TR-QC-07-2014

A collection of models of a bike-sharing case study

Revision: 1; May 28, 2014

Author(s): Maurice H. ter Beek (CNR–ISTI), Alessandro Fantechi (CNR–ISTI), Stefania Gnesi (CNR–ISTI), and Franco Mazzanti (CNR–ISTI)

Publication date: May 28, 2014

Funding Scheme: Small or medium scale focused research project (STREP)
Topic: ICT-2011 9.10: FET-Proactive ‘Fundamentals of Collective Adaptive Systems’ (FOCAS)
Project number: 600708

Coordinator: Jane Hillston (UEDIN)
e-mail: `Jane.Hillston@ed.ac.uk`
Fax: +44 131 651 1426

Part. no.	Participant organisation name	Acronym	Country
1 (Coord.)	University of Edinburgh	UEDIN	UK
2	Consiglio Nazionale delle Ricerche – Istituto di Scienza e Tecnologie della Informazione “A. Faedo”	CNR	Italy
3	Ludwig-Maximilians-Universität München	LMU	Germany
4	Ecole Polytechnique Fédérale de Lausanne	EPFL	Switzerland
5	IMT Lucca	IMT	Italy
6	University of Southampton	SOTON	UK

Contents

Abstract

Bike-sharing systems are gaining popularity not only as a sustainable means of smart transportation in urban environments, but also as a challenging case study that offers interesting run-time optimization problems. As a case study within Quanticol, we have observed how such systems possess a wide variety of different features. We have therefore applied variability analysis to define a family of bike-sharing systems, covering the specification of a discrete feature model, specification of several non-functional quantitative properties, and behavioural specifications. Subsequently, we have sought support in available tools, establishing a tool chain of (academic) tools each covering a different aspect of the system, from feature modelling to product derivation and from quantitative evaluation of the attributes of products to model checking value-passing modal specifications. The tool chain was experimented to complement more sophisticated product-based analyses. This technical report collects the complete specifications of the models that were used in the tool chain.

1 Introduction

In [2, 3, 5, 6], we presented a case study for Quanticol by defining a family of bike-sharing systems that involved the specification of a discrete feature model, the specification of several kinds of non-functional quantitative properties, and behavioural specifications. In particular, in [6] we established a tool chain of (academic) tools each covering a different aspect of the system, from feature modelling to product derivation and from quantitative evaluation of the attributes of products to model checking value-passing modal specifications. The tools included are S.P.L.O.T. [12], FeatureIDE [16], Clafer [1] and ClaferMOO [14], and VMC [4, 7].

This technical report collects the complete specifications of the models that were used in the tool chain and as such it forms a companion to [6]. It moreover forms a reference for those interested in a case study combining discrete features, non-functional quantitative properties and behavioral specifications with safety properties, in which both the discrete domain features and the non-functional properties were derived from analysis of an actual industrial case. For this we collaborated with "PisaMo S.p.A. azienda per la mobilità pisana", an in-house public mobility company of the Municipality of Pisa, that recently introduced the public BSS *CicloPi* in the city of Pisa.

2 Feature models of a bike-sharing system

To develop a feature model of a bike-sharing system we performed a requirements elicitation in the form of text mining documents from the literature describing current bike-sharing systems [8, 13] and the bike-sharing system of Pisa [10, 15]. This allowed us to extract the main features of bike-sharing systems and to identify their commonalities and variabilities. This led to bikes equipped with an optional localization feature (RFID or GPS) and an optional antithieves feature (which requires GPS though), parking stations with a capacity that is either fixed permanent or fixed portable or flexible, optional maintenance and redistribution of bikes, and an optional incentive scheme based on rewards.

2.1 Feature model of a bike-sharing system in S.P.L.O.T.

S.P.L.O.T. [12] allows to edit, debug, analyze, configure, share, and download feature models. In particular, it allows to save models online to consult them later or to export them in the Simple XML Feature Model (SXFM) format, which is a concise textual format. It does not allow code generation, nor does it provide a way to render feature models in the graphical FODA syntax [11]. The main reason for which we decided to specify the feature model also in S.P.L.O.T. is that it is a de facto standard for sharing feature models publicly (its feature model repository currently has nearly 400 entries).

```
<feature_model name="Bikesharing">
<meta>
<data name="description">Bike-sharing system</data>
<data name="creator">S. Gnesi</data>
<data name="address"></data>
<data name="email">gnesi@isti.cnr.it</data>
<data name="phone"></data>
<data name="website"></data>
<data name="organization">ISTI-CNR, Italy</data>
<data name="department"></data>
<data name="date"></data>
<data name="reference"></data>
</meta>
<feature_tree>
:r Bikesharing(_r)
        :o Status(_r_1)
                :g (_r_1_7) [1,*]
                        : RTInfoWeb(_r_1_7_8)
                        : AllBikesNow(_r_1_7_9)
        :m Bike(_r_2)
        :o Localization(_r_2_10)
                :g (_r_2_10_11) [1,*]
                                : GPS(_r_2_10_11_12)
                                : RFID(_r_2_10_11_13)
                :o Antithieves(_r_2_14)
        :m DockingStation(_r_3)
                :g (_r_3_15) [1,1]
                        : Fixed(_r_3_15_16)
                        : FixedPortable(_r_3_15_17)
                        : Flexible(_r_3_15_18)
        :o Maintenance(_r_4)
        :o Redistribution(_r_5)
                :o Reward(_r_5_19)
        :m Users(_r_6)
                :m AccessMgmtSys(_r_6_20)
                        :g (_r_6_20_21) [1,*]
                                : SmartCard(_r_6_20_21_22)
                                : SmartPhone(_r_6_20_21_23)
                                : Keycard(_r_6_20_21_24)
                :m UserRegistration(_r_6_28)
                        :g (_r_6_28_29) [1,*]
                                : KioskReg(_r_6_28_29_30)
                                        :o TouchScreen(_r_6_28_29_30_33)
                                        :o KeycardReader(_r_6_28_29_30_35)
                                        :o CreditCards(_r_6_28_29_30_36)
                                        :o KeycardDispenser(_r_6_28_29_30_30)
                                : DockStat(_r_6_28_29_31)
                                : WebReg(_r_6_28_29_32)
</feature_tree>
<constraints>
constraint_1:~_r_1_7_9 or _r_2_10_11_12
constraint_8:_r_2_10_11_12 or ~_r_2_14
constraint_10:_r_6_20_21_24 or ~_r_6_28_29_30_30
constraint_11:~_r_6_20_21_24 or _r_6_28_29_30_35
constraint_13:~_r_6_20_21_24 or _r_6_28_29_30_30
</constraints>
</feature_model>
```

2.2 Feature model of a bike-sharing system in FeatureIDE

FeatureIDE [16] is a tool that does allow to generate feature models in the graphical FODA syntax as well as code (Java or C++), starting from a feature model. It accepts feature models in an XML format and it offers the automatic conversion of SXFM files (i.e. feature models created with S.P.L.O.T.).

```
<featureModel chosenLayoutAlgorithm="1">
    <struct>
        <and mandatory="true" name="Bikesharing">
            <and name="Status">
                <or mandatory="true" name="Or_Group">
                    <feature mandatory="true" name="RTInfoWeb"/>
                    <feature mandatory="true" name="AllBikesNow"/>
                </or>
            </and>
            <and mandatory="true" name="Bike">
                <and name="Localization">
                    <or mandatory="true" name="Or_Group_1">
                        <feature mandatory="true" name="GPS"/>
                        <feature mandatory="true" name="RFID"/>
                    </or>
                </and>
                <feature name="Antithieves"/>
            </and>
            <and mandatory="true" name="DockingStation">
                <alt mandatory="true" name="Alt_Group">
                    <feature mandatory="true" name="Fixed"/>
                    <feature mandatory="true" name="FixedPortable"/>
                    <feature mandatory="true" name="Flexible"/>
                </alt>
            </and>
            <feature name="Maintenance"/>
            <and name="Redistribution">
                <feature name="Reward"/>
            </and>
            <and mandatory="true" name="Users">
                <and mandatory="true" name="AccessMgmtSys">
                    <or mandatory="true" name="Or_Group_2">
                        <feature mandatory="true" name="SmartCard"/>
                        <feature mandatory="true" name="SmartPhone"/>
                        <feature mandatory="true" name="Keycard"/>
                    </or>
                </and>
                <and mandatory="true" name="UserRegistration">
                    <or mandatory="true" name="Or_Group_3">
                        <and mandatory="true" name="KioskReg">
                            <feature name="TouchScreen"/>
                            <feature name="KeycardReader"/>
                            <feature name="CreditCards"/>
                            <feature name="KeycardDispenser"/>
                        </and>
                        <feature mandatory="true" name="DockStat"/>
                        <feature mandatory="true" name="WebReg"/>
                    </or>
                </and>
            </and>
        </and>
    </struct>
    <constraints>
        <rule>
            <disj>
                <not>
                    <var>AllBikesNow</var>
                </not>
                <var>GPS</var>
            </disj>
        </rule>
        <rule>
            <disj>
                <var>GPS</var>
                <not>
                    <var>Antithieves</var>
                </not>
            </disj>
        </rule>
        <rule>
            <disj>
                <var>Keycard</var>
                <not>
                    <var>KeycardDispenser</var>
                </not>
            </disj>
        </rule>
        <rule>
            <disj>
                <not>
                    <var>Keycard</var>
                </not>
                <var>KeycardReader</var>
            </disj>
        </rule>
        <rule>
            <disj>
                <not>
                    <var>Keycard</var>
                </not>
                <var>KeycardDispenser</var>
            </disj>
        </rule>
    </constraints>
    <comments/>
    <featureOrder userDefined="true">
        <feature name="Bikesharing"/>
        <feature name="Status"/>
        <feature name="Or_Group"/>
        <feature name="RTInfoWeb"/>
        <feature name="AllBikesNow"/>
        <feature name="Bike"/>
        <feature name="Localization"/>
        <feature name="Or_Group_1"/>
        <feature name="GPS"/>
        <feature name="RFID"/>
        <feature name="Antithieves"/>
        <feature name="DockingStation"/>
        <feature name="Alt_Group"/>
        <feature name="Fixed"/>
        <feature name="FixedPortable"/>
        <feature name="Flexible"/>
        <feature name="Maintenance"/>
        <feature name="Redistribution"/>
        <feature name="Reward"/>
        <feature name="Users"/>
        <feature name="AccessMgmtSys"/>
        <feature name="Or_Group_2"/>
        <feature name="SmartCard"/>
        <feature name="SmartPhone"/>
        <feature name="Keycard"/>
        <feature name="UserRegistration"/>
        <feature name="Or_Group_3"/>
        <feature name="KioskReg"/>
        <feature name="TouchScreen"/>
        <feature name="CreditCards"/>
        <feature name="KeycardDispenser"/>
        <feature name="KeycardReader"/>
        <feature name="DockStat"/>
        <feature name="WebReg"/>
    </featureOrder>
</featureModel>
```

2.3 Attributed feature model of a bike-sharing system in Clafer

In the context of Quanticol we are specifically interested also in quantitative analyses of bike-sharing systems, in the sense that we consider the behaviour of the components of a bike-sharing system to exhibit variability not only in the kind of features they possess, but also in the quantitative (non-functional) characteristics of their features. To model this, we can add attributes and quantitative constraints among attributes and features to our bike-sharing system specification and then perform quantitative analyses, i.e. we model and analyze attributed feature models. Neither S.P.L.O.T. nor FeatureIDE currently cater for attributed feature models.

Clafer [1, 14] is a lightweight textual modelling language for software (product lines) that does allow to specify attributed feature models. It moreover allows the automatic translation of files in S.P.L.O.T.'s SXFM format into Clafer's CFR format. After translation, we annotated the features of the feature model of the bike-sharing system with attributes and defined global quantitative constraints over these attributes. We limited this to the cost and customer satisfaction and, in specific cases, capacity and security. Consequently, the global constraints aim to minimize the total cost of a configuration (product) and at the same time maximize customer satisfaction, capacity, and security of a bike-sharing system. We obtained realistic numbers for the costs and security from documents from companies that sell bike-sharing systems (obtained via PisaMo) and kept their ratio in our model. The numbers for customer satisfaction stem from discussions with PisaMo. We had to reduce the Clafer specification by leaving out the entire Users feature and its subfeatures, the reason being that ClaferMOOVisualizer [14] currently runs out of memory in case of too large models.

This ClaferMOO extension of Clafer was specifically introduced to support attributed feature models as well as the resulting complex multi-objective optimization goals. A multi-objective optimization problem has a set of solutions, known as the Pareto front, that represents the trade-offs between two or more conflicting objectives. Intuitively, a Pareto-optimal solution is thus such that no objective can be improved without worsening another objective. A set of Pareto-optimal variants generated by ClaferMOO can be visualized (as a multi-dimensional space of optimal variants) and explored interactively in ClaferMOOVisualizer, which was specifically designed to support SPL scenarios. The tool can help understand differences among variants, establish their positioning with respect to various quality dimensions, select the most desirable variants, possibly by resolving trade-offs, and understand the impact that changes made during a product line's evolution have on a variant's quality dimensions.

```
abstract Feature
    customersat : integer
    cost : integer

abstract SecurityFeature : Feature
    security : integer

abstract CapacityFeature : Feature
    capacity : integer

abstract Bikesharing
    or Status : Feature ?
        [ customersat = 0 ]
        [ cost = 0 ]
        RTInfoWeb : Feature
            [ customersat = 10 ]
            [ cost = 5 ]
        AllBikesNow : Feature
            [ customersat = 20 ]
            [ cost = 10 ]
    Bike : SecurityFeature
        [ customersat = 0 ]
        [ cost = 0 ]
        [ security = 0 ]
        or Localization : SecurityFeature ?
            [ customersat = 0 ]
            [ cost = 0 ]
            [ security = 0 ]
            RFID : SecurityFeature
                [ customersat = 10 ]
                [ cost = 10 ]
                [ security = 1 ]
            GPS : SecurityFeature
                [ customersat = 15 ]
                [ cost = 15 ]
                [ security = 2 ]
        Antithieves : SecurityFeature ?
            [ customersat = 5 ]
            [ cost = 7 ]
            [ security = 4 ]
    xor DockingStation : CapacityFeature
        [ customersat = 0 ]
        [ cost = 0 ]
        [ capacity = 0]
        Fixed : CapacityFeature
            [ customersat = 17 ]
            [ cost = 30 ]
            [ capacity = 10]
        FixedPortable: CapacityFeature
            [ customersat = 20 ]
            [ cost = 35 ]
            [ capacity = 10]
        Flexible: CapacityFeature
            [ customersat = 23 ]
            [ cost = 40 ]
            [ capacity = 20]
    Maintenance : Feature ?
        [ customersat = 15 ]
        [ cost = 10 ]
    Redistribution : Feature ?
        [ customersat = 15 ]
        [ cost = 10 ]
        Reward : Feature ?
            [ customersat = 5 ]
            [ cost = 10 ]
    [ Antithieves => GPS ]
    [ AllBikesNow => GPS ]

    total_customersat : integer = sum Feature.customersat
    total_cost : integer = sum Feature.cost
    total_security : integer = sum SecurityFeature.security
    total_capacity : integer = sum CapacityFeature.capacity

Mybike : Bikesharing

<< max Mybike.total_customersat >>
<< min Mybike.total_cost >>
<< max Mybike.total_security >>
<< max Mybike.total_capacity >>
```

3 Behavioural models of bike-sharing systems in VMC

We first model a simple bike-sharing system with the possibility of having a dynamic redistribution scheme as an optional feature in §3.1, followed by the addition of a user in §3.2. Without loss of generality, we assume in both cases a bike-sharing station with 2 as its maximum capacity.

Inspired by [9], we consequently model a possibly infinite number of users that take a bike from station I to station J. Initially, station I has N bikes, which it delivers (when available) to a requesting user or accepts from a returning user. If the station receives more than M bikes, the exceeding $N-M$ bikes are distributed to station J. Station I must accept all bikes distributed by other stations or returned by a user (possibly for redistribution). It could easily be extended to N stations and K groups of users that take a bike from one station to another. We consider the case of one user group in §3.3 and that for two user groups in §3.4.

We specify these behavioural models of a family of bike-sharing systems in the value-passing modal process algebra accepted by VMC v6.0, which is formally defined in [6]. Processes can pass and receive integer parameter values (and store them in a variable preceded by a '?'), actions can be optional in which case they are typed `may`, and nondeterministic choice can be guarded by a comparison of values. A system definition must be complemented with a top term of the form `net SYSTEM = P`, where `P` is the initial process (or composition of processes). Synchronous parallel composition is parametrized by the actions /.../ to synchronize.

3.1 A docking station with capacity 2

```
Station(X) = request.StationBikeRequested(X)
StationBikeRequested(Y) =
    [Y<1] ( nobike.Station(Y) + redistribute(may).Station(Y+2) ) +
    [Y>0] givebike.Station(Y-1)

net BSS = Station(2)
```

3.2 A docking station with capacity 2 and a user

```
Station(X) = request.StationBikeRequested(X)
StationBikeRequested(Y) =
    [Y<1] ( nobike.Station(Y) + redistribute(may).Station(Y+2) ) +
    [Y>0] givebike.Station(Y-1)
```

```
User = request.(givebike.User + nobike.User + redistribute.User)

net BSS = Station(2) /request,givebike,nobike,redistribute/ User
```

3.3 Two docking stations with capacity N and one group of users

```
Station(I,N,J,M) =
    request(I).
        ( [N=0] nobike(I).Station(I,N,J,M) +
          [N>0] givebike(I).Station(I,N-1,J,M) ) +
    deliver(I).Station(I,N+1,J,M) +
    redistribute(may,?FROM,?TO,?K).
        ( [TO = I] Station(I,N+K,J,M) +
          [TO /= I] Station(I,N,J,M) ) +
    [N > M] redistribute(may,I,J,N-M).Station(I,M,J,M)

Users(I,J) =
    request(I).
        ( givebike(I).deliver(J).Users(I,J) +
          nobike(I).Users(I,J) )

net STATIONS = Station(s1,2,s2,2) /redistribute/ Station(s2,2,s1,2)
net USERS = Users(s1,s2)
net BSS = STATIONS /request,givebike,nobike,deliver/ USERS
```

3.4 Two docking stations with capacity N and two groups of users

```
Station(I,N,J,M) =
    request(I).
        ( [N=0] nobike(I).Station(I,N,J,M) +
          [N>0] givebike(I).Station(I,N-1,J,M) ) +
    deliver(I).Station(I,N+1,J,M) +
    redistribute(may,?FROM,?TO,?K).
        ( [TO = I] Station(I,N+K,J,M) +
          [TO /= I] Station(I,N,J,M) ) +
    [N > M] redistribute(may,I,J,N-M).Station(I,M,J,M)

Users(I,J) =
    request(I).
        ( givebike(I).deliver(J).Users(I,J) +
          nobike(I).Users(I,J) )

net STATIONS = Station(s1,2,s2,2) /redistribute/ Station(s2,2,s1,2)
net USERS = Users(s1,s2) // Users(s2,s1)
net BSS = STATIONS /request,givebike,nobike,deliver/ USERS
```

Acknowledgements

Research partly supported by the EU FP7-ICT FET-Proactive project QUANTICOL (600708) and by the Italian MIUR project CINA (PRIN 2010LHT4KM).

The authors would like to thank Marco Bertini from PisaMo s.p.a. for kindly sharing his expertise on the public *CicloPi* bike-sharing system with us.

References

[1] K. Bąk, K. Czarnecki, and A. Wąsowski. Feature and Meta-Models in Clafer: Mixed, Specialized, and Coupled. In *Revised Selected Papers of the 3rd International Conference on Software Language Engineering (SLE'10)* (B.A. Malloy, S. Staab, and M. van den Brand, eds.). *Lecture Notes in Computer Science* 6563, Springer, 2010, 102–122.

[2] M.H. ter Beek, A. Fantechi, and S. Gnesi. Challenges in Modelling and Analyzing Quantitative Aspects of Bike-Sharing Systems. To appear in *Proceedings of the 6th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA'14)* (T. Margaria and B. Steffen, eds.). *Lecture Notes in Computer Science*, Springer, 2014.

[3] M.H. ter Beek, S. Gnesi, and A. Fantechi. Chaining available tools to support the modelling and analysis of a bike-sharing product line: An experience report. Technical Report TR-QC-02-2013, December 2013.

[4] M.H. ter Beek, S. Gnesi, and F. Mazzanti. Demonstration of a model checker for the analysis of product variability. In *Proceedings 16th International Software Product Line Conference (SPLC'12)*. ACM, 2012, 242–245.

[5] M.H. ter Beek, S. Gnesi, and F. Mazzanti. Model Checking Value-Passing Modal Specifications. Technical Report TR-QC-03-2013, December 2013.

[6] M.H. ter Beek, S. Gnesi, and F. Mazzanti. Model Checking Value-Passing Modal Specifications. To appear in *Perspectives of System Informatics: Revised selected papers of the 9th International Andrei Ershov Memorial Conference (PSI'14)* (I. Virbitskaite and A. Voronkov, eds.). *Lecture Notes in Computer Science*, Springer, 2014.

[7] M.H. ter Beek, F. Mazzanti, and A. Sulova. VMC: A Tool for Product Variability Analysis. In *Proceedings 18th International Symposium on Formal Methods (FM'12)* (D. Giannakopoulou and D. Méry, eds.). *LNCS* 7436, Springer, 2012, 450–454.

[8] P. DeMaio. Bike-sharing: History, Impacts, Models of Provision, and Future. *Journal of Public Transportation* 12, 4 (2009), 41–56.

[9] C. Fricker and N. Gast. Incentives and Redistribution in Bike-Sharing Systems with Stations of Finite Capacity. arXiv:1201.1178v3 [nlin.AO], September 2013.

[10] Luisa Gianfrotta and Simone Topazzini. Progettare Servizi Pubblici: Elaborazione di un Modello per lo Sviluppo di Nuovi Servizi e sua Applicazione al caso Bike Sharing di Pisa. Master's Thesis, Università di Pisa, 2013. In Italian.

[11] Kyo C. Kang, Sholom G. Cohen, James A. Hess, William E. Novak, and A. Spencer Peterson. Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-21, Carnegie Mellon University, November 1990.

[12] M. Mendonça, M. Branco, and D.D. Cowan. S.P.L.O.T.: Software Product Lines Online Tools. In *Companion Proceedings of the 24th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'09)* (S. Arora and G.T. Leavens, eds.). ACM, 2009, 761–762.

[13] P. Midgley. Bicycle-Sharing Schemes: Enhancing Sustainable Mobility in Urban Areas. Background Paper CSD19/2011/BP8, Commission on Sustainable Development, United Nations Department of Economic and Social Affairs, May 2011.

[14] A. Murashkin, M. Antkiewicz, D. Rayside, and K. Czarnecki. Visualization and Exploration of Optimal Variants in Product Line Engineering. In *Proceedings of the 17th International Software Product Line Conference (SPLC'13)* (T. Kishi, S. Jarzabek, and S. Gnesi, eds.). ACM, 2013, 111–115.

[15] Chiara Niccolai and Eleonora Zanzi. Progettare i servizi: Creazione di un modello di validità generale e applicazione al servizio di Bike Sharing a Pisa. Master's Thesis, Università di Pisa, 2013. In Italian.

[16] T. Thüm, C. Kästner, F. Benduhn, J. Meinicke, G. Saake, and T. Leich. FeatureIDE: An Extensible Framework for Feature-Oriented Software Development. *Science of Computer Programming* 79 (2014), 70–85.