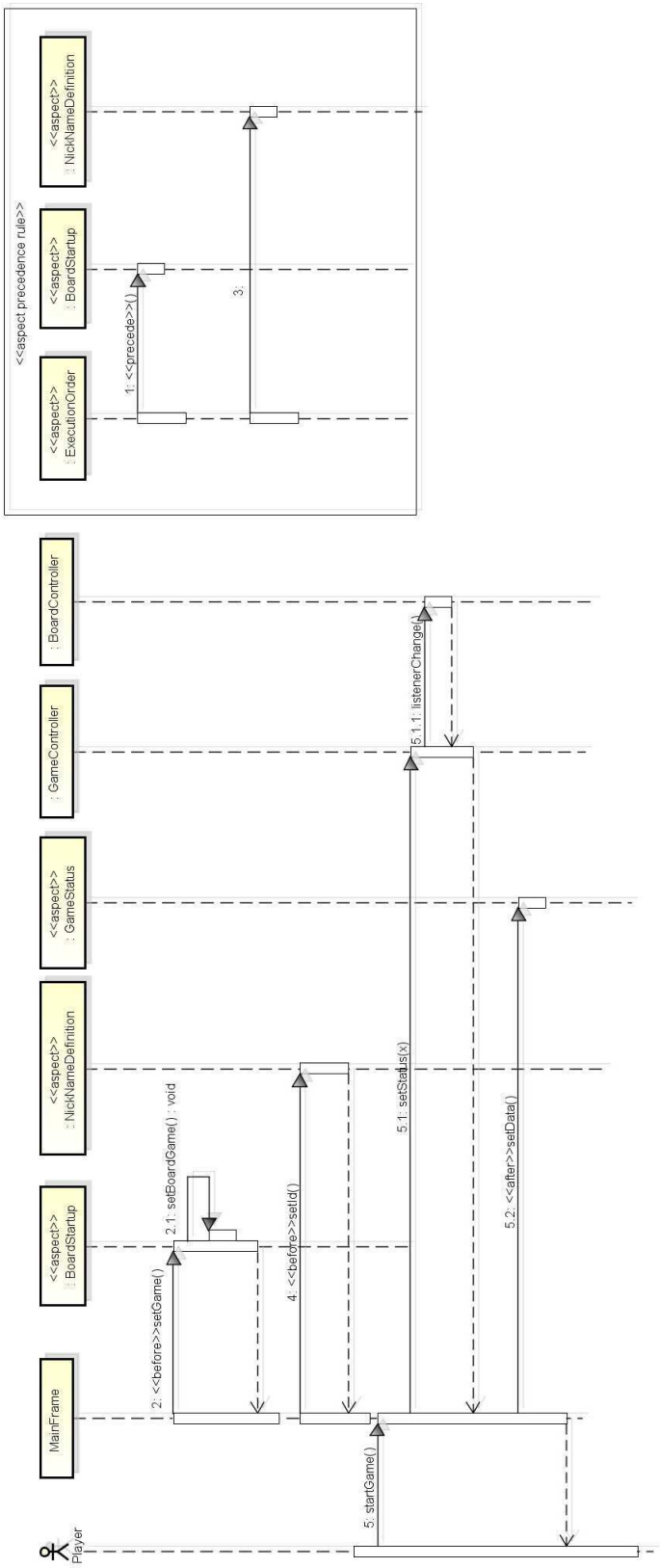

Figure C.4: UML+ Design for Task 1 - Sequence Diagram

2. The player's status is indicated by a colored button. When the button is green (status=0) it indicates that the player can play. If the button is red (status=1) the player cannot play due to some restrictions and another match must be started. Having said this, you are required to add a new functionality which aims to change the button color to yellow (status=2) when a player suffers a penalty and passes your next turn on to your opponent. This new functionality must be implemented in a separate aspect using an after returning advice.

Code C.2: Code of Task 2

```
class Player {
    int status = 0;
    int id =0;
    void setStatus(int x){
        status = x;
    }
}
```

```
class GameController {
   Player p;
   void addPenality(int code){
      p.setStatus(code);
   }
   int checkMove(){
      //check if a move is valid
   }
}
```

```
class Score{
   ...
   void setScore (Player player, int score){
      if(player.getStatus()==0) {
        //update current score
      }else {//bring the score to zero}
   }
}
```

```
class MainFrame{
   GameController g;
   void initPlay (){
     g.checkMove();
   }
   ...
}
```

```
abstract aspect ControllerBase {
   pointcut setData(): call(* *.set*());

}
```

```
aspect GameStatus extends ControllerBase{
   after( ): setData( ){ //Print a message }
}
```

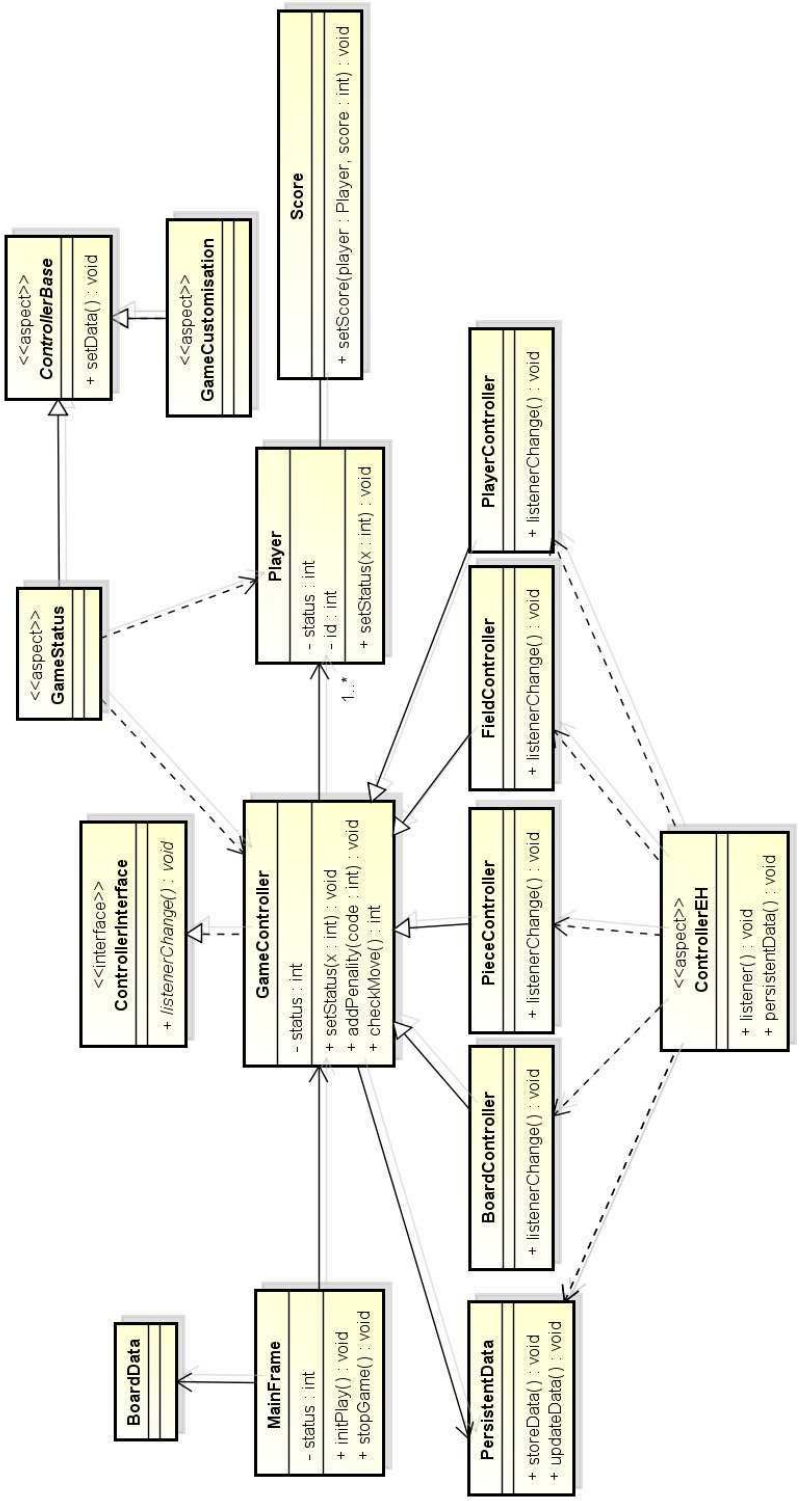

Figure C.5: UML Design for Task 2 - Class Diagram

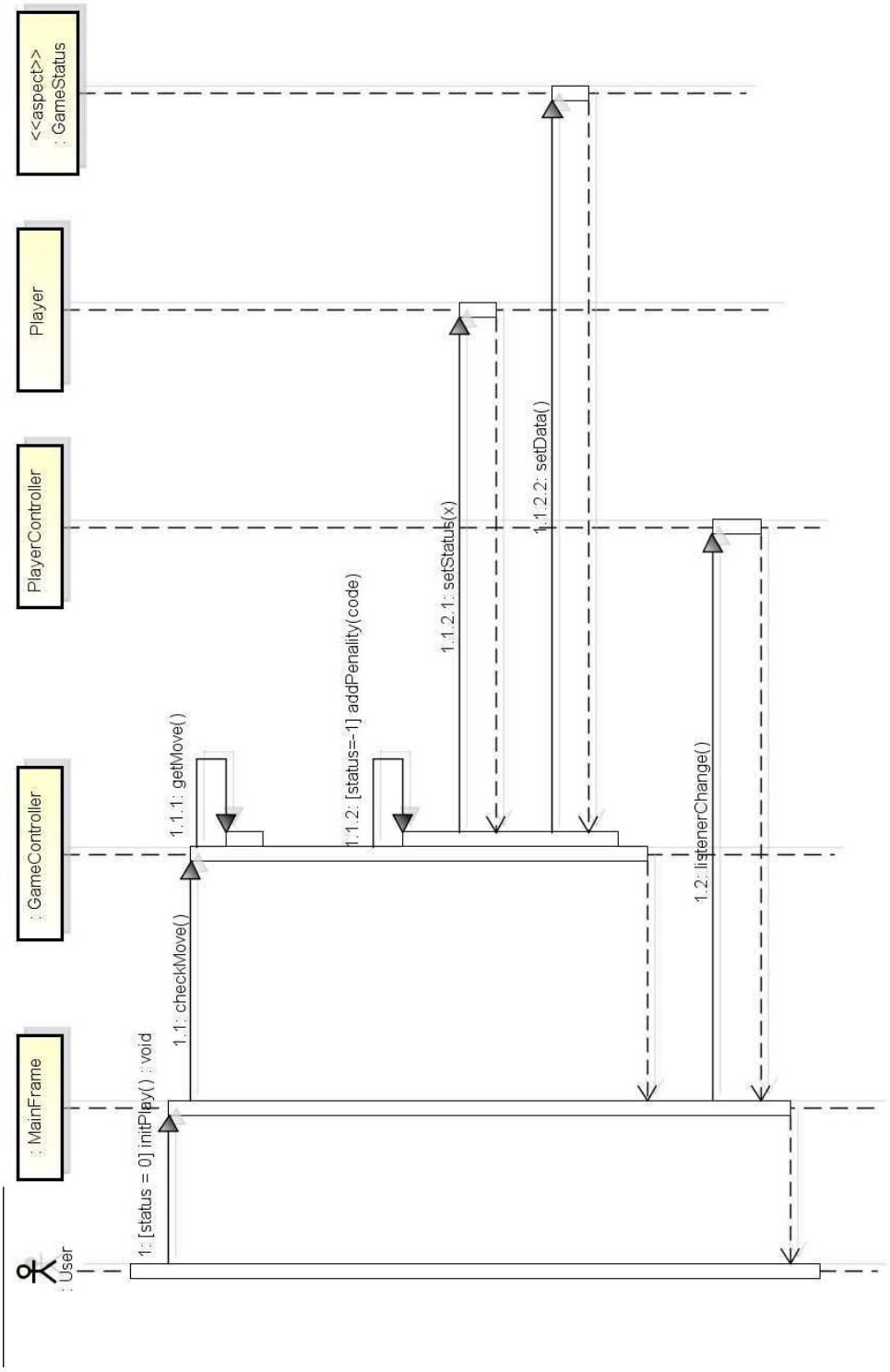

Figure C.6: UML Design for Task 2 - Sequence Diagram

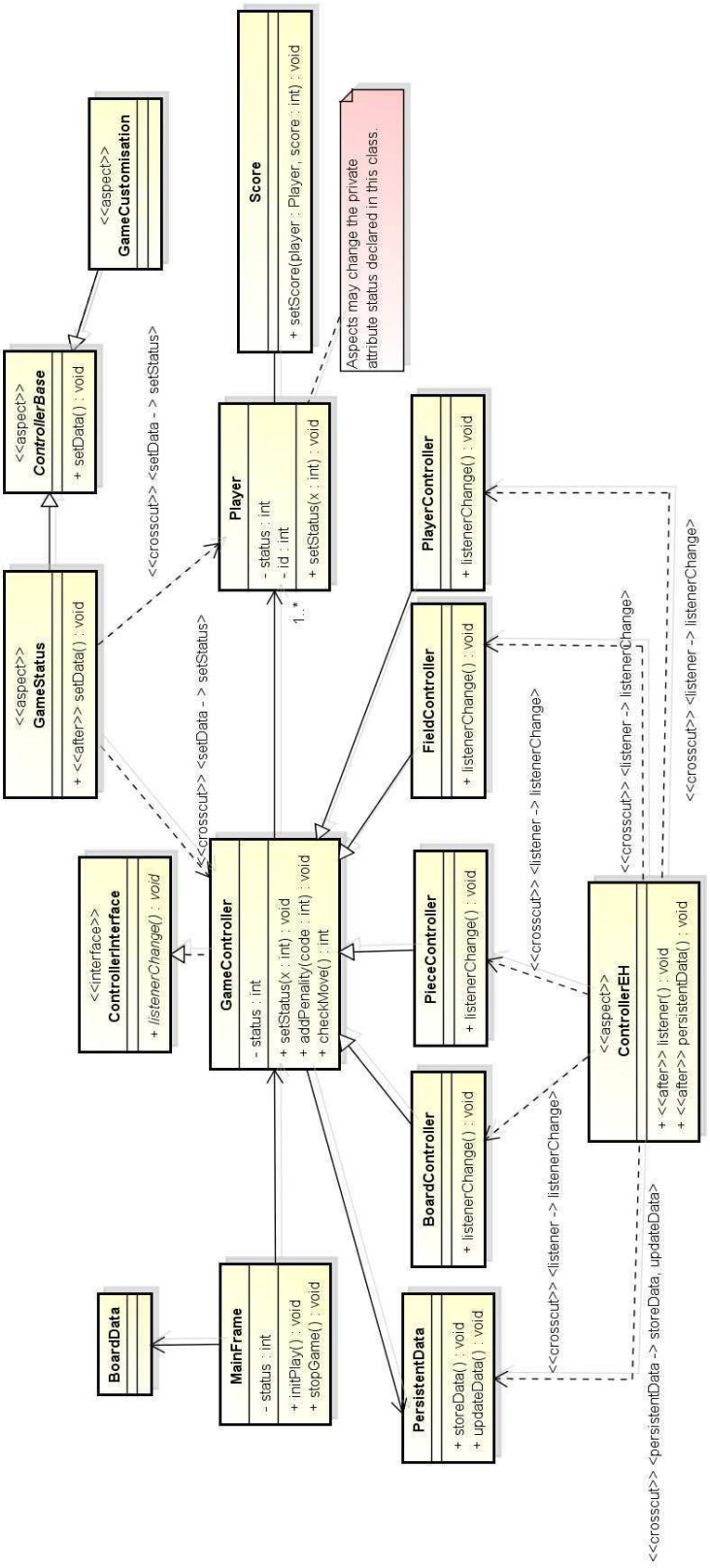

Figure C.7: UML+ Design for Task 2 - Class Diagram

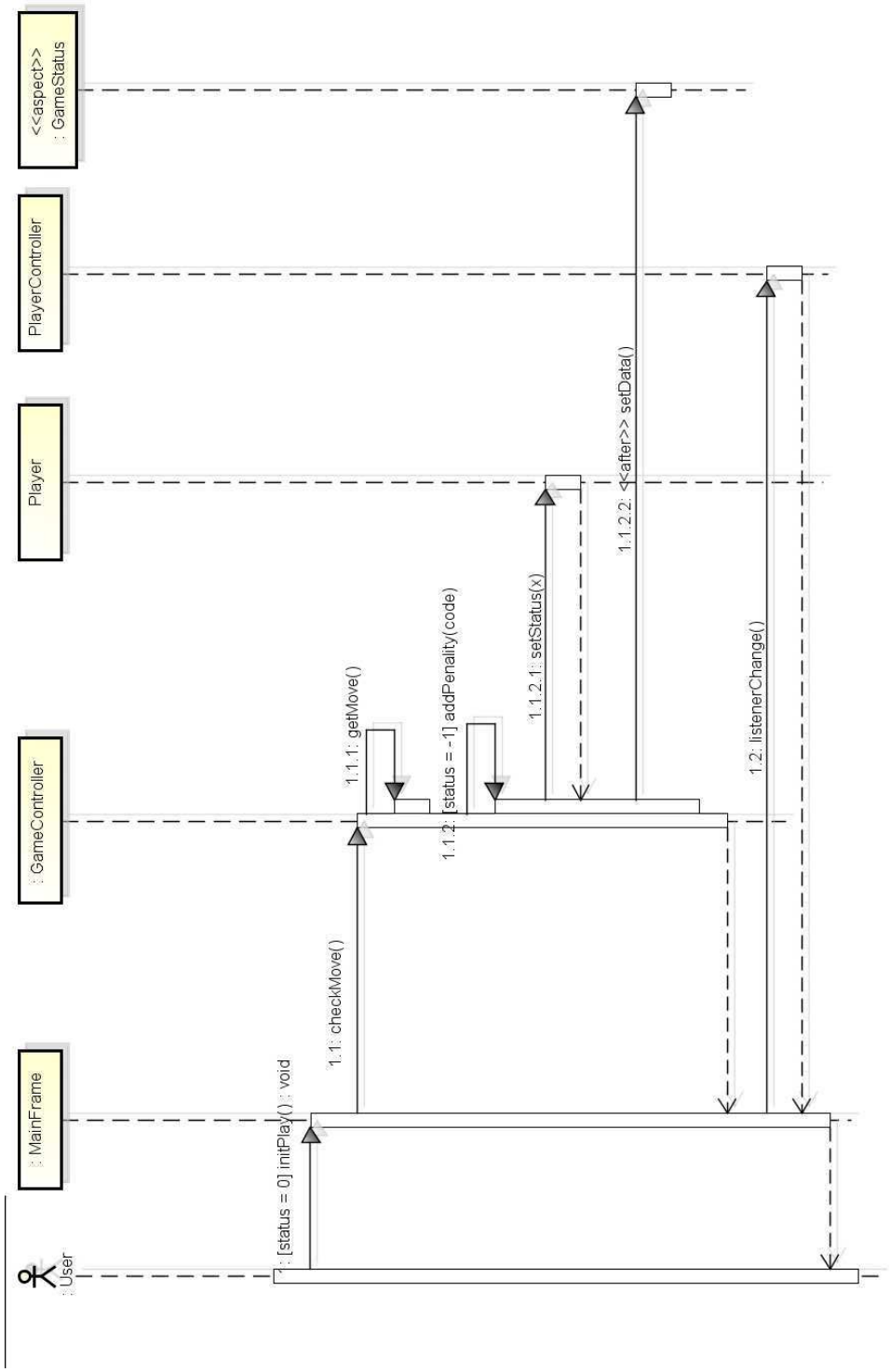

Figure C.8: UML+ Design for Task 2 - Sequence Diagram

3. The current version of Game UP allows both saving the board configuration in JPG format and informing the player about the result of such operation by message on the screen. The GameUP developers need your help to evolve the source code below in order to allow saving the game in XML format in addition to the JPG format. The level of detail expected in your answer is the same as illustrated in the code bellow.

Code C.3: Code of Task 3

```
class GameController{
   void saveBoard ( ){
     //save board in JPG format
...}
   ...
}
-----------------------------------------------
aspect BoardTracing{
   pointcut traceSaveBoard():call(void GameController.
      saveBoard( ))
   after( ): traceSaveBoard () {
      System.out.println(An JPG board image was successfully
         saved);
   }
   ...
}
-----------------------------------------------
abstract aspect ControllerBase {
   pointcut setData(): call(* *.set*());

}
-----------------------------------------------
aspect GameStatus extends ControllerBase{
   after( ): setData( ){ //Print a message }
}
```

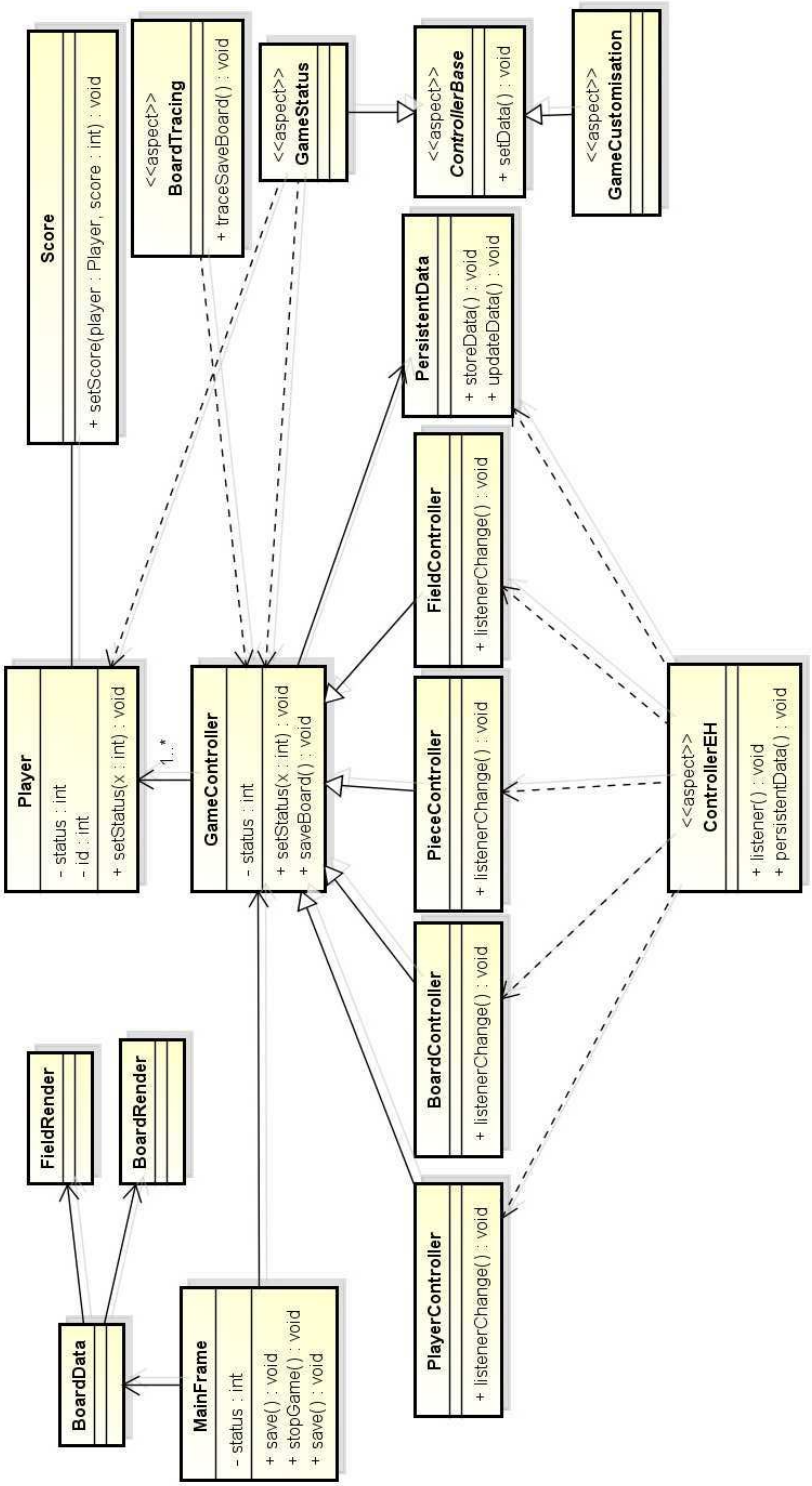

Figure C.9: UML Design for Task 3 - Class Diagram

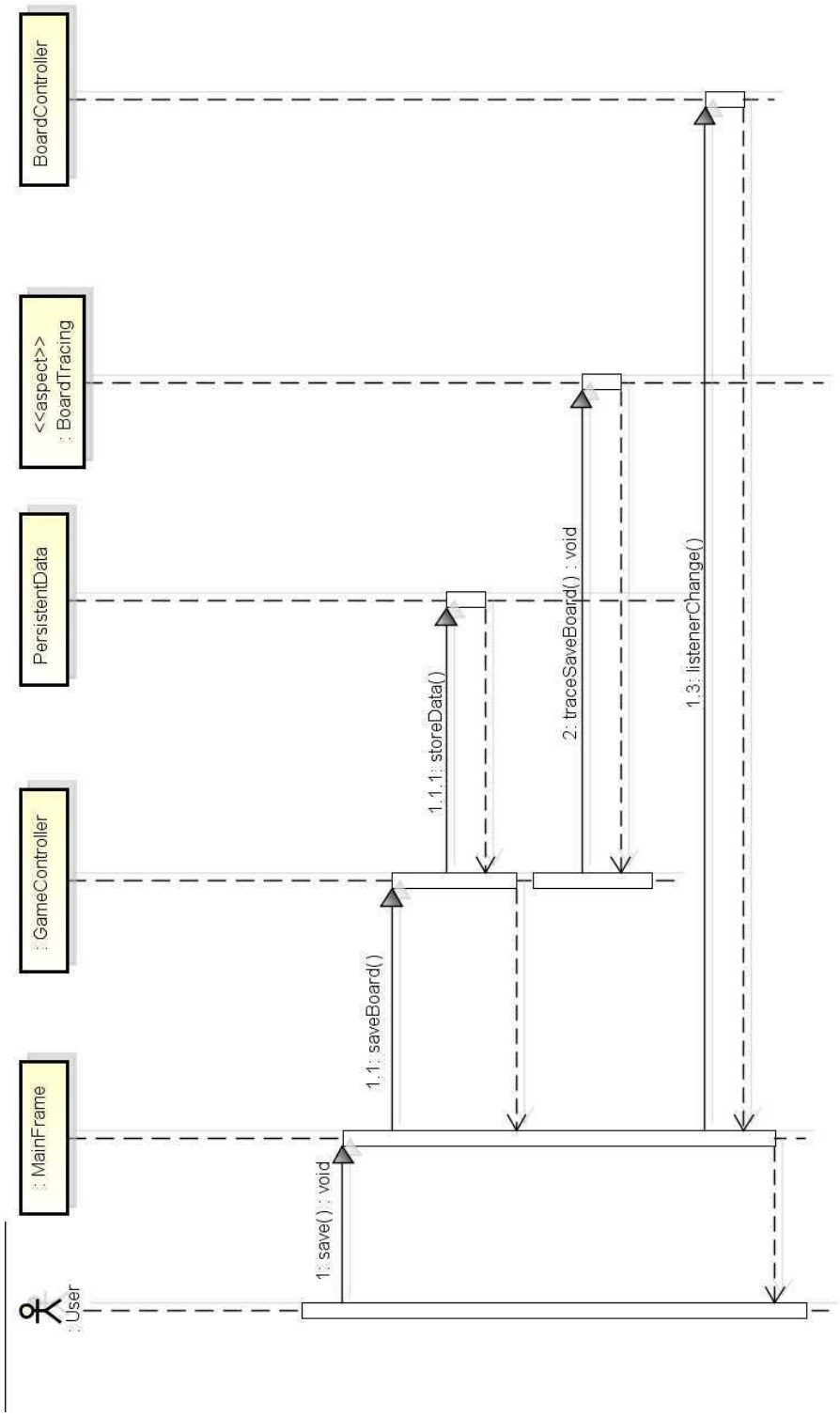

Figure C.10: UML Design for Task 3 - Sequence Diagram

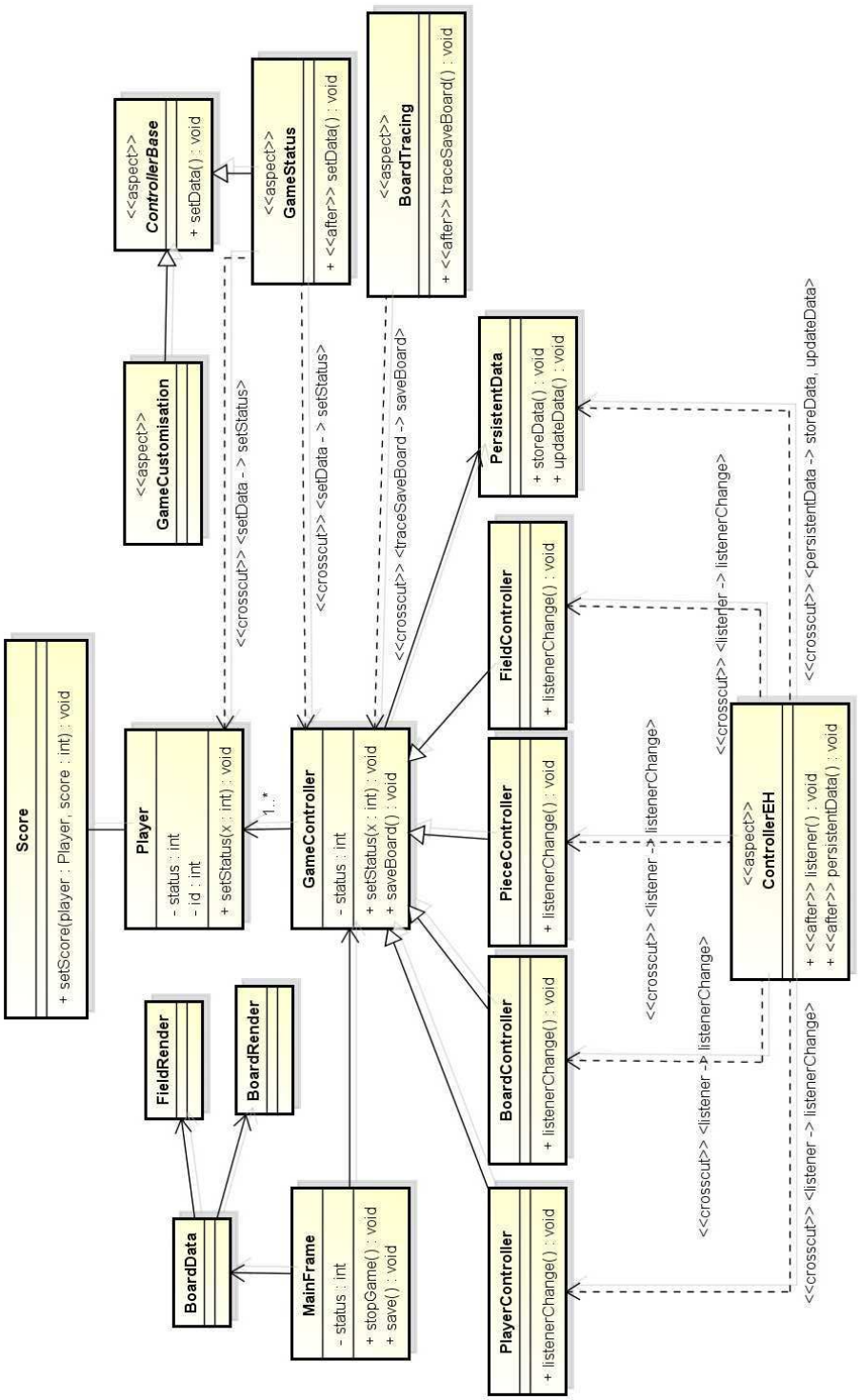

Figure C.11: UML+ Design for Task 3 - Class Diagram

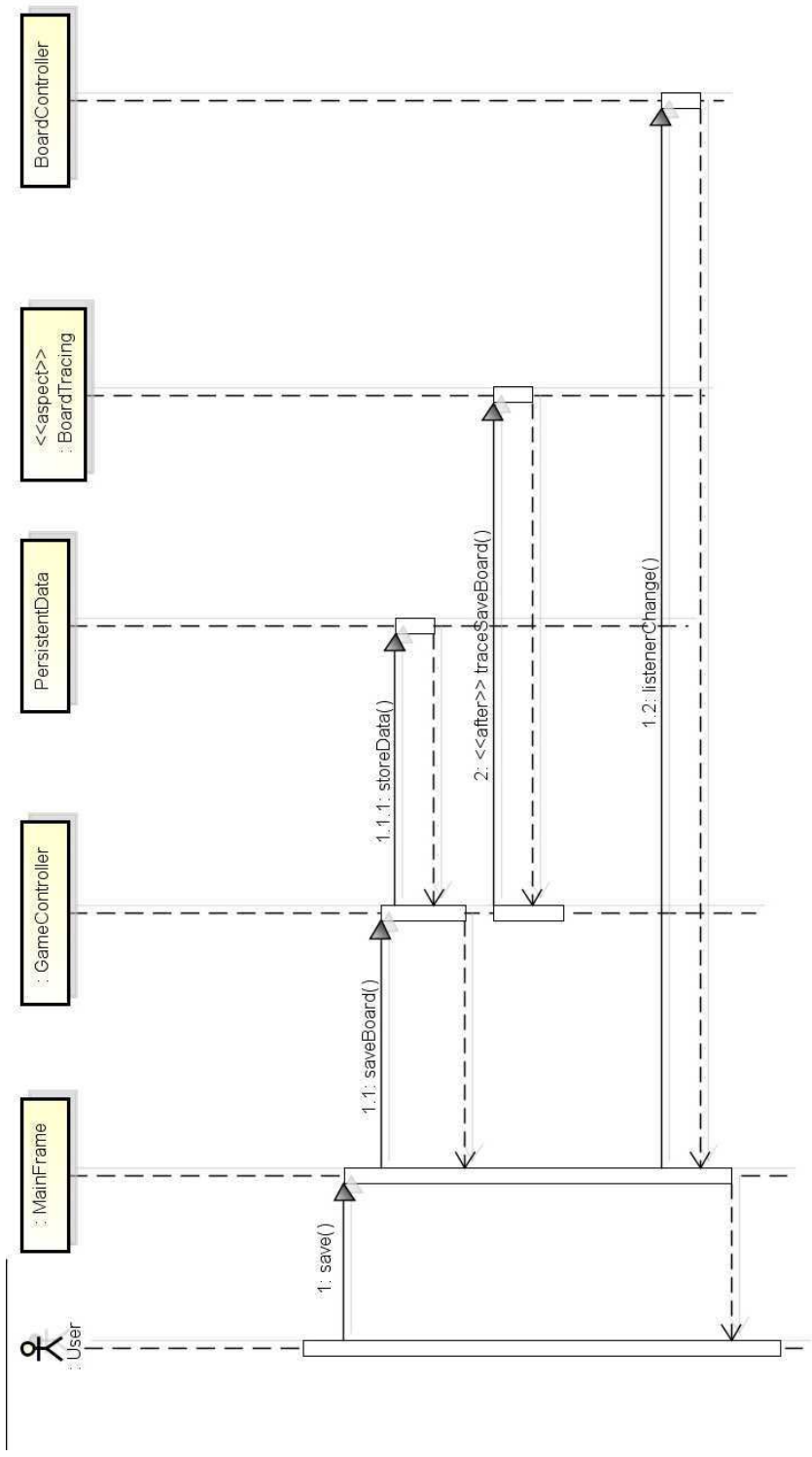

Figure C.12: UML+ Design for Task 3 - Sequence Diagram

4. Help GameUP developers to add the method `int saveScore( )` to the class `PersistentData` using AspectJ's mechanisms. This method aims at providing a new functionality, which is to save the player's score at the end of each game match.

Code C.4: Code of Task 4

```
class PersistentData {
   ...
   int saveJPG( ) { ... }
   int saveXML( ) { ... }
   ...
}
```

```
aspect BoardTracing{
   pointcut traceSaveBoard():call(int PersistentData.save*( )
      );
   ...
   after( ): traceSaveBoard (){
      System.out.println("Board Saved");
   }
   ...
}
```

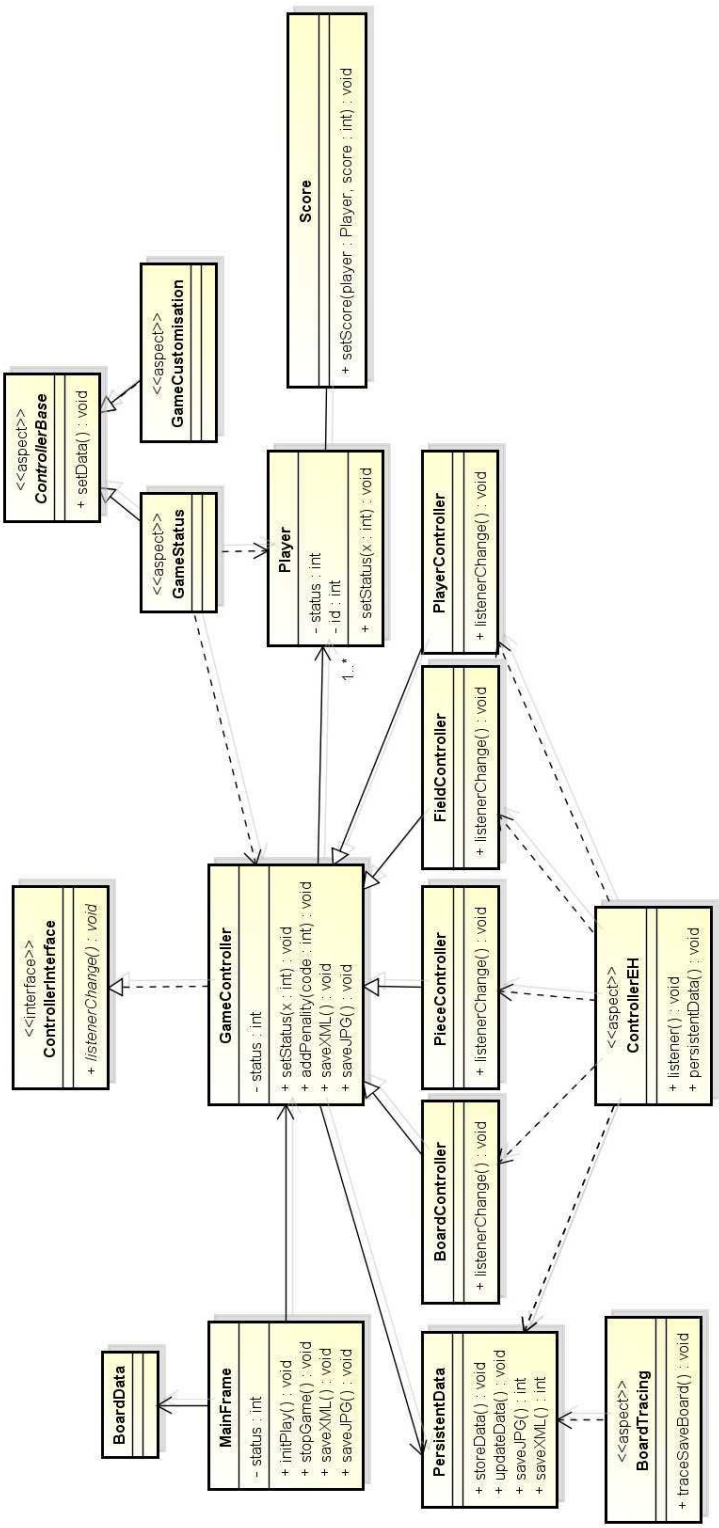

Figure C.13: UML Design for Task 4 - Class Diagram

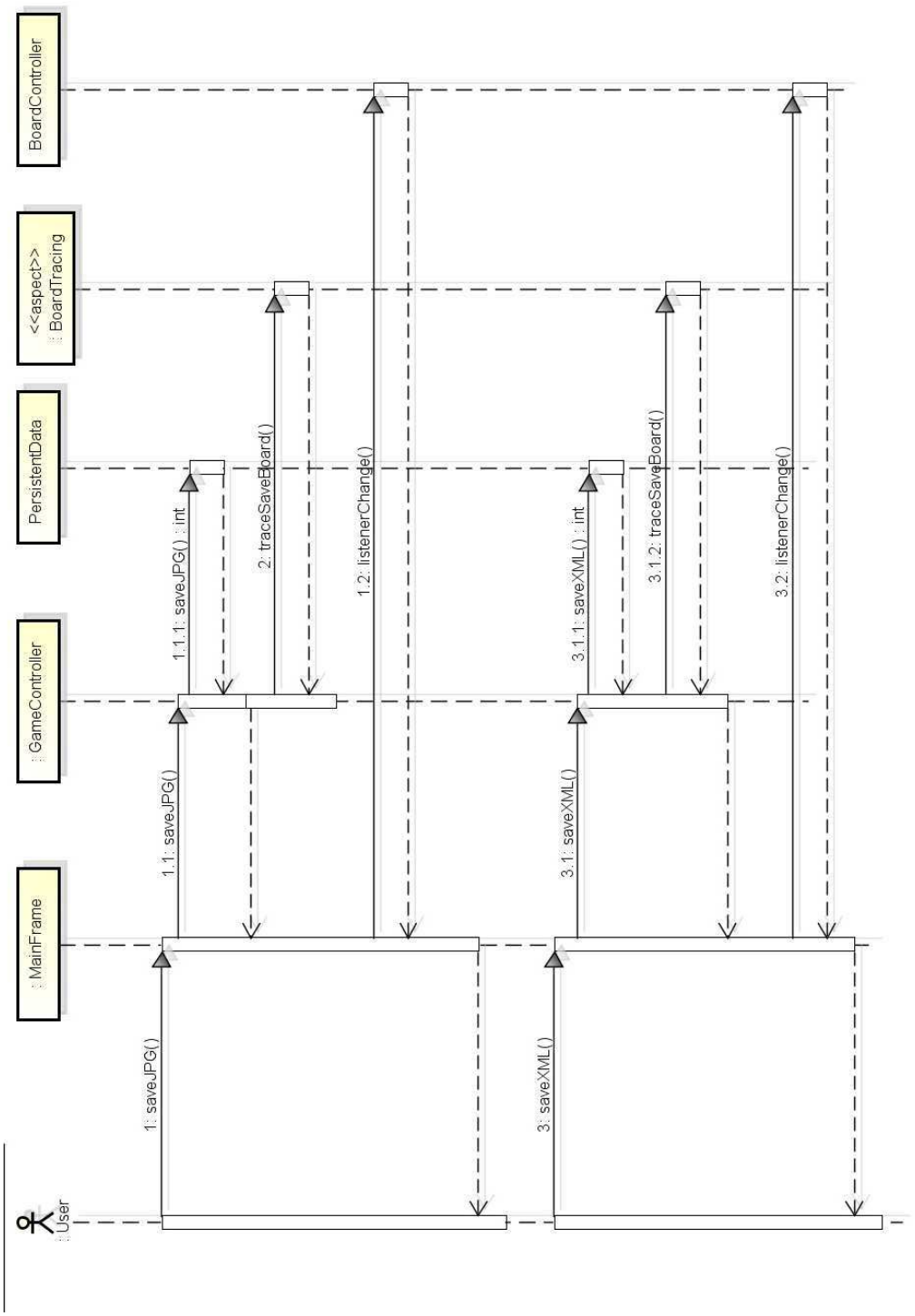

Figure C.14: UML Design for Task 4 - Sequence Diagram

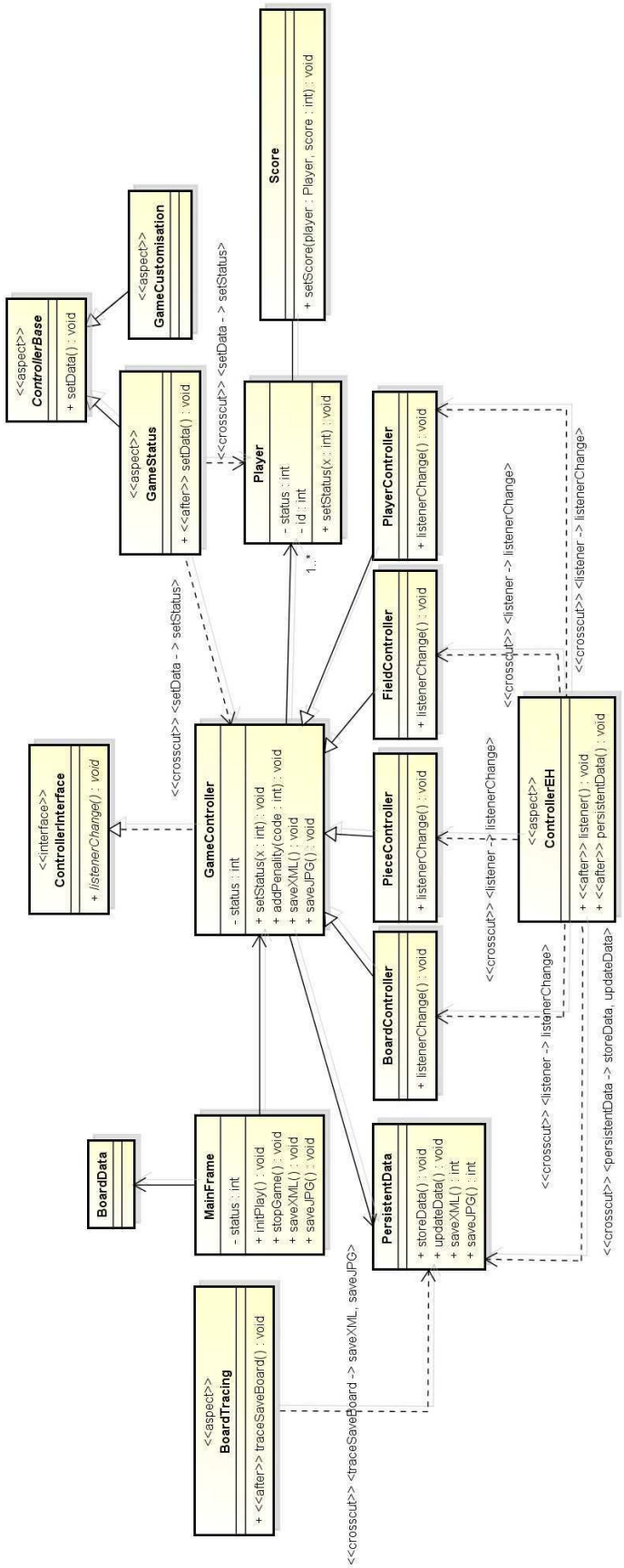

Figure C.15: UML+ Design for Task 4 - Class Diagram

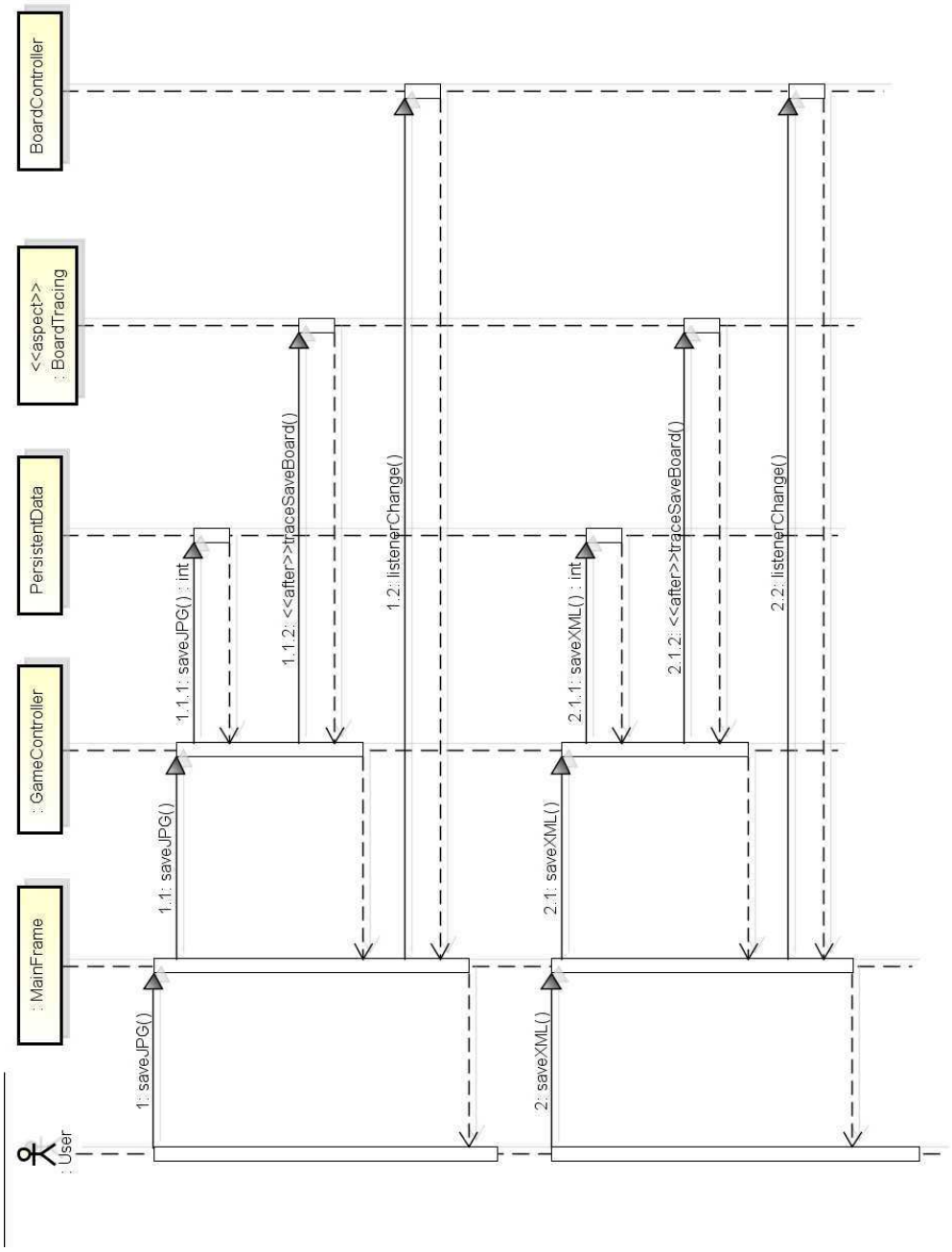

Figure C.16: UML+ Design for Task 4 - Sequence Diagram

C.2

Feedback Questionnaire

1. In your opinion, which were the characteristics present in the source code and also in the UML-based diagrams that make the changes more difficult to be performed? Why were these specific characteristics hindering the implementation of the required changes?

   _____

   _____

2. Please qualify to what extent the factors bellow exerted some influence on your execution of the changes required in the experiment. For each factor you must choose only one option considering the range from "No Influence" to "Extremely High".

   Factor 1: The scope of the pointcuts involved in the changes. Scope refers to the set of program elements (joinpoints) picket out by a given pointcut.

   a. No Influence
   b. Extremely Low
   c. Low
   d. High
   e. Extremely High

   Factor 2: The occurrence of multiples aspects sharing the same joinpoint.

   a. No Influence
   b. Extremely Low
   c. Low
   d. High
   e. Extremely High

   Factor 3: The existence of different types of modules, i.e. classes and aspects, involved in a pointcut.

   a. No Influence
   b. Extremely Low
   c. Low
   d. High
   e. Extremely High

   Factor 4: The dependency among aspects and classes are highly based on the program language syntax.

a. No Influence
b. Extremely Low
c. Low
d. High
e. Extremely High

Factor 5: The occurrence of methods with similar signature pattern (e.g. names starting with the prefixes "set" and "get").

a. No Influence
b. Extremely Low
c. Low
d. High
e. Extremely High