Global Decision Making Over Deep Variability in Feedback-Driven Software Development


To cite this version:

HAL Id: hal-03770004
https://inria.hal.science/hal-03770004
Submitted on 6 Sep 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Global Decision Making Over Deep Variability in Feedback-Driven Software Development

Jörg Kienzle  
McGill University  
Canada  
Joerg.Kienzle@mcgill.ca

Benoit Combemale  
University of Rennes  
France  
benoit.combemale@irisa.fr

Gunter Mussbacher  
McGill University  
Canada  
Gunter.Mussbacher@mcgill.ca

Omar Alam  
Trent University  
Canada  
omaralam@trentu.ca

Francis Bordeleau  
Ecole de technologie superieure  
Canada  
francis.bordeleau@etsmtl.ca

Lola Burgueño  
Open University of Catalonia  
Spain  
lburguenoc@uoc.edu

Gregor Engels  
Paderborn University  
Germany  
engels@upb.de

Jessie Galasso  
Université de Montréal  
Canada  
jessie.galasso-carbonnel@umontreal.ca

Jean-Marc Jezequel  
University of Rennes  
France  
jezequel@irisa.fr

Bettina Kemme  
McGill University  
Canada  
kemme@cs.mcgill.ca

Sebastien Mosser  
McMaster University  
Canada  
mossers@mcmaster.ca

Houari Sahraoui  
DIRO, Université de Montréal  
Canada  
sahraouh@iro.umontreal.ca

Max Schiedermeier  
McGill University  
Canada  
max.schiedermeier@mcgill.ca

Eugene Syriani  
University of Montreal  
Canada  
syriani@iro.umontreal.ca

ABSTRACT

To succeed with the development of modern software, organizations must have the agility to adapt faster to constantly evolving environments to deliver more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, business, development, and IT. However, stakeholders do not have sufficient automated support for global decision making, considering the increasing variability of the solution space, the frequent lack of explicit representation of its associated variability and decision points, and the uncertainty of the impact of decisions on stakeholders and the solution space. This leads to an ad-hoc decision making process that is slow, error-prone, and often favors local knowledge over global, organization-wide objectives. The Multi-Plane Models and Data (MP-MODA) framework explicitly represents and manages variability, impacts, and decision points. It enables automation and tool support in aid of a multi-criteria decision making process involving different stakeholders within a feedback-driven software development process where feedback cycles aim to reduce uncertainty. We present the conceptual structure of the framework, discuss its potential benefits, and enumerate key challenges related to tool supported automation and analysis within MP-MODA.

CCS CONCEPTS

• Software and its engineering → Collaboration in software development.

KEYWORDS

MODA, Iterative Software Development, Feedback Loop

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ASE ’22, October 10–14, 2022, Rochester, MI, USA  
© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 978-1-4503-9475-8/22/10… $15.00  
https://doi.org/10.1145/3551349.3559551  
https://doi.org/10.1145/3551349.3559551
1 INTRODUCTION

The software industry is experiencing revolutionary changes on how software is built and how its functionalities are offered to users [28]. Contemporary software systems are highly complex and heterogeneous, and their development involves a tremendous amount of knowledge from different domains. Furthermore, the unpredictability of an ever-changing environment, and the unprecedented pace of emergence and decay of technologies require deep changes on how we develop, deploy, and evolve software [1]. To succeed with the development of modern software, organizations must have the agility to adapt faster to constantly changing environments and to deliver more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, business, development, and IT.

For certain software, e.g., for smart phone applications, it has become easy to frequently release and deploy new versions, e.g., to address bug fixes, add support for new technology or introduce new features [25]. Because of this ease of updating, it is nowadays often expected that such software continuously evolves and improves.

One way to deal with the uncertainty of how an application must evolve is to explicitly plan for feedback-driven software evolution. The idea is to gather telemetry data from or about the running system [22], e.g., to measure the effects of the heterogeneity of execution environments, or to capture user preferences and expectations [5]. The data is then analyzed and the gained knowledge is used to guide the development of the next software release.

Over time, such evolution of software requires expertise and knowledge from many domains and levels of abstraction, ranging from low-level technical skills to legal and business know-how, and decisions made on one level and in one domain can have an impact on other domains as well. While DevOps [11] aims at structuring the teams and the system architecture to enable agility and autonomy to take decisions and make modifications locally, some decisions and modifications still have a more cross-cutting impact. Unfortunately, stakeholders currently do not have sufficient support for global decision making, considering the increasing variability of the solution space, the frequent lack of explicit representation of its associated variability and decision points, and the uncertainty of the impact of decisions. This leads to an ad-hoc decision making process that is slow, error-prone, and often favors local knowledge over global, organization-wide objectives.

In this new idea paper, we outline a framework called Multi-Plane MODA, as an extension of the existing Models and Data (MODA) framework [8], that explicitly represents and manages variability of the problem and solution space, impacts, and decision points to enable automation and tool support in aid of a multi-criteria decision making process involving different stakeholders within a feedback-driven software development process. In Section 2, we present a motivating example and relevant background information. Then we describe the conceptual structure of the MP-MODA framework, discuss its potential benefits, and revisit the proposed motivating example in Section 3. We list key challenges of MP-MODA related to tool supported automation and analysis in Section 4.

2 MOTIVATION AND BACKGROUND

Motivating Example. Let us consider a company developing a new version of a software for a Heating, Ventilation and Air Conditioning (HVAC) system, which comprises both software and hardware components such as humidity and temperature sensors, coordinated processing units for data analysis, and actuators. Numerous stakeholders are involved and deal with different decisions. For instance, the business team decides on the functionality to be offered whereas the development team determines the infrastructure of the solution (whether to host it in the cloud or on premise). In addition, a decision may impact other decisions, e.g., deciding to use a public cloud rather than a private one requires deciding on the provider and services. Documenting the different available alternatives and decisions made by the different stakeholders as well as their impact on each other is essential to help manage this complexity.

Iterative development and evolving environments require making decisions throughout the whole development life cycle of the product as well as delaying some of them due to uncertainties in early iterations. For instance, when the development team lacks details about the amount of data that the client’s sensors will provide per second, they will not be able to determine the server requirements necessary to process data in real-time. Furthermore, the HVAC system is subject to a process of continuous improvement, which means that the solution will be monitored and the stakeholders will use that information to revisit/reflect their decisions. For instance, during the development of the HVAC, insufficient performance in extended analytics may indicate the need to update the cloud storage. In all these situations, introducing or changing previously made decisions is of vital importance.

MODA. In order to support an informed decision-making process in building and maintaining complex solutions, all relevant data, used models and particularly their interplay have to be understood and managed. The Models and Data (MODA) framework provides a conceptual structure, where different types of telemetry data (input, output, measured) and external data, as well as different roles of models (descriptive, predictive, prescriptive) are distinguished and where their role in a closed feedback loop are defined [8]. Hereby, a model in a descriptive role describes aspects of an existing, running system, in a predictive role aspects of future behaviour of a system, e.g., by means of simulation or extrapolation, and in a prescriptive role desired aspects of a system to be built.

Variability and Impact Modelling. Feature models (FMs) [16] are typically used to expose the available variability of an artifact that a stakeholder can choose from when building an application. A feature model expresses different features that an artifact encapsulates and describes their optional, mandatory, include, and exclude relationships. Impact models are used to reason about trade-offs when the developer chooses between alternative solutions. When developing a system, a developer would typically select the variant(s) with the best impact on relevant stakeholder goals and system qualities. These impacts can be specified using, e.g., goal models with GRL, which is part of the User Requirements Notation (URN) standard [15], or the NFR framework [7], i* [33], and KAOS [10], but any other technology for multi-criteria decision-making could be used for impact models. There are approaches that allow for automated reasoning of impacts when selecting features [3, 29]. Feature
and impact models are being used to support reuse and making decisions during the software development process [18, 27]. For example, when reusing the Authentication artifact in an application, the developer can use its feature model to select the appropriate variant, e.g., a biometric-based solution, or a password-based solution. The impact model would allow to make a trade-off analysis based on different impacts on high-level goals such as level of security or user convenience. Software Product Lines (SPL) [23] and self-adaptive systems technology [6, 17] help in that regard, and there exist already several multi-objective optimization tools for SPLs [14, 19, 29].

3 MULTI-PLANE MODA

The new idea put forward in this paper is to use a novel conceptual framework entitled Multi-Plane MODA (MP-MODA) to explicitly represent and manage variability of the problem and solution space, impacts of the different variants, and decision points. As such, MP-MODA enables automation and tool support in aid of a multi-criteria decision making process involving different stakeholders within a feedback-driven software development process where feedback cycles aim to reduce uncertainty. Figure 1 presents an overview of MP-MODA.

3.1 MODA Planes

At the center of the framework lies the notion of a MODA plane, which serves as a unit of modularization that addresses a development concern of interest related to the application under development. MODA planes, in contrast to other typical modularization units used in software development such as components [31], are significantly bigger units that encapsulate several variants or alternative ways of addressing a development issue and include one or multiple feedback loops. For example, typical MODA planes might encapsulate technologies such as different operating systems and ways of configuring them, or different cloud providers and service architectures, or different ways of authentication and variants of how to deal with unsuccessful authentication attempts and expiring credentials. However, MODA planes address business objectives just as they address technical objectives. Hence, a MODA plane may also describe business objectives (e.g., market share) and include features for various business opportunities.

The main elements of a MODA plane are shown on the left hand side of Figure 1 (red colors). To render decision making explicit, a MODA plane must define a Variability Model (cf. VM in Figure 1) that exposes the set of variants / configuration options encapsulated within in form of features. The VM should also make feature dependencies explicit, e.g., when one feature requires the presence of another one, or when one feature can not co-exist with another one. A MODA plane must also provide an Impact Model (IM) that lists the non-functional Plane Properties and Qualities of the encapsulated variants and how the different features affect those qualities to allow trade-off analysis when making selections.

A plane encapsulates not only executable code, but also other development artefacts such as models, documentation and sometimes even hardware. In line with Model-Driven Engineering practices [26] and as explained by the MODA framework reviewed in Section 2, the models encapsulated within a MODA plane can play a descriptive, predictive or prescriptive role. Furthermore, each plane defines potentially several feedback loops that define how to monitor the running system and gather Telemetry Data in order to gain insight into the properties of the current configuration of the plane. The data can be used to create descriptive models of the current version of the application for analysis purpose or to update the impact models.

Finally, each MODA plane also comes with Plane Experts, i.e., stakeholders that understand the concern that a plane addresses, that have knowledge about the variants the plane offers and know the involved trade-offs of selecting one variant over another. These experts are depicted as stick men to the left of the planes in Figure 1.

While there will always be MODA planes that are specific to the application that is being developed, we imagine that very quickly a set of standard MODA planes will emerge, encapsulating typical execution environments and platforms, or frameworks. For example, there could be a MODA plane for Java-based development with the Spring framework, or one for Unity-based development for computer games. These MODA planes basically encapsulate libraries and technologies that are already heavily being reused today, but augment them with a variability model, impact model and feedback loop. The standard MODA planes will themselves evolve over time, with new features being added and outdated ones retired, or when new measures are put in place to gather additional telemetry data to improve the feedback loops.

3.2 Inter-Plane Dependencies

To develop an application, multiple planes must be used in combination as shown by the stack of planes with different colors in Figure 1. Each plane exposes its features and configuration options in its variability model, and the consequences of choosing a feature for its stakeholders in the impact model. Each plane gathers telemetry data which drives one or several feedback loops.

When multiple planes are used in conjunction to develop an application, a Decision Space is formed in MP-MODA, visualized by the grey models and grey feedback loop in Figure 1. The experts and developers of each plane have already expressed feature dependencies internal to the plane, e.g., when a feature requires some other feature, or when features are mutually exclusive. Within a decision space, though, there can be additional cross-plane feature constraints, which MP-MODA captures with explicit inter-plane feature dependencies (cf. black lines between VMs of different planes in Figure 1) as well as automatically discovered ones.

Furthermore, there can be indirect feature dependencies, caused by the fact that features in different planes impact the same properties and qualities of the software under development. To this aim, MP-MODA allows developers to define Decision Space Properties (cf. D.S. Props in Figure 1) and express mappings or conversions between the different plane properties and the decision space properties (cf. black lines from IMs to D.S. Props).

Inter-plane feature dependencies, decision space properties and property conversions fuse all variability and impact models together to enable global decision making.
3.3 Making Decisions

The goal of the decision space is to run a global feedback loop in which decisions are made with respect to the current or subsequent software versions by considering feedback from one or several planes, involving experts and stakeholders from all relevant planes.

The descriptive model of the decision space is essentially composed of the union of all variability models of the contained planes and their current configurations. When gathered telemetry data or external data (e.g., gathered knowledge from environment) points to a situation that needs improvement, the fused variability models and impact models of the different planes can be used to explore different configurations and determine whether any of them would result in an improvement. The inter-plane dependencies prevent incorrect configurations from being explored.

When developing a software, a set of objectives is imposed to the development team to satisfy the many requirements imposed by stakeholders as well as legal and business constraints. When decisions are made it is of paramount importance that the objectives with hard constraints are still fulfilled, and the soft objectives are optimized. This is where MP-MODA shines. Whenever feedback from one or several planes suggests to update a plane’s configuration, the inter-plane feature dependencies will determine whether this change requires other changes.

Furthermore, thanks to the connections between features, plane properties and decision space properties, MP-MODA can detect indirect dependencies that can be just as important as the direct dependencies. This is where MP-MODA shines. Whenever feedback from one or several planes suggests to update a plane’s configuration, the inter-plane feature dependencies will determine whether this change requires other changes.

3.4 Motivating Example Revisited

Let us consider how MP-MODA could help in our HVAC motivating example by exploring how decisions made by one stakeholder could impact the remaining stakeholders.

The HVAC company decided to define 3 planes addressing business, software development and operations concerns, respectively. The experts of each plane make decisions among the available options within their attributed plane, and thanks to the documented dependencies between decisions (intra- and inter-planes), they can also assess the impact of their decisions on the decisions made in other planes. For example, the operations experts may realize that the current CPUs are most of the time at 90% usage and consider replacing them with more powerful processing units. While examining the potential options captured in their variability model, MP-MODA alerts them that changing the processors can impact choices made by the experts of the development plane, as it may cause incompatibilities with some of the used software libraries. Thanks to the input provided by MP-MODA, the operations experts decide to meet with the development experts. The former experts were already aware of the computing issue and had a plan to introduce software optimizations to fix the computational cost.
When the development team assesses whether it is better to proceed with their planned optimizations or replace the hardware and therefore update their libraries, MP-MODA also helps them make the best decision by pointing to their intra-plane dependencies. E.g., MP-MODA may point to the fact that updating libraries has an impact on the architecture of the solution. Based on knowledge gathered from previous iterations provided through the feedback loop, they estimate that adapting the architecture can introduce important delays in the delivery of the product, which can further impact decisions made by the business experts about release dates.

By being aware of these inter- and intra-plane dependencies, the development and operations experts were able to make an informed global decision and agree that the best solution is not to update the hardware but to prioritize the work of the development team on the software optimizations.

4 AUTOMATION AND TOOL CHALLENGES

MP-MODA requires significant automation and tool support to be effective in practice. This section briefly points to the most important challenges we identified, and we call on the ASE and SE community at large to intensify research efforts in this direction.

Support Within and Across Planes. Automation and tool support needed within a plane to realize MP-MODA exists for the most part already. For example, many execution platforms or frameworks provide logging support that can be used to gather telemetry data at runtime. Monitoring and data aggregation technology exists that makes it possible to reduce uncertainty or deal with the overwhelming amount of gathered data to focus on the essential information [32]. Techniques such as those promoted by the models@runtime community allow to reason about the current state of the system at a higher level of abstraction [4]. All this should help to capture and quantify the impact of features offered by the plane. Finally, tool support for configuring planes and enacting new configurations, as well as customizing planes with open variability to specific applications faces of course a significant scalability challenge. Techniques such as managing diversity in the decision space exploration can speed up the convergence [2].

First, the framework has to determine the stakeholders and experts that have to be involved in a decision that needs to be made. Sophisticated support is hence needed to discover and keep track of key stakeholders and experts for individual features. Then, tool support enabling effective human-computer interaction is crucial. For example, advanced visualization techniques to help understand configurations and dependencies are needed [20, 30]. Intuitive exploration of what-if scenarios generated using predictive models is highly important [21]. But most importantly, MP-MODA tools need to communicate with all the stakeholders to inform them of the proposed modifications and decisions that need to be made, get their input, engage them in discussions and allow them to approve the final decision. Functionality offered nowadays by modern version control software and collaborative platforms, e.g., GIT pull requests, could be adapted for that purpose. Chat-bots might also be an interesting direction to explore [24], especially for stakeholders from non-technical communities.

REFERENCES


