Global Decision Making Support for Complex System Development

Lola Burgueño, Damien Foures, Benoît Combemale, Jörg Kienzle, Gunter Mussbacher

Global Decision Making Support for Complex System Development

Lola Burgueño
ITIS Software
University of Malaga
Malaga, Spain
lolaburgueno@uma.es

Damien Foures
DDMS
Airbus Group
Toulouse, France
damien.da.foures@airbus.com

Benoit Combemale
ESIR & IRISA
University of Rennes
Rennes, France
benoit.combemale@irisa.fr

Jörg Kienzle
ITIS Software / School of Computer Science
University of Malaga / McGill University
Malaga, Spain / Montreal, Canada
joerg.kienzle@uma.es / mcgill.ca

Gunter Mussbacher
Department of Electrical and Computer Engineering
McGill University / INRIA
Montreal, Canada / Rennes, France
gunter.mussbacher@mcgill.ca

***Abstract*—To succeed with the development of modern and complex systems (e.g., aircrafts or production systems), organizations must have the agility to adapt faster to constantly evolving requirements in order to deliver more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, suppliers, and partners. However, stakeholders do not have sufficiently explicit and systematic support for global decision making, considering the vast decision space and complex inter-relationships. This decision space is characterized by increasing yet inadequately represented variability 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. As a result, one team's design decisions may impose too restrictive requirements on another team. In this paper, we evaluate our understanding of global decision making in the context of complex system development based on a conceptual model which explicitly represents and manages decision spaces including variability and impacts. We have conducted our evaluation by means of an exploratory case study where we interviewed domain experts with an average of 20 years of experience in complex system industries and report the key findings and remaining challenges. In the future, we aim at providing explicit and systematic tool-supported approaches for global decision making support for complex systems.**

***Index Terms*—Global Decision Making, Multi-Stakeholder, Variability, Impact, Requirements, Design.**

I. Introduction

The development of modern and complex systems (e.g., aircrafts or production systems) requires the collaboration of a multitude of specialized teams, each focusing on their domain of expertise. Each team must make decisions with the aim of achieving an optimal result for their task at hand. When teams are working in separate silos, decisions from one team may impose requirements on other teams that are too constraining or even contradictory, hence preventing the organization from reaching a better global result. Global decision making that considers not only the solution space of each specialized team, but also the overall solution space of the organization is required. Furthermore, organizations must have the agility to adapt faster to constantly evolving requirements to succeed with complex system development by delivering more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, suppliers, and partners. Without automated support, teams have to revert to an ad-hoc decision making process for requirements and design that is slow, error-prone, and often favors local knowledge over global, organization-wide objectives. The vast decision space with ever increasing, but inadequately represented variability, and the uncertainty of the impact of decisions on stakeholders and the solution space further exacerbate this decision making problem. While there is a growing body of knowledge around variability management and decision making (see related work in Section VI), the challenges of global decision making in complex system development in an industrial context are not yet well understood.

Based on an initial conceptual model, we evaluate our understanding of global decision making by means of an exploratory case study in the context of complex system development and report our findings in this paper. The conceptual model explicitly represents and manages decision spaces including variability and impacts. The conceptual model is based on the results of a one-week long workshop of experts in model-driven engineering and variability management, which resulted in a vision paper on global decision making with deep variability [31]. The conceptual model presented here formalizes the ideas put forward in that vision paper. The conceptual model is also influenced by the previous experiences of the authors of this paper with complex systems development. In the exploratory case study, we use the conceptual model to guide the questions for a semi-structured interview of four

We are very grateful to the four experts in complex system development who agreed to take part in our interview.

domain experts working for the Airbus company. The interviewed domain experts have at least 15 years and an average of 20 years of experience in complex system development. We follow the interview with an informal debriefing among the authors of this paper, which include a domain expert from the Airbus company.

The following Research Questions (RQs) guided our study:

RQ1: Given the current state of practice for decision making during complex system development, would more advanced global decision support be useful?

We aim to better understand whether more explicit and systematic global decision making as captured in the conceptual model is desired (i.e., usefulness).

RQ2: According to domain expert experience of, and needs for, complex systems development, does our conceptual model accurately capture the elements involved in the decision making process and their relationships?

We aim to study the coverage of our conceptual model: (i) whether the elements that it captures are actually used in practice when making decisions in complex system development; (ii) whether there are elements considered during complex system development that are not present in our conceptual model; and (iii) whether and which elements need to be refined to be applied and used in practice. Answering these points will allow us to determine the completeness, conciseness, and applicability of the proposed conceptual model.

RQ3: What are the challenges that domain experts face when making decisions for complex system development?

We aim to gain a better understanding of said challenges to be able to classify them into categories that will help derive future work needed to address the identified challenges.

Our findings for these three research questions contribute to a better understanding of the problems faced when making decisions during complex system development. Our exploratory case study determined that: (i) there is a clear need for advanced global decision support (RQ1); (ii) while our conceptual model is concise and applicable, it is not complete (RQ2); and (iii) it is challenging to reduce information overload in a cross-discipline environment, to support balanced decisions while prioritizing difficult decisions, to address uncertainty, and to maintain equilibrium of the ecosystem. The overarching challenge is to achieve continuous decision making that evolves over time, built on an integrated, model-based tooling infrastructure (RQ3).

The remainder of the paper is structured as follows. Section II presents a motivating case from Airbus, a multinational aircraft company, followed by the conceptual model capturing our initial understanding for global decision support in Section III. Section IV evaluates this understanding on the basis of the conceptual model and through an exploratory case study where we have interviewed industrial experts. Section V discusses threats to validity. Section VI reviews the most relevant related work to this contribution. Finally, Section VII concludes the paper and draws the main perspectives.

II. Motivating Example: The Case of Airbus

Airbus is evolving in a complex, ambiguous, uncertain and ever fast changing environment, as captured by the VUCA (volatility, uncertainty, complexity, ambiguity) framework [20]. In this context, the development of a new aircraft program through strong *sequential* pillars (from design to manufacturing to services) is no longer sustainable and is highly challenging for the company. With the Digital Manufacturing Design & Services program (DDMS), Airbus decides to provide a digital environment where the design, the manufacturability, and maintainability of their products will be extensively modelled and simulated before detailed design and production start. Through these modelling and simulation efforts, design variants are explored and the requirements for downstream activities are to be determined with greater control. In this context, Airbus decides to investigate new capacities to take continuous and global decisions for requirements and design in complex systems.

Airbus involves various internal stakeholders over the organization entities (e.g., engineering, manufacturing, services). Each entity of the organization requires internal decisions to be made. Those decisions often have inter-dependencies with the other entities of the organization. For instance, over an aircraft program, the engineering entity has an impact on the manufacturing entity. As a concrete example, the size of the aircraft (e.g., wing span and main body volume) – coming from the engineering entity – has an impact on the potential reuse of existing factories, which is taken into account by the manufacturing entity. Another similar example is the size of the aircraft sections (e.g., fuselage, wings, cockpit, engine, tail assembly, and landing gear). The design of these parts is the responsibility of the engineering entity, and has an impact on the possible application of the orbital joints robots from the manufacturing entity.

Some decisions made by Airbus also have an impact beyond the company itself, and require global decision making in the context of an extended enterprise (considering suppliers, manufacturers, and partners like engine providers), and even broader consortia including airlines, airports, (inter)national regulators, etc. For instance, the thrust-to-weight ratio of the engine needs to be discussed among Airbus and the various engine providers. This is an example of a key decision that requires global decision making that must be right the first time to secure program objectives.

An aircraft program is a long process (i.e., about a decade) from the elicitation of the assumptions pack to the targeted delivery rate of aircrafts within the actual manufacturing system. During the implementation of such a program, many decisions need to be regularly re-evaluated to keep pace with a dynamic environment (e.g., governmental regulations, citizens expectations, strategies of competitors, etc.). This requires the decision making approach to be agile, with the ability to easily analyse the impact on the overall program, including already performed activities.

While Airbus has been facing these needs for a while, the accelerating pace and globalization of the world we live in requires the company to introduce an explicit and systematic approach to support the continuous decision making process regarding a given aircraft program. The *explicit approach* aims to capture the past knowledge (e.g., why a specific decision has been made in a given context), while the *systematic approach* aims to leverage this knowledge to provide a methodological context, to be combined with design space exploration tools.

III. Conceptual Model to Support Global Decision Making

On the basis of the aforementioned motivating example, we introduce in this section the main concepts of our conceptual model to support global decision making (cf. Figure 1). We focus on the *main structural concepts and their relationships* for a global decision making framework and leave the definition of concepts for the use of the framework as well as other process-related issues for future work.

We introduce the central notion of *Plane* in Section III-A, discuss how planes are composed into hierarchies of *CompositePlanes* in Section III-B, and then explain how planes form a *DecisionSpace* for trade-off reasoning in Section III-C[1].

A. Plane

At the center of the conceptual model lies the notion of a *Plane*, which serves as a unit of modularization that addresses a concern of interest related to the system under development. *Planes*, in contrast to other typical modularization units used in software and systems engineering such as components [52], 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 *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, *Planes* address other concerns as well, including requirements concerns. For example, a *Plane* may also describe business concerns (e.g., related to market share objectives) and include features for various business opportunities and business strategies that can be pursued.

To render decision making explicit, the *Interface* of a *Plane* must define a *VariabilityModel* (VM) that exposes the set of variants / configuration options encapsulated within in the form of *Features*. The *VariabilityModel* should also make *FeatureDependencies* explicit, e.g., when one *Feature* requires the presence of another one, or when one *Feature* cannot co-exist with another one. These dependencies are expressed using a boolean constraint that must always hold (abstracted in the data type *BooleanFormula*).

[1]Note that our conceptual model is domain-independent and that *Plane* is a generic concept which does not refer to aircrafts. Throughout this paper, we use *Plane* to refer to the concept in the conceptual model and aircraft to refer to the physical entity.

A *Plane* encapsulates not only executable code, but also other development artefacts, such as requirements and design models, documentation, and sometimes even hardware. In our conceptual model, code, models, documentation, and hardware are abstracted with the class *Model*. *Models* can realise *Features* (*realizedFeatures*), for example when a feature is implemented with code or system design. *Models* that deal with feature interactions realise several *Features*, for example when a sequence diagram specifies the priority ordering of the behaviour expressed in two *Features*. In line with Model-Driven Engineering practices [46] and as explained by the MODA framework [12], the models encapsulated within a *Plane* can play three roles: *descriptive* (e.g., aggregates information about an existing system), *predictive* (e.g., performs what-if analyses), and/or *prescriptive* (e.g., specifies the blueprint for a future system).

A *Plane* must also provide an *ImpactModel* (IM) that lists the *PlaneProperties*, i.e., measures of quality of the encapsulated variants. The *PlaneProperty* defines a *Formula* that specifies how the different *Features* of the *Plane* affect the *PlaneProperty* to allow trade-off analysis when making decisions. In the case where *Models* are used to measure a quality, the *Formula* can access elements from these *Models* through the *source* relationship. *Source Models* always play *descriptive* or *predictive* roles. For example, a performance model can be used to predict the performance of a specific deployment of a set of services on a specific cloud infrastructure.

Furthermore, the *ImpactModel* also comes with a set of *PlaneObjectives*, which encode the objectives that the decision makers that are responsible for a *Plane* typically try to pursue, i.e., their requirements. *PlaneObjectives* involve optimizing *PlaneProperties*, which again can be combined using some *Formula*.

Finally, each *Plane* comes with a *StakeholderRole* that is responsible for making decisions about the *Plane*. The assumption is that the person playing that role is a *Stakeholder* that deeply understands the concern that the *Plane* addresses, i.e., that has knowledge about the variants the *Plane* offers and knows the involved trade-offs of selecting one variant over another. Of course, complex *Planes* might require an entire team of stakeholders for effective decision making.

In the case of Airbus, planes can be used to address the different entities (e.g., engineering, manufacturing, services), possibly in a hierarchical way to cope with the structure of the organization and the decision making process through it. The variability model captures the various decision choices in and across the entities. For instance, the variability model of the engineering plane would capture the choice between several aircraft architectures from the engineering entity. The impact model captures the properties of each feature (e.g., raw material quantity) and are related to specific objectives of the plane (e.g., a targeted weight). Decision making over the various choices are the responsibility of stakeholder roles (e.g., aircraft architect).

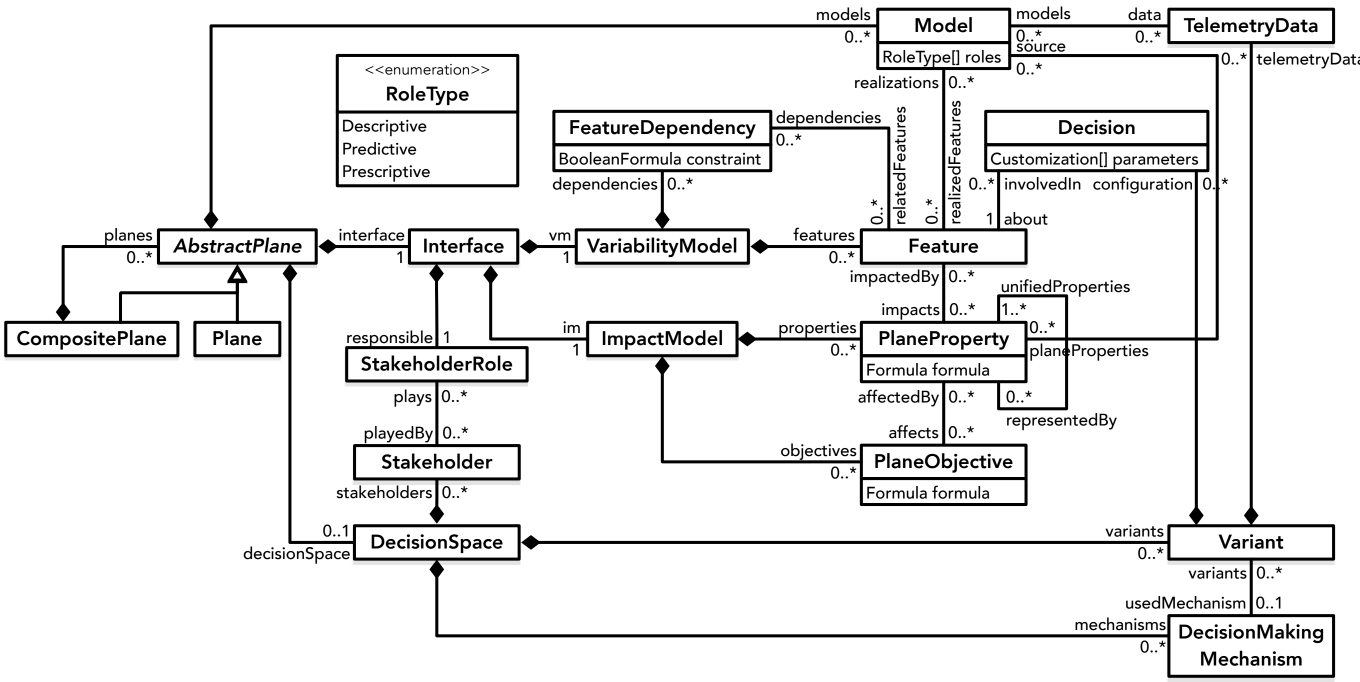

Fig. 1. Conceptual Model to Support Global Decision Making

B. Composite Plane and its Interplane Dependencies

To reason about a complex system, multiple planes must be used in combination and hence form a hierarchy of *CompositePlanes* as shown in the conceptual model. In its *Interface*, each *Plane* exposes its *Features* and configuration options in its *VariabilityModel*, and the consequences of choosing a *Feature* for its *StakeholderRoles* in the *ImpactModel*.

The stakeholders of each *Plane* have already expressed *FeatureDependencies* internal to the plane, e.g., when a *Feature* requires some other *Feature*, or when *Features* are mutually exclusive. Within a *CompositePlane*, though, there can be additional cross-plane feature constraints, which the conceptual model captures with interplane *FeatureDependencies* (i.e., they connect *Features* from different *Planes*).

In the case of Airbus, this enables to capture the aforementioned inter-dependencies between the entities, as well as with suppliers, partners, and broader consortia. For instance, the weight of both the wing and the fuselage (among others) will then impact the overall design of the landing gear (i.e, requirements are imposed on the team working on the landing gear by the design choices of the team(s) working on the wing and fuselage). Beyond the engineering entity, the weight of the aircraft has also impact on partners, such as the engine provider, which eventually impacts the thrust / weight ratio with the objective of avoiding a too high value of excess thrust.

Furthermore, there can be indirect feature dependencies, caused by the fact that *Features* in different *Planes* impact the same *PlaneProperty* of the system. Therefore, the conceptual model allows *PlaneProperties* from different *Planes* to be unified into a corresponding *PlaneProperty* of the *CompositePlane*. For instance, the choices regarding the material of the wing (from its specific plane) and the fuselage (also from its specific plane) may be unified (among others) in a *PlaneProperty* capturing the overall weight of the engineering entity.

Since a *PlaneObjective* of a *CompositePlane* draws from the *PlaneProperties* of the *CompositePlane*, which may now be unified, a *PlaneObjective* may be impacted by all the constituent *Planes* of its *Plane*. Direct and indirect interplane feature dependencies fuse all *VariabilityModels* and *ImpactModels* together to enable global decision making. While a non-composite *Plane* can only declare dependencies among its own features and its *PlaneObjectives* are only affected by its own *PlaneProperties*, *CompositePlanes* can draw from features and properties of all its composed planes.

C. Decision Space

Finally, to enable decision making, a *DecisionSpace* can be attached to a *Plane*. The *DecisionSpace* allows *Variants* of the *Plane* to be assessed based on a *DecisionMakingMechanism* for *Stakeholders*. Concrete *Stakeholders* are mapped to *StakeholderRoles* of the constituent *Planes* of the *Plane*. A *Plane*'s *Variant* represents the feature configuration across the constituent *Planes* of the *Plane*. For each desired *Feature*, a *Decision* indicates that the *Feature* is part of the *Variant*. The *Customization parameters* of a *Decision* allow a *Feature* to be refined (e.g., an Aircraft feature may be configured to have two or three *Landing Gear* features; other attributes of features such as cost or weight that are covered by *PlaneProperties* could also be customized as the *Formula* of a property may use a *Customization parameter*).

To ensure proper modularity, a *Variant* can only make decisions for features that are contained in the *Plane* of its *DecisionSpace* or any of its (sub)planes.

A *DecisonMakingMechanism* represents the employed approach for decision making for a *Variant* (e.g., voting with simple majority, based on simulation results, or based on a predefined formula).

One of the essential benefits of the conceptual model is that it reduces the inherent epistemic uncertainty of decision making during development by means of feedback loops. Whenever possible, a *Plane* therefore gathers *TelemetryData* which drives one or several feedback loops (e.g., from previous versions of the aircraft). The *TelemetryData* allows to gain insight into *PlaneProperties* of a plane. The data can be used to create or update descriptive or predictive models of the system for analysis purposes. In some cases it could even lead to updates of a *PlaneProperty*'s *Formula*.

IV. Evaluation

In this section, we first present our study design in Section IV-A, then elaborate on data collection and data analysis in Sections IV-B and IV-C, respectively, before reporting our findings in Section IV-D.

A. Study Design

To evaluate our initial understanding of the problems faced by companies during decision making for complex system development, we validate the completeness, conciseness, usefulness, and applicability of our conceptual model introduced in Section III. For this, we followed an empirical evaluation through a case study following the guidelines for conducting and reporting case study research in software engineering by Runeson et al. [45], and considering the essential attributes required by Empirical Standards [43].

Case studies are meant to investigate a phenomenon within its real-life context, with the goal to gain a deeper understanding of how and why the phenomenon occurs [18]. In our case, our goal is to study decision making processes, and we do so in the context of the Airbus company and guided by our proposed conceptual model. In particular, we have carried out an exploratory case study [18], since our goal is not to test a pre-conceived hypothesis, but to gain new insights and a better understanding of decision making processes and identify and outline new areas of research, for instance, improved or new methodologies, better traceability, etc.

To address the research questions that guided our study and were presented in Section I, we have collected qualitative data by means of semi-structured interviews with domain experts. In particular, we interviewed four representative stakeholders of the socio-technical system: domain experts for complex systems with at least 15 years of experience and with an average of 20 years of experience in the field. The main background of our interviewees is not only technical, but also managerial. Their duties involve leading and planning, organizing, strategizing, and solving problems, hence making decisions. These experts were not only reporting on their experience at Airbus but their whole career.

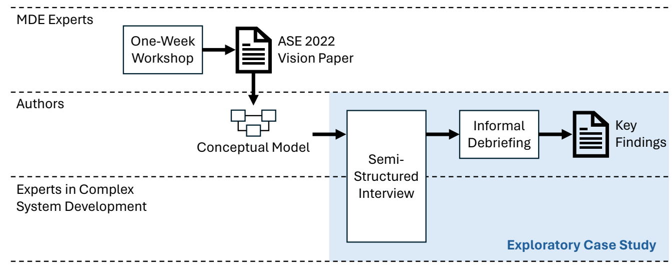

Fig. 2. Design of the Exploratory Case Study

The goal of the interviews was to gain a better understanding of the viewpoints of the interviewees on the required support for complex decision making based on their decades-long experience in complex system industries.

In case studies, data collection is performed with respect to a unit of analysis. In our case, the unit of analysis is the interview session with the four experts. During the interview, the interviewees contributed with their experience, perspectives and insights. We gathered data from the responses regarding the completeness, conciseness, applicability, and usefulness of the conceptual model we propose. We believe that interviewing domain experts from the Airbus company is an appropriate context to study decision making processes since Airbus is a multinational company with different stakeholders who need to make decisions during complex systems development taking into account local and global objectives, strategies, different departments, suppliers, etc. This choice is also opportunistic as one of the authors works at the company and some of the authors had previous collaborations with Airbus. Finally, the interviewed domain experts are leveraging on their entire experience at Airbus and the previous companies they worked at.

Figure 2 summarizes our study design. Three of the authors co-organized a one-week long workshop of experts in model-driven engineering, which resulted in a vision paper on global decision making with deep variability [31]. The conceptual model presented in Section III formalizes the ideas put forward in the vision paper, while also considering the previous experiences of the authors of this paper with complex systems development. The questions for the semi-structured interview are guided by the conceptual model. This is followed by an informal debriefing of the authors, which include an expert from the Airbus company. The outcome are the key findings presented in Section IV-D.

B. Data Collection

The data collection was performed through a group interview allocated in a two-hour session arranged by our author working at Airbus. We interviewed four domain experts who were opportunistically selected by the author who works at Airbus. They were selected based on the fact that they have different roles that require decision making as well as extensive experience in several companies in complex systems development. The interview was led by another author. In total, four authors attended the interview and took extensive notes.

The interview session with the domain experts was divided into a number of phases:

- *(15 mins)* During the first phase, the leading researcher(s) presented the objectives of the interview and, for informative purposes, the conceptual model. They also explained how the data from the interview was going to be used.
- *(100 mins)* The second phase was a semi-structured interview, where the dialog between the researchers and the interviewees was guided by a set of questions that were planned in advance. The researcher leading the interview had the freedom to adapt the order of the questions depending on the progression of the interview.
  - *(80 mins)* First, predominantely open-ended questions were asked to explore our understanding of complex decision making in terms of whether (i) our conceptual model is complete, concise, and applicable (i.e., the concepts and relationships it contains are enough to capture and represent global decision making during complex system development, i.e., there are no missing concepts and/or relationships in the conceptual model); (ii) our conceptual model is useful (i.e., it could help the interviewees in their daily jobs and the company in general).
  - *(20 mins)* This was followed by a general discussion session where the interviewees had the freedom to add and discuss anything they liked. The target of this session was to collect general feedback.
- *(5 mins)* Towards the end of the interview, the major findings were discussed and verified in order to avoid misunderstandings.

As mentioned above, we used predominantely open-ended questions to allow the participants to describe their experiences. To make sure we covered the entire conceptual model, we created questions for each concept and relationship.

As an example, Table I shows some of the questions that our interviewees were asked. Q1 and Q2 are examples of questions targeted at discovering the challenges that are currently being faced related to decision making, as well as the need for better decision making support. Q3 and Q4 investigate whether the way we calculate the values for *PlaneProperties* using a formula is realistic. Q5 and Q6 inquire whether the way our conceptual model captures *Stakeholders* and relates them to *StakeholderRoles* is sufficient and useful. Finally, Q7 to Q9 are about when *Decisions* are monitored and revisited. The complete list of questions is on our Github Repository[2].

C. Data Analysis

The language spoken during the interview was English. Due to confidentiality policies, we did not record the interview. However, four authors were present and took extensive notes. The notes of all four researchers were collected and merged

[2]https://github.com/atenearesearchgroup/global-decision-making

TABLE I
QUESTIONS ASKED TO THE INTERVIEWEES

Q#	Question
Q1	Do you use tools when making decisions? (Dashboards, Excel sheets, proprietary tools, etc.)
Q2	When there is a decision that involves several people and they need to meet, how are these meetings organized (i.e., physically, who calls for a meeting)? How many people more or less need to meet? How long does it take?
Q3	In your opinion, to which extent are decisions based on evidence or personal feelings/intuition? In case they are based on feelings, could the reasoning be expressed using evidence?
Q4	We assume that you collect and analyse data to help you check whether your objectives are being met (e.g., KPIs). To what extent is the calculation of these indicators automated?
Q5	When you need to make a decision that affects other stakeholders (either within the team, the company or outside), do you always know precisely who you need to contact to get the information that allows you to make the decision? If yes, is that knowledge captured explicitly somewhere?
Q6	Do you need to keep track of people who made certain decisions and the input they used to make their decision? If so, is this information stored explicitly?
Q7	Are decisions periodically revisited?
Q8	Based on feedback (from people or collected data), do you continuously monitor and revisit your decisions to ensure that they are still valid? If so, to what extent is this automated?
Q9	What are common triggers that make you revisit a previously made decision? (Is it data? Feedback from people? New hires? Changes in technology?)

prior to the analysis. This was of particular importance since during the merging process, new insights could potentially emerge.

This was followed by a face-to-face meeting in which all the authors went through the notes and discussed, first for each question and second in general, what the findings were.

Once the data was analysed and the findings written down, we shared the outcome with the company for verification and approval.

D. Findings

RQ1: Given the current state of practice for decision making during complex system development, would more advanced global decision support be useful?

One of the interviewees explicitly mentioned the current lack of cross-discipline decision making in that sometimes decisions are made considering a single discipline (i.e., *Plane*) to keep the right benefit/risk balance. Our conceptual model is inherently cross-discipline.

The interview revealed that there is a need to be able to determine in which order different entities (e.g., engineering, manufacturing, services) (i.e., the *Planes*) should be used for optimal decision making, taking priority, cost, time, uncertainty, and maturity into account. Furthermore, support is needed to determine which decisions have priority over other decisions (i.e., which decisions need to be made now and which ones can be delayed). The key is to focus on uncertain or hard areas first.

Additionally, an interviewee said that they find it challenging "to propagate knowledge and information from risk reviews/milestone meetings/forums etc. without over-soliciting a too large population. This currently depends heavily on the level of networking of the person that take up the role of messengers", i.e., information / knowledge distributors. Our conceptual model makes it possible to link the collected data (i.e., *TelemetryData*) to models, and attach those models to features and impacts. This makes it possible to show the impacts, models (aggregated data), and even the raw data, if needed, to the stakeholders who need to make a decision involving those features.

Furthermore, the interviewees pointed out that there is a fundamental need to filter unimportant data or aggregate data to not overwhelm decision makers. The goal is to "move away from requirements-based systems engineering and move towards model-based systems engineering" (i.e., make decisions based on a clear representation of the system). The interviewees mentioned that, 10 to 15 years ago, they had what is called requirements-based engineering (i.e., taking decisions directly as one is checking whether something is covering a list of requirements). Now, model-based systems engineering helps determine more factually whether requirements are covered or not. Going beyond decision making within a company, an interviewee remarked that there is a need for models of collaboration / competition mechanisms for exploration to understand their importance during decision making in the ecosystem of companies involved in complex system development.

Finally, while a multitude of useful tools already exist for decision making (e.g., simulation tools, data gathering tools, dashboards), there is a need to integrate them for decision making.

> **Answer to RQ1.** There is a clear need for advanced global decision support that is cross-discipline and reduces information overload while prioritizing uncertain or hard areas. This support needs to be built on an integrated, model-based tooling infrastructure.

RQ2: According to domain expert experience of, and need for, complex systems development, does our conceptual model accurately capture the elements involved in the decision making process and their relationships?

As explained above, we had prepared questions for each and every concept and relationship in our conceptual model.

Through the interview it became clear that the conceptual model is applicable. The interviewees confirmed that there are *TelemetryData*, different *Stakeholders* and *StakeholderRoles*, different *Planes* (i.e., airlines, suppliers, departments, etc.), that they use a plethora of descriptive and prescriptive *Models* as well as more and more predictive *Models*, and that they set *Objectives*.

One interviewee mentioned that making a decision is "not always a catalogue of options, but sometimes a continuous design space, like choosing the dihedral angle of the wings of an aircraft". Our conceptual model supports, by means of the *VariabilityModel*, choosing among a set of discrete options through the selection of *Features*, and choosing of values from a continuous space through the *Customization parameters* of a *Decision*.

Furthermore, the interviewees mentioned that they need to determine the order in which decisions should be made when there are dependencies across different organization entities. These dependencies are what our conceptual model can capture directly using *FeatureDependencies* and indirectly with unified *Properties* of a *Plane's ImpactModel*.

Everything considered, the interview confirmed that all our model elements are necessary elements, i.e., all of them are being used by the experts for decision making during complex system development.

However, the interview also revealed that our conceptual model did not cover the following aspects:

Time: One of the interviewees mentioned that "Sometimes a decision is not about choosing among several alternatives, but doing something *at a given time*, i.e., now vs. later." Our conceptual model should capture the time dimension explicitly, i.e., at what time decisions have to be / are made. This will enable reasoning about trade-offs between cost / time / uncertainty.

Decision Provenance: The interviewees emphasized the importance of provenance. Our conceptual model could be extended with additional support, e.g., to not only capture *who made a decision*, but also *who has been contributing towards a decision* by providing information or advice. It would also make sense to capture explicitly the target values for objectives that influence the decisions that are made.

Stakeholder Details: The information about stakeholders and stakeholder roles needs additional details. For example, the interviewees mentioned several times that in big companies there is natural turnover, i.e., people occupy different positions over time. It would therefore make sense to keep track of the history of stakeholder assignments to stakeholder roles. Similarly, decision power is given to people with certain qualifications for accountability reasons, or because it is required by law. Our conceptual model needs to be extended to understand these concepts so that it can provide support to ensure that the right person / people make the decisions.

Uncertainty: During the interview, we asked our interviewees whether they take into account the uncertainty associated with decisions. They said that uncertainty should not be neglected, and if possible it should be quantified. However, in case of limited impact of the decision, people limit the use of uncertainty due to the cost/benefit balance that it adds. On the other hand, when safety properties are at stake, the way of working is different and requires uncertainty management. It would therefore be highly beneficial to address uncertainty explicitly in the conceptual model.

Answer to RQ2. Our conceptual model is concise, i.e., it does not contain unnecessary elements, but not complete, i.e., there is additional information that is needed or would be useful for decision making in complex systems. In particular, the time dimension for decisions, decision provenance, and stakeholder involvements, as well as uncertainty related to telemetry data and properties need to be considered.

RQ3: What are the challenges that domain experts face when making decisions for complex systems development?

Information Overload: It is a key challenge to have the right information at the right time while not overwhelming decision makers with information. The interviewees likened this to the *dark cockpit* approach for pilots, where issues are presented in a prioritized fashion for immediate attention. They claimed that there is not necessarily a need to make decisions earlier, but it is more about making decisions at the right time. While sometimes required data is not available, often too much data is available. Hence, interviewees stressed that it is crucial to investigate data filtering or data aggregation so as to reduce information overload. They said that this is more and more important as complexity increases and more and more data is generated, because more and more things are digital and connected. While at lower levels of decision making many options may have to be evaluated, this is not possible at higher levels of decision making, as the experts involved in those decisions have a lot of work to do and will not be able to evaluate many options. Our interviewees stated that the challenge is rather to select the right top options (a maximum of three) to present to the higher-level experts and be able to explain why.

According to our interviewees, it is important to determine in which order *Planes* should be used for optimal decision making, taking priority, cost, time, uncertainty, and maturity into account. It is also important to provide priority analyses that help identify those decisions that one needs to make now or one can delay. The interviewees believe that the benefit of advanced decision support is in providing assistance to human decision makers, not to fully automate decision making (at least in the near future). This is a challenge in an environment that requires multi-disciplinary analytics / optimization. However, the interviewees state that these barriers need to be broken down. At the same time, it is a challenge to determine whether enough data is available to make a decision (i.e., was enough elaboration done to make a decision? how is a decision made?). The interviewees remarked that (a) there is very limited place for judgement without data in the context of complex system development, and (b) while smaller companies give themselves the right to fail, this is not that easy in a big company.

Balanced Decisions including Uncertainty: It is a challenge to determine the right balance between priority, cost, time, uncertainty, and maturity. The interviewees stated that there are decisions that are very expensive or for which it is even impossible to quantify the impact. Typically, the cost of decisions is monitored. While it would be nice to have more assistance, this also implies more cost, which they need to keep under control. To keep cost under control, assumptions must be made. Hence, there is a need to continuously support the monitoring of decisions with regards to the data collected and the possible evaluation of the associated risk.

Depending on the decisions, the impact can be quantified or not. The interviewees remarked that impact quantification might be easier for technical decisions, but impossible for less technical decisions (e.g., raising the salary of a person). It is not easy to capture and propagate uncertainty in a product development activity when making decisions that involve humans. There are attempts to apply uncertainty quantification in these cases, but the interviewees pointed out the credibility of the figures can be challenged: "People end up acting and reacting to what is happening, which eventually leads to reduced uncertainty".

Our interviewees pointed out that the key is to determine the right people and the right data needed to make the decision. They said that "if one uses more time to make a decision, one has more maturity, but in the end there is a time to make a decision". There are trade-offs between efficiency and effectiveness. The interviewees highlighted that some criteria are needed to choose the right methodologies for the right decisions. There are places where decisions are taken with the quantitative assessment of uncertainties, but this is adding a lot of cost since it adds an additional dimension.

Ecosystem Equilibrium: It is a challenge to determine how to reach an equilibrium in the ecosystem of companies involved in complex system development (e.g., there is no balance if an aircraft manufacturer is profitable, but airlines are not). The way decisions are reached need to be considered. Some decisions have a designated person (i.e., military-like), for others its more collaborative (i.e., "agree", "I can live with this" voting). Nevertheless, the interviewees highlighted that having everyone involved in a decision may not be ideal, as it could lead to a lack of accountability.

"Sometimes time constraints make it impossible to consult others. However, providing opinions and making decisions influences the evolving ecosystem". The interviewees highlighted that it is important to identify and acknowledge three things: (i) to whom the decision is important / whom the decision will impact, (ii) to make sure that the person making the decision sees themselves as the most appropriate person to do so, and (iii) assuming the consequences of giving this opinion.

Tooling: It is a challenge to make a tool for the framework useful. The interviewees pointed out that many tools currently already exist that help in decision making (for simulation, for gathering data that is passed to algorithms, there are dashboards, etc.) but they are not integrated for decision making. "In the end the decision is human-based and the tool needs to be able to help the human understand what the problem is". A tool needs to capture the flow of information and structure the huge amount of telemetry data, which is difficult to capture in a tool so that decision makers will not be more and more overwhelmed.

Continuous Decision Making: The interviewees pointed out that one of the dilemmas is that they are not taking decisions in a static environment. They are taking decisions in a dynamic adaptive system and it is a challenge to reach continuous decision making. This involves, e.g., determining which models are needed to make decisions. This also involves periodically, or even continuously, reexamining decision that have been made. The interviewees remarked that while the outcomes of decisions are monitored (i.e., are we reaching what was decided (the requirements), e.g., through testing/integration activities?) and mechanisms are in place if this is not the case, what led to certain decisions is not often revisited. Hence it is difficult to learn from the past to improve the decision making process itself. Continuous decision making also requires knowledge and information to be propagated from risk reviews, milestone meetings, forums, etc. to the right stakeholder. This currently depends heavily on the right kind of people that take up the role of messengers (information / knowledge distributors). In continuous decision making, this information flow needs to occur more systematically. Overall, continuous decision making is an overarching challenge to which the other challenges contribute, i.e., dealing with information overload, balancing decisions including uncertainty, and tooling issues as well as how to keep a continually balanced ecosystem.

> **Answer to RQ3.** It is a challenge to reduce information overload while making balanced decisions that address uncertainty and maintain ecosystem equilibrium to achieve continuous decision making based on integrated decision making tools.

V. Threats to validity

Construct Validity. This refers to the extent to which the case study measures what it is intended to measure. In our case, a potential threat is that the interview questions are not well-defined or they are ambiguous in their interpretation, in which case the participants could provide useless or inaccurate information. The option of running a pilot to mitigate this threat was not possible given the limited number and availability of the participants from Airbus. Hence, to mitigate this threat, before the interview, three of the authors carefully looked through the questions to make sure that they were concise and not misleading.

Another threat is that the interviewer's bias may potentially influence the responses of the participants. To mitigate this threat, the author running the interview did not actively work on the creation of the conceptual model that was used to guide the questions. Furthermore, during the interview, the other three authors were present and could have intervened, if needed.

Internal Validity. This refers to whether an empirical evaluation is sufficient evidence to support the claim, and whether there were issues during that evaluation that may affect the reliability of the findings.

In our case, the empirical interview covered the conceptual model that has been defined before the interview, and scoped the evaluation to completeness, conciseness, usefulness, and applicability. While the interview with domain experts brought several challenges and additional concepts needed in the conceptual model to light, we cannot assure that other concepts may not be necessary beyond the needs expressed by the experts. To mitigate this as much as possible, we selected high profile domain experts with strong experience, and instead of interviewing one person we interviewed four.

Furthermore, another threat could be that the participants may have provided limited answers to protect the intellectual property of the company. To mitigate this threat and make the participants comfortable during the interview, we did not record the interview and we agreed that, once this paper is written, it would be submitted to the company to ensure that their policies are not violated and their competitive advantage is not disclosed.

External Validity. This refers to the extent to which the findings can be generalized. We carefully selected high profile domain experts with an average of 20 years of expertise in several companies on complex system development. We are confident this is representative of modern systems. However, we could only interview four experts who are currently all in the same company (Airbus). This may influence the generalization of the results with respect to other backgrounds and other settings, even though the interviewees were drawing from their experiences from their whole careers, and not just from the Airbus company.

Conclusion Validity. This aspect deals with the degree to which the conclusions reached are reasonable based on the data collected. Researcher bias, for example, can greatly impact conclusions reached and it is possible that an author interprets the answers of an interviewee in a biased manner. To mitigate this threat, all our findings were elaborated among the four academic authors that participated in the interview, and in the end we checked the conclusions with the author from Airbus.

VI. Related Work

Our conceptual model of global decision making support for requirements and design in complex systems encompasses modelling of variability and impacts as done in software product lines (SPL), modelling and exploiting of feedback loops in self-adaptive systems, as well as dealing with uncertainty and decision making. The most relevant works are summarized in this subsection.

A. Variability and Impacts

Feature models (FMs) [28] are typically used to expose the available variability of an artifact that a stakeholder can choose from when building an application. They are heavily used in Software Product Lines (SPL) [42]. 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 tradeoffs 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 [26], or the NFR framework [10], i* [58], and KAOS [14], but any other technology for multi-criteria decision-making could be used for impact models. Feature models can also be augmented with feature attributes, which can be used to capture qualities and non-functional requirements [3].

Approaches exist that allow for automated reasoning of impacts when selecting features [4], [48], and there exist already several multi-objective optimization tools for SPLs [21], [35], [48]. Feature and impact models in particular are also used to support reuse and tradeoff analysis in software design [2], [17], [32] and other domains such as human value analysis [41]. Furthermore, research efforts exist that investigate how to handle interactions among features [16], [49].

B. Feedback in Self-Adaptive Systems

The definition of a feedback loop has been investigated in the context of Self-Adaptive Systems [9], with a field mature enough to provide time-honored patterns such as the MAPE-K loop [30]. The MAPE-K loop provides a pattern to implement a feedback loop in terms of four main functions (**M**onitoring, **A**nalysis, **P**lanning, and **E**xecution) and a common **K**nowledge. A recent framework [12] for the combined use of data and models describes the various roles that models of all types (e.g., engineering models, scientific models, and machine learning models) can play (i.e., descriptive, predictive and prescriptive roles), and exemplifies them in the context of the MAPE-K loop. In addition to patterns, reference architectures (three-layer architecture [33], MORPH [6], PLASMA [53]) and frameworks (Executable Runtime Megamodels [56], DCL [38], ActivFORMS [25], Ponder2 [55]) provide abstractions that help the design of self-adaptive systems. Some frameworks provide abstractions for the implementation of the feedback loop and the decision process. For example, the Ponder2 framework [55] provides abstractions to define policy-based self-adaptive systems and allows to configure and control managed elements using the PonderTalk language. While PonderTalk eases the management of the policies applied in Ponder2 systems, it does not abstract the policies definition and adaptation implementation from its users. Executable Runtime Megamodels [56] offer to abstract the implementation of feedback loops by defining them explicitly at a higher level of abstraction. Similarly, Dynamic update of Control Loops (DCL) [38] abstracts the implementation of the feedback loop by associating elements of the system goal-model to feedback loop functions. However, in these cases, the framework's users remain in charge of the implementation of the feedback loop, including trade-off reasoning, which is a considerable burden.

C. Feedback at Design Time

Recent work has started investigating how to feed back usage or production data directly into the IDE [11], [57] for improved decision making by the developer at design time.

D. Uncertainty and Decision Making

Different kinds of uncertainty [54] have been identified and classified in the literature. The authors also analyzed how uncertainty is represented in software models and used in the context of model-based software engineering (MBSE). In many occasions, belief uncertainty is expressed by probabilities (interpreted in Probability theory [15] or in Uncertainty theory [34]), possibilities (in Fuzzy set theory [60]), plausibilities (in the Dempster–Shafer theory of evidence [47]), or opinions (in subjective logic [27]).

There exist several decision-making techniques and tools to make the right decisions when stakeholders with different expertise are involved [1], [8]. Some of the more commonly used techniques are the analytic hierarchical process (AHP) [59], the WinWin approach [5], or the Kano model [29], among many others [1], [24]. These techniques are mainly employed as prioritization methods for requirements engineering.

Several works deal with uncertainty and probabilities in SPLs for decision-making at different stages of the SPL process, including requirements elicitation [19], [50], [51], SPL evolution [7], product configuration [13], [36], [37], [39], [40], [44], and automated analysis of feature models [22], [23].

VII. Conclusions and Perspectives

In this paper, we report on the challenges faced for global decision making support during complex system development, validated through an exploratory case study involving an interview of four high-profile domain experts in the field. We formalize our initial understanding in a conceptual model capturing decision spaces including variability and impacts. We use this conceptual model as support of the interview and through the interview evaluate our understanding of the challenges and the completeness, conciseness, usefulness, and applicability of the conceptual model.

Our findings contribute to a better understanding of the problems faced when making decisions during complex system development: (a) Advanced global decision support is clearly needed; (b) Our conceptual model is concise and applicable but not complete; and (c) Several challenges need to be addressed in future work. Consequently in the near term, the goal is to investigate how to reduce information overload in a cross-discipline environment and how to reach balanced decisions while prioritizing difficult decisions, addressing uncertainty, and maintaining ecosystem equilibrium. The long-term goal of this work is to provide a framework that supports a more explicit and systematic approach for global decision making in complex system development. Corresponding tool support then enables the ability to continuously evaluate decision objectives regarding possibly changing, complex, and interdependent requirements for continuous decision making.

Acknowledgment

This work was partially funded by the Spanish Government (Ministerio de Ciencia e Innovación–Agencia Estatal de Investigación) under projects PID2021-125527NB-I00, TED2021-130523B-I00; by Junta de Andalucía under project QUAL21 010UMA; and by the Universidad de Málaga.

References

[1] P. Achimugu, A. Selamat, R. Ibrahim, and M. N. Mahrin. A systematic literature review of software requirements prioritization research. *Information and Software Technology*, 56(6):568–585, 2014.

[2] O. Alam, J. Kienzle, and G. Mussbacher. Concern-oriented software design. In *MODELS'13*, LNCS 8107, pages 604–621. Springer, 2013.

[3] D. Benavides, S. Segura, and A. Ruiz-Cortés. Automated analysis of feature models 20 years later: A literature review. *Information Systems*, 35(6):615–636, 2010.

[4] D. Benavides, P. Trinidad, and A. Ruiz-Cortés. Automated reasoning on feature models. In *CAiSE'05*, pages 491–503. Springer, 2005.

[5] B. Boehm and H. Kitapci. *The WinWin Approach: Using a Requirements Negotiation Tool for Rationale Capture and Use*, pages 173–190. Springer, 2006.

[6] V. Braberman, N. D'Ippolito, J. Kramer, D. Sykes, and S. Uchitel. Morph: A reference architecture for configuration and behaviour self-adaptation. In *Int. Workshop on Control Theory for Software Engineering*, CTSE 2015, page 9–16. ACM, 2015.

[7] L. Burgueño, J. M. Horcas, and J. Kienzle. Development and evolution of software product lines driven by stakeholder beliefs. In *SPLC'23*, pages 34–40. ACM, 2023.

[8] L. Burgueño, P. Muñoz, R. Clarisó, J. Cabot, S. Gérard, and A. Vallecillo. Dealing with belief uncertainty in domain models. *ACM Trans. Softw. Eng. Methodol.*, 32(2):31:1–31:34, 2023.

[9] B. H. C. Cheng, R. de Lemos, H. Giese, P. Inverardi, J. Magee, J. Andersson, B. Becker, N. Bencomo, Y. Brun, B. Cukic, G. Di Marzo Serugendo, S. Dustdar, A. Finkelstein, C. Gacek, K. Geihs, V. Grassi, G. Karsai, H. M. Kienle, J. Kramer, M. Litoiu, S. Malek, R. Mirandola, H. A. Müller, S. Park, M. Shaw, M. Tichy, M. Tivoli, D. Weyns, and J. Whittle. *Software Engineering for Self-Adaptive Systems: A Research Roadmap*, pages 1–26. Springer, 2009.

[10] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos. *Non-Functional Requirements in Software Engineering*. Springer, 2000.

[11] J. Cito, P. Leitner, M. Rinard, and H. C. Gall. Interactive production performance feedback in the ide. In *ICSE'19*, pages 971–981, 2019.

[12] B. Combemale, J. Kienzle, G. Mussbacher, H. Ali, D. Amyot, M. Bagherzadeh, E. Batot, N. Bencomo, B. Benni, J.-M. Bruel, J. Cabot, B. H. Cheng, P. Collet, G. Engels, R. Heinrich, J.-M. Jezequel, A. Koziolek, S. Mosser, R. Reussner, H. Sahraoui, R. Saini, J. Sallou, S. Stinckwich, E. Syriani, and M. Wimmer. A hitchhiker's guide to model-driven engineering for data-centric systems. *IEEE Software*, 38(4):71–84, 2021.

[13] K. Czarnecki, S. She, and A. Wasowski. Sample spaces and feature models: There and back again. In *SPLC'08*, pages 22–31. IEEE Computer Society, 2008.

[14] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Science of Comp. Prog.*, 20:3–50, 1993.

[15] B. De Finetti. *Theory of probability: A critical introductory treatment*, volume 6. John Wiley & Sons, 2017.

[16] M. Duran and G. Mussbacher. Investigation of feature run-time conflicts on goal model-based reuse. *Information Systems Frontiers*, 18(5):855–875, 2016.

[17] M. B. Duran, G. Mussbacher, N. Thimmegowda, and J. Kienzle. On the Reuse of Goal Models. In *SDL'15*, volume 9369 of *Lecture Notes in Computer Science*, pages 141–158. Springer, November 2015.

[18] S. Easterbrook, J. Singer, M. D. Storey, and D. E. Damian. Selecting empirical methods for software engineering research. In *Guide to Advanced Empirical Software Engineering*, pages 285–311. Springer, 2008.

[19] M. Famelis, J. Rubin, K. Czarnecki, R. Salay, and M. Chechik. Software product lines with design choices: Reasoning about variability and design uncertainty. In *MODELS'17*, pages 93–100, 2017.

[20] Forbes, 2024. https://www.forbes.com/sites/hbsworkingknowledge/2017/02/17/vuca-2-0-a-strategy-for-steady-leadership-in-an-unsteady-world.

[21] C. Henard, M. Papadakis, M. Harman, and Y. Le Traon. Combining multi-objective search and constraint solving for configuring large software product lines. In *ICSE'15*, volume 1, pages 517–528, 2015.

[22] R. Heradio, D. Fernández-Amorós, C. Mayr-Dorn, and A. Egyed. Supporting the statistical analysis of variability models. In *ICSE'19*, pages 843–853. IEEE / ACM, 2019.

[23] J. M. Horcas, J. A. Galindo, R. Heradio, D. Fernández-Amorós, and D. Benavides. A monte carlo tree search conceptual framework for feature model analyses. *J. Syst. Softw.*, 195:111551, Jan. 2023.

[24] A. Hudaib, R. Masadeh, M. H. Qasem, A. Alzaqebah, et al. Requirements prioritization techniques comparison. *Modern Applied Science*, 12(2):62, 2018.

[25] M. U. Iftikhar and D. Weyns. Activforms: Active formal models for self-adaptation. In *Int. Symposium on Software Engineering for Adaptive and Self-Managing Systems*, SEAMS 2014, page 125–134. ACM, 2014.

[26] International Telecommunication Union (ITU-T). Recommendation Z.151 (10/12): User Requirements Notation (URN) - Language Definition, approved October 2012.

[27] A. Jøsang. *Subjective Logic - A Formalism for Reasoning Under Uncertainty*. Artificial Intelligence: Foundations, Theory, and Algorithms. Springer, 2016.

[28] K. Kang, S. Cohen, J. Hess, W. Novak, and S. Peterson. Feature-oriented domain analysis (FODA) feasibility study. Technical Report CMU/SEI-90-TR-21, SEI, Carnegie Mellon University, November 1990.

[29] N. Kano. Attractive quality and must-be quality. *Journal of the Japanese society for quality control*, 31(4):147–156, 1984.

[30] J. Kephart and D. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[31] J. Kienzle, B. Combemale, G. Mussbacher, O. Alam, F. Bordeleau, L. Burgueño, G. Engels, J. Galasso, J. Jézéquel, B. Kemme, S. Mosser, H. A. Sahraoui, M. Schiedermeier, and E. Syriani. Global decision making over deep variability in feedback-driven software development. In *ASE'22*, pages 178:1–178:6. ACM, 2022.

[32] J. Kienzle, G. Mussbacher, P. Collet, and O. Alam. Delaying decisions in variable concern hierarchies. In *GPCE'16*, pages 93–103, 2016.

[33] J. Kramer and J. Magee. Self-managed systems: an architectural challenge. In *Future of Software Engineering*, pages 259–268, 2007.

[34] B. Liu. *Uncertainty Theory*, pages 1–79. Springer, 2010.

[35] J. Martinez, D. Strüber, J. Horcas, A. Burdusel, and S. Zschaler. Acapulco: An extensible tool for identifying optimal and consistent feature model configurations. In *SPLC'22*, July 2022.

[36] J. Martinez, T. Ziadi, R. Mazo, T. F. Bissyandé, J. Klein, and Y. L. Traon. Feature relations graphs: A visualisation paradigm for feature constraints in software product lines. In *VISSOFT'14*, pages 50–59. IEEE Computer Society, 2014.

[37] R. Mazo, C. Dumitrescu, C. Salinesi, and D. Diaz. Recommendation heuristics for improving product line configuration processes. In *Recommendation Systems in Software Engineering*, pages 511–537. Springer, 2014.

[38] H. Nakagawa, A. Ohsuga, and S. Honiden. Towards dynamic evolution of self-adaptive systems based on dynamic updating of control loops. In *ACSOS'12*, pages 59–68, 2012.

[39] A. Nöhrer and A. Egyed. C2O configurator: a tool for guided decision-making. *Autom. Softw. Eng.*, 20(2):265–296, 2013.

[40] J. A. Pereira, P. Matuszyk, S. Krieter, M. Spiliopoulou, and G. Saake. Personalized recommender systems for product-line configuration processes. *Comput. Lang. Syst. Struct.*, 54:451–471, 2018.

[41] H. Perera, G. Mussbacher, W. Hussain, R. Ara Shams, A. Nurwidyantoro, and J. Whittle. Continual human value analysis in software development: A goal model based approach. In *RE'20*, pages 192–203, 2020.

[42] K. Pohl, G. Böckle, and F. J. van der Linden. *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer, 2005.

[43] P. Ralph, S. Baltes, D. Bianculli, Y. Dittrich, M. Felderer, R. Feldt, A. Filieri, C. A. Furia, D. Graziotin, P. He, R. Hoda, N. Juristo, B. A. Kitchenham, R. Robbes, D. Méndez, J. S. Molléri, D. Spinellis, M. Staron, K. Stol, D. A. Tamburri, M. Torchiano, C. Treude, B. Turhan, and S. Vegas. ACM SIGSOFT empirical standards. *CoRR*, abs/2010.03525, 2020.

[44] J. Rodas-Silva, J. A. Galindo, J. García-Gutiérrez, and D. Benavides. Selection of software product line implementation components using recommender systems: An application to wordpress. *IEEE Access*, 7:69226–69245, 2019.

[45] P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. *Empir. Softw. Eng.*, 14(2):131–164, 2009.

[46] D. Schmidt. Guest x-editor's Introduction: Model-Driven Engineering. *Computer*, 39(2):25–31, feb 2006.

[47] G. Shafer. *A mathematical theory of evidence*, volume 42. Princeton university press, 1976.

[48] N. Siegmund, M. Rosenmüller, M. Kuhlemann, C. Kästner, S. Apel, and G. Saake. Spl conqueror: Toward optimization of non-functional properties in software product lines. *Software Quality Control*, 20(3-4):487–517, Sept. 2012.

[49] L. R. Soares, P.-Y. Schobbens, I. do Carmo Machado, and E. S. de Almeida. Feature interaction in software product line engineering: A systematic mapping study. *Information and Software Technology*, 98:44–58, 2018.

[50] A. Sree-Kumar, E. Planas, and R. Clarisó. Extracting software product line feature models from natural language specifications. In *SPLC'18*, volume 1, pages 43–53. ACM, 2018.

[51] A. Sree-Kumar, E. Planas, and R. Clarisó. Validating feature models with respect to textual product line specifications. In *VaMoS'21*, pages 15:1–15:10. ACM, Feb. 2021.

[52] C. Szyperski, D. Gruntz, and S. Murer. *Component Software: Beyond Object-Oriented Programming*. ACM Press and Addison-Wesley, 2 edition, 2002.

[53] H. Tajalli, J. Garcia, G. Edwards, and N. Medvidovic. Plasma: A plan-based layered architecture for software model-driven adaptation. In *ASE'10*, page 467–476. ACM, 2010.

[54] J. Troya, N. Moreno, M. F. Bertoa, and A. Vallecillo. Uncertainty representation in software models: a survey. *Softw. Syst. Model.*, 20(4):1183–1213, 2021.

[55] K. Twidle, N. Dulay, E. Lupu, and M. Sloman. Ponder2: A policy system for autonomous pervasive environments. In *ICAS'09*, pages 330–335, 2009.

[56] T. Vogel and H. Giese. A language for feedback loops in self-adaptive systems: Executable runtime megamodels. In *SEAMS'12*, pages 129–138, 2012.

[57] J. Winter, M. Aniche, J. Cito, and A. v. Deursen. Monitoring-aware ides. In *ESEC/FSE'19*, page 420–431. ACM, 2019.

[58] E. Yu. *Modelling strategic relationships for process reengineering*. PhD thesis, Department of Computer Science, University of Toronto, 1995.

[59] G. Zhang, H. Ye, and Y. Lin. Quality attribute modeling and quality aware product configuration in software product lines. *Softw. Qual. J.*, 22(3):365–401, 2014.

[60] H.-J. Zimmermann. *Fuzzy set theory—and its applications*. Springer Science & Business Media, 2001.