Uncovering Sustainability Concerns in Software Product Lines

Ruzanna Chitchyan[1]* Iris Groher[2] and Joost Noppen[3]

[1] *Department of Computer Science*
University of Leicester, University Road, Leicester, LE1 7RH, United Kingdom
E-mail: rc256@leicester.ac.uk
[2] *Johannes Kepler Universität Linz, Department of Business Informatics – Software Engineering*
Altenbergerstr. 69, A-4040 Linz, Austria
E-mail: iris.groher@jku.at
[3] *School of Computing Sciences, University of East Anglia*
Norwich Research Park, Norwich NR4 7TJ, United Kingdom
E-mail: j.noppen@uea.ac.uk

SUMMARY

Sustainable living, i.e., living within the bounds of the available environmental, social, and economic resources, is the focus of many present-day social and scientific discussions. But what does sustainability mean within the context of Software Engineering? In this paper we undertake a comprehensive analysis of 8 case studies to address this question within the context of a specific SE approach, Software Product Line Engineering (SPLE). We identify the sustainability-related characteristics that arise in present-day studies that apply SPLE. We conclude that technical and economic sustainability are in prime focus on the present SPLE practice, with social sustainability issues, where they relate to organisations, also addressed to a good degree. On the other hand, the issues related to the personal sustainability are less prominent, and environmental considerations are nearly completely amiss. We present feature models and cross-relations that result from our analysis as a starting point for sustainability engineering through SPLE, suggesting that any new development should consider how these models would be instantiated and expanded for the intended socio-technical system. The good representation of sustainability features in these models is also validated with two additional case studies.

KEY WORDS: sustainability, software product line engineering, case study analysis, text analysis

1. INTRODUCTION

Scientists across the world agree that the human society is facing major sustainability challenges [2]. Software, as one of the cornerstones and main drivers for change in society, should both adhere to sustainability, and promote it. Yet, at present there is no unified understanding either of what sustainability is, or how to incorporate sustainability concerns into design of software systems [7].

Sustainability is generally defined as the capacity of a system *to endure*[†] . For the human society to endure, it must undertake economic activities within some natural environment. Presently, there is an ongoing debate in the scientific circles as to how the society, economy, and ecology are really interrelated. Some, for instance, argue that human and natural dimensions are interchangeable,

*Correspondence to: F27 Computer Science Building, Department of Computer Science, University of Leicester, University Road, Leicester, LE1 7RH, United Kingdom. E-mail: rc256@leicester.ac.uk

[†] The Oxford Dictionary of English. Oxford University Press, 2010

consequently, reduction of natural resources is acceptable, if accompanied by increase in human capital (e.g., reduction of woodlands that produce oxygen is acceptable if a technology for artificial oxygen generation is found and the quality of human life is improved). Others insist that nature provides services that humans cannot reproduce (e.g., ozone layer), and so social and economic systems are encapsulated sub-systems of the environment (and human capital cannot substitute the natural one) [‡] .

The present paper takes a step towards informing the conversation on social, environmental, and economic concern manifestations and interdependencies within the discipline of Software Engineering (SE). It evidences the factors reported to make up the constituent dimensions of sustainability within a specific SE paradigm (that of software product line engineering), as well as the cross-dimensional dependencies of these factors. We take a pragmatic view on the notion of sustainability, based on the "lowest common denominator" of the sustainability debate, i.e. accepting the observations that: (i) sustainability relates to environmental, social, and economic concerns; (ii) these concerns are interdependent, and interact across time and space; (iii) within the context of human society, sustainability refers to keeping up/maintaining (for present and future) a preferred lifestyle, but accommodating it within the boundaries of available natural, economic, and societal resources (irrespective of the resource substitutability debate); (iv) as the total sum of resources (natural, human, and social) is limited[§] , to keep up a preferred lifestyle, these limited resources must be reused and recycled [9].

In line with Penzenstadler [39], Becker [7], and Lago [28] we refer to the following (slightly amended[¶]) set of sustainability concerns (also termed dimensions):

- Social dimension includes notions of societal communities (i.e., groups and organisations) and their interrelationships (e.g., employment, democracy), as well as individuals' concerns of skills, education, well being, etc.
- Economic dimension is centred on finances, assets, income, and alike.
- Environmental dimension is concerned with natural ecosystems, biodiversity, resources, effects of waste and pollution, etc.
- Technical dimension relates to the longevity of information, systems, and infrastructure, their adequate evolution, maintenance, obsolescence, etc.

Since **Software Engineering** practices differ quite significantly – from waterfall to agile, from Product Lines to Extreme Programming – the study of how the sustainability concerns manifest and interact within the environmental and social activities of the given development approach could also differ substantially. Thus, the study of such interactions must first be nuanced with the specific practice context, and only then generalisations could be drawn across contexts, as and where appropriate. Should an aggregated view on SE be taken without a number of contextualised studies, such generalised conclusions, if at all valid, may not be relevant or informative to any specific SE practice or community. Thus, in this first study of such sustainability dimensions and interactions for SE, we focus on the specific practice of Software Product Line (SPL) Engineering.

Software Product Line Engineering (SPLE) [40] [30] is an approach to SE that fosters systematic reuse through utilisation of commonalities among a set of similar products. Unlike most other SE practices, SPLE is explicitly focused on the notions of reuse and coping with evolution [40,17], where evaluation aims for the continuous reuse of the previously developed systems with the incremental (re-)development undertaken, where necessary. Given that under the constraint of the limited resources (e.g., electricity used for development, developer time, funds availability, etc.), the available resources must be reused and recycled, SPLE is the one SE approach that is aiming to support sustainability by aiming to: i) reduce the amount of new development to only the variable elements of a new product, thus reducing also development effort, time to market, etc.; ii) reuse the

[‡] Please see [7] for further details and references on this discussion.

[§] We note that though some resources, like sunlight and wind are classed as renewable, even their total sum is limited within each a given time period. The vast majority of resources, such as water, air, metals, etc. are non-renewable.

[¶] We consider society to include both groups and individuals as sub-categories under the common Social Dimension topic, whereas the listed references list Personal Dimension separately from the Social one.

core assets in as many products as possible of a given product line; and iii) recycle the core and variable assets of each product line by setting up product line families. Thus, we focused on the SPLE as an approach that explicitly aims to support (at least some aspects of) sustainability, and is ahead of other SE approaches with this respect.

Some work, such as [32] [44], discusses aspects of sustainability in SPLE, but no systematic analysis has yet been conducted to assess how SPLE addresses sustainability concerns.

The main goals behind this study are twofold:

- Promote awareness within SPLE and SE communities that sustainability is already a key topic within their domains, though various aspects of sustainability are still addressed in silos, without appreciation of their contributions to the holistic sustainability concern.
- Elicit the set of factors and their relationships, that are reported to make up the constituent dimensions of sustainability within the SPLE domain, as well as the cross-dimensional dependencies (beyond only the economic and technical aspects).

To ensure a structured and reproducible analysis, we adopt a qualitative analysis methodology that is a combination of text analysis and Grounded Theory (GT) [23] approaches, with references to all used and openly available materials provided.

The key contributions of this study are in that it shows that present-day SPLE work (as exemplified here by the analysed overall sample of 10 studies):

- Extensively addresses the topics of the economic and technical sustainability when undertaking a systems engineering project. The topics of the social sustainability, where these address the organisational concerns, such as processes, structure, and culture are also handled to a satisfactory degree. The issues related to personal sustainability, where employees are concerned, are represented to a much lesser degree, while the broader human aspect of an individual is absent in our analysed sample of SPLE projects. Similarly, topics related to environmental impact of these SPLE projects and environmental sustainability in general are nearly completely amiss. The identified factors that constitute sustainability dimensions are summarised as feature models.
- Has a large number of cross-dimensional interdependencies between various factors of sustainability, which are often implicit and poorly understood, yet could have a profound effect on success or failure of the product and business as a whole. We present these cross-dimensional dependencies in Figure5. Expanding on the work of Lago and colleagues [28], we suggest that this cross-dimensional dependency framework could be used to identify dependencies, synergies and conflicts between different features, supporting systematic multi-dimensional sustainability assessment of SPLE products.
- Has a large number of stakeholders that are typically associated with sustainability features simultaneously and will interact when such concerns are considered. We suggest that, combined with the cross-dimensional interdependencies, knowledge of stakeholders can be used to guide an SPL development process. This analysis helps to expose and explicitly account for the complex interdependencies between the stakeholders and their often conflicting requirements for sustainability.

This work can be used as a starting point for analysis and integration of sustainability requirements and effects into the SPLE and more general SE practice and on the structure of sustainability dimensions in Software Engineering and the interdependence between the constituent factors in these dimensions.

The remainder of the paper is structured as follows. In Section2we introduce a number of case studies used as a basis for our analysis of sustainability in SPLE and present the methodology used for this analysis. Section3discusses findings that emerge from our analysis - a set of feature models that capture what are the issues and concerns related to sustainability that are currently considered relevant within SPLE research and practice, as well as their cross-dimensional dependencies and stakeholder relations to sustainability features, as evidenced by the documents used in our study. Section4discusses the validity of our feature models and potential threats to the study validity.

In Section5we discuss emergent topics from our analysis, as well as how current metrics and measurement in SPLE support or hinder sustainability design into SPLE products and processes. Related work is presented in Section6; and conclusions are given in Section7.

2. STUDY OUTLINE

In order to elicit an understanding of how the topic of sustainability relates to SPLE we analyse eight previously published and publicly available studies of SPL development using adapted version of the qualitative Grounded Theory analysis approach [23,46].The case studies and the methodology applied for analysis are presented below. The raw data generated by this study is available as an online resource[1].

2.1. Case Studies

All eight selected case studies address the use of SPLE within a number of application domains and for different purposes. The main driving criteria for choosing these studies are that they address a wide set of domains (home automation, industrial automation, mobile games, ERP systems) and purposes (technology projects, interview studies, pedagogical product lines) and the studies themselves are publicly available from respectable publication sources" (such as a refereed conference or a report from verified research/practice institution). These case studies are not explicitly related to sustainability, and any other similar SPLE project publications could have been equally suitable for analysis. The selection process of the case studies is further explained in Section2.2. The case studies used in this analysis are summarised in the TableI.

2.2. Study Method: Adapted Grounded Theory Analysis

For identifying sustainability factors in the case studies we performed a Grounded Theory (GT) analysis, combined with the deductive categorisation approach suggested by Mayring [35]. Grounded Theory (GT) is an approach for building a theory on basis of the available data [23,46]. It works by systematically sorting the given data into labelled groups (called codes, e.g., "spl management"), and keeping notes (termed memos) on the observed relationships and dependencies between these codes as well as on the insights obtained on the domain of study throughout the data sorting process. The codes must represent concepts of key importance within the given data. The data coding and memo writing is an iterative process, which leads to an emergent theory that is fully grounded in (i.e., explained by) the original data.

Two key features of the GT are the principles of *constant comparison* and *theoretical sampling*. Under the *constant comparison* principle, as the collected data is analysed, it is continuously compared against the current reality, already analysed data, available prior theoretical knowledge (with the proviso that this could enrich, but must not restrict the freedom of the present analysis [46]), and perspectives of the other analysts. Throughout the analysis stage a decision on what new/additional data needs to be collected is made, in accordance with the *theoretical sampling* principle. Thus, what new data is selected to be added to the analysis sample set is influenced by the emerging results from the analysis itself.

In this work, the initial selection of case studies for analysis included only three - Arcade Game Maker, DiVA, and Smart Home. These three sources were analysed using the *constant comparison* principle, with the emerging set of sustainability models and interrelationships constructed. During the analysis it was observed that new domains and types of study are likely to facilitate identification of more concerns and relations. Thus the *theoretical sampling* rule used in new data selection was that the additional case studies must be taken from varying domains and study types. Thus, we added

"Please note that the reference from Dhungana et al. [19] contains a set of four different SPLE case studies. The four case studies use the same variability management tool for modelling variability and related assets (DOPLER tool suite) but have been conducted in different domains with different purposes. We thus consider the case studies independent with respect to the addressed sustainability concerns.

Table I. Coding case studies

Main Focus and Summary
1. *Arcade Game Maker (AGM)* **Domain:** Games development; **Type:** Pedagogical software product line Arcade Game Maker SPL supports a range of arcade games (e.g. bowling and pong) and consists of around 20 features at various levels of abstraction. The case study itself consists of a wide variety of documents outlining the economic, social and technical considerations and challenges that were taken into account in designing and implementing the software product line. The SPL follows the general principles of SPL development using feature models to describe variability and architecture plans and unit tests for the actual design and realisation. **Project Aim:** to supporting the teaching of SPL concepts in a university curriculum or an industrial consulting setting. **URL:** http://www.sei.cmu.edu/productlines/ppl/
2. *Dynamic Variability In Complex, Adaptive Systems (DiVA)* **Domain:** Crisis Management and CRM; **Type:** Technology project Diva project delivered a set of tools and a methodology for managing dynamic variability in adaptive systems using the crisis management and customer relationship management business cases. The Diva solution combines aspect-oriented and model-driven techniques. The approach is divided into two stages: (i) design time, which focused on analysing and modelling the requirements to drive the adaptation model and architecture, to specify the relationships between the variability aspects and the static part of the system, and (ii) runtime, where the adaptation model is processed to produce the system configuration for execution. **Project Aim:** to demonstrate usability and utility of the combined methodology and tools. **Reference:** https://sites.google.com/site/divawebsite/home
3. *Smart Home (SH)* **Domain:** Home automation; **Type:** Technology project Here a product line for home automation systems was developed. A smart home is an inhabited building equipped with sensors and actuators that supports intelligent sensing and controlling of the building's devices. The inhabitants of the home monitor and control the home from various user interfaces. The home also allows the devices to coordinate their behaviour and fulfil complex tasks (e.g., climate control, security system) without human intervention, following pre-set policies. Using a model-driven, aspect-oriented infrastructure this project takes the description of a home as input and outputs the controller software for a smart home. **Project Aim:** to demonstrate the utility of SPL approach applying aspects and models in the smart home domain. **Reference:** Rashid et al [42]
4. *Very-Large-Scale Software Systems (VLSSS)* **Domain:** Industrial automation and metallurgical plants; **Type:** Interview study This is an exploratory study with expert interviews involving engineers and technical project managers of an industrial automation of very-large-scale software systems (VLSSS). Nine people participated in the case study. The study provides empirical evidence on how VLSSS are tested, commissioned, and operated in practice. Within the study practical challenges and industrial requirements regarding process and tool support are identified. **Project Aim:** to find out how VLSSS are tested, commissioned, and operated in practice. **Reference:** Vierhauser et al [49]

5. *Eclipse-based Maintenance and Setup Tool (MSS)*
Domain: Industrial automation; **Type:** Technology project

The MSS supports developers and operation staff of Siemens VAI customers to fine-tune deployment parameters of the CC-L2 process automation system on site. The system comprises 100 Eclipse plug-ins organised into 20 Eclipse features that can be combined flexibly and customised through diverse parameters. The size of the code base is about 200 kLOC. MSS is adapted to more than 20 customer environments per year.

Project Aim: to analyse and model the variability of an Eclipse-based product used in industrial process automation.

Reference: Dhungana et al. [19]

6. .*NET-based Business Software (BMD)*
Domain: Enterprise resource planning; **Type:** Technology project

BMD Software is a comprehensive suite of enterprise applications for customer relationship management, accounting, payroll, enterprise resource planning, as well as production planning and control. Customised products are an essential part of the company's marketing strategy. The case study investigates the variability of distinct user-visible features of the business applications represented by plug-ins. Variability is relevant because the business processes require user interfaces specific to the company's needs and roles of the users.

Project Aim: to evaluate variability of distinct user-visible features of the business applications represented by plug-ins.

Reference: Dhungana et al. [19]

7. *Steel Plant Automation (SPA)*
Domain: Industrial automation and metallurgical plants ; **Type:** Technology project

This case study has been conducted in cooperation with Siemens and VAI Metals Technologies. VAI Metals has developed and maintains the Continuous Casting Level 2 (CC-L2) product line of steel plant continuous casting automation software. About 40 software engineers are involved in the development and maintenance of the system which contains about 1.6 million lines of Java code. The case study focuses on the variability of the CC-L2 software which is responsible for process supervision, optimisation, material tracking, etc. During the case study, the variability of the assets and the corresponding decisions are modelled. Customer-specific instances of the product line are built by selecting the necessary components and customer-specific extensions.

Project Aim: to analyse and model the variability of the CC-L2 software using the DOPLER tool suite.

Reference: Dhungana et al. [19]

8. *IEC-61499 Industrial Automation Systems (IEC)*
Domain: Industrial automation; **Type:** Technology project

The case study investigates the usefulness of the DOPLER tools and techniques for modelling product lines in the domain of industrial automation systems (IAS). Such systems are often based on a distributed architecture consisting of multiple physically and/or logically distributed components. Many IAS are based on the standard IEC 61499 which provides a component-based framework for automation systems and standardises the use of function blocks in distributed industrial process measurement and control systems. The case study investigates the variability in function block based systems conforming to the standard IEC 61499. Variability in this case deals with runtime variability of function blocks that supports automated runtime reconfiguration.

Project Aim: to evaluate variability in function block based systems conforming to the standard IEC 61499 in the domain of industrial automation systems.

Reference: Dhungana et al. [19]

five more case studies, as summarised in TableII. The sampling was stopped when the set of codes from newly added sources became saturated, i.e., no substantially new category was introduced and the number of additional sub-categories was very small [46]. As shown in TableII, the last two case studies together provided one new high-level and four lower level specialisation codes, leading us to conclude that the set of codes at that point was sufficiently saturated.

We must note that the SPLE is a wide and mature area of SE, thus, it is likely that additional studies could further expand the models presented in this paper. Yet, as the aim of this work is to present a sufficiently comprehensive (though in no way complete) view of sustainability in SPLE, the relative saturation observed in our analysis serves as a satisfactory point of data collection termination for this study.

We used GT to freely identify and integrate the codes from the eight selected studies into a single codeset. However, we adapted the classical GT approach for this activity in several aspects, by combining it with the deductive categorisation suggested by Mayring [35]. Mayring's model contains four steps: *data preparation*, *definition of categories*, *codification*, and *interpretation* which are performed iteratively.

1) Data preparation: Whereas with the classical GT the full content of the given data is expected to be coded, we opted for use of selective coding - limiting the analysis only to the notions relevant to the topic of our interest - the sustainability domain. Thus, as per data preparation step of Mayring, the relevant text parts of the published case studies were identified. We chose parts that describe the case studies themselves, their realisation, and findings. We did not use text that, for example, described related work in the respective papers. Here, to support our analysis, we also copied the relevant text from each case study into the Saturate** web-based qualitative analysis tool.

2) Definition of coding categories: Unlike the classical GT, which presumes no starting codes at all, we started with the minimal set of codes related to the five commonly accepted aspects of sustainability: *economic*, *environmental*, *social*, *personal*, and *technical* aspects. Beyond this, the categorisation of data into relevant groups and emergence of theory, was carried out as per GT - no limits or restrictions were set on new code emergence.

The category system has been iteratively refined and extended by subcategories during the analysis process. New categories were always discussed by all three analysts (also the authors this paper). The final category system is presented in Section3.

3) Codification: The codification step for each case study included the detailed reading of each case study text. Each text fragment, considered relevant, was assigned to one or more (sub)categories. Where the set of available codes did not suffice, a new category was created to represent the relevant concept. Definition of each new category was performed using the "4-eyes principle", whereby at least two independent analysts (in our case it had to be all three authors) would need to agree on the relevance and necessity of the new category. Thus, three researchers carried out coding and validation of the relevant concepts in the case studies. Where disagreements arose, the researchers resolved these and came to a unified code set. Simultaneously, notes on the relevance and interrelatedness of the coded concepts were collected. In total, we assigned 556 text fragments from the case studies to the different categories. Less than 1% of the text assignments were made to the environmental sustainability category, 33.5% of the text assignments were made to the social sustainability, 26% to economic sustainability, and 40% to the technical sustainability categories.

4) Interpretation: The resulting set of categories was then reviewed to ensure that any duplication and redundancy was removed. As coding progressed, we observed that the nested structure of codes (e.g., "spl management/analysis", "spl management/implementation") lent itself directly to the feature-tree representation format. Thus, our analysis has resulted in construction of feature trees that represent the sustainability concerns in present SPL research and practice (as per our analysed data).

We also observed that a number of codes were clustered around the same text fragments. As discussed in [41,16], collocation of broadly scoped concerns around a single textual requirement

**saturateapp.com

indicates potential interdependency relationship between these concerns [††]. In other words, such clustering implied interdependencies between the features represented by the code categories. For example if a text fragment has both economic and social tags attached to, it is likely that the economic and social concerns are interacting around that text. Such interdependencies are documented in Fig5and discussed further in Section3.5.

Finally, we have aimed to visualise the impact of the various stakeholder roles on the factors of sustainability. For this, we have identified the stakeholder roles using the set of role templates given by Alexander [4] and related these to the sustainability concerns within the context of which these roles were mentioned. While it is clear that the relevance of these roles to various topics is very much case and domain specific, and , since we work with secondary requirements sources, in many cases (many of the potentially) relevant stakeholders are not mentioned, we were rather interested in possibility of the overall trend emergence. In total we assigned 130 text fragments from the case studies to the different stakeholder types with 33% assigned to management/administration, 36% to the developer role. 12% has been assigned to the customer, 7% to the user and 9% to the expert role. Finally the operator role was addressed in only 3% of text fragments.

3. FINDINGS ON SUSTAINABILITY CONCERNS IN SPL

Through the Grounded Theory's coding and model building process (as presented in the above section) the theory of the constituents of sustainability in SPLE is elicited, as embodied in the feature – and their interdependency models, shown in Figures 1–5 below. These models represent all factors found relevant to further the economic, social, environmental, technical, and personal dimensions of a software product, while employing SPL engineering method, given our analysis data. Note, that even factors negatively correlated to these dimensions are relevant, as they too have bearing on the success of the noted product aspects.

Sustainability, or "keeping up" of the given dimension is accomplished by continuously retaining or increasing the given target level of the relevant dimensional aspect. This can be measured, for instance, in monetary income for the economic dimension, core asset utilisation for the technical dimension, or CO_2 emission equivalent for the environmental dimension, as further discussed in subsection5.2under the "Sustainability as a topic of interest in SPL: Metrics Review" heading.

Here we also discuss the cross-dimensional dependencies between the factors of various dimensions, which negatively impact the "keeping up" the target levels of the given dimensional values, or sometime positively reinforce them. This is presented in Figure5and discussed in the subsection3.5.

While we do not discuss how specifically to achieve the up-keeping (i.e., sustainability) in the SPLE, we focus on what factors comprise the various sustainability dimensions and what cross-dimensional dependencies can be observed between these factors.

The analysis of 8 case studies, discussed in TableIabove, resulted in 127 codes (summarised in TableII), 40 of which relate to economic sustainability concerns, 48 to technical, 38 to social (21 to personal and 17 to organisational), and 1 to environmental. Note that in TableIIthe SPA and IEC case studies have been treated as a combined case study in accordance with Grounded Theory as the case studies addressed the same domain [46]. Below we discuss each of these code categories in more detail. We also provide a few extracts from the text analysis to illustrate the relevant concepts.

3.1. Economic Sustainability

Economic sustainability is the ability of a business to "keep up" its operation successfully. A software product line, essentially, is a business model - a way to organise the production, customisation, and reuse of software assets, which dictates a respective organisational structure

[††] Since the code categories represent sustainability features that affect multiple areas of a software system, these features are crosscutting/aspectual concerns. When aspects crosscut the same point (called join-point), they are likely to demonstrate trade-off, conflict, or complementarity relationships towards each other, as further discussed in [41,16]

Table II. Number of codes added per each case study analysis

	AGM	DiVA	SH	VLSSS	MSS	BMD	SPA/IEC	Total
Environmental	-	-	1	-	-	-	-	**1**
Social: Personal	3	-	6	5	2	1	4	**21**
Social: Organ.	9	3	-	3	2	-	-	**17**
Economic	26	2	9	1	-	2	-	**40**
Technical	9	19	14	3	1	1	1	**48**
Total	**47**	**24**	**30**	**12**	**5**	**4**	**5**	**127**

and work roles. It should therefore come as no surprise that economic sustainability is one of the largest topics we identified; it produced 40 codes. As most businesses go, the SPL model is adopted if a company considers it relevant and that model is maintained for as long as it contributes towards the business profitability. At the core of economic sustainability topics identified in the case studies lies a division between the improvement goals sought by businesses and the strategies that can be followed to achieve such goals, as shown by the two major sub-features of the Economic Sustainability feature tree derived through our study (Figure 1).

3.1.1. Business Improvements

The improvements to the business naturally focus on increasing the **profitability** and **efficiency** of the company, thereby achieving a greater economic sustainability of the organisation (e.g., "*Efficiency: Does work package 1 (WP1) simplify the analysis of large requirements...*" DiVA) and "productivity" (e.g., "*...in which measure does it reduce the time needed...*" or "*... the productivity has been evaluated...*" DiVA).

Software quality is generally identified as relevant for business improvement, but the key returning themes in the majority of the case studies are **reduction in costs and reduced time to market**. The **cost reduction** is observed, for instance, in the Arcade study through automation, reuse and reduction in personnel required (e.g., "*[...] goal is to increase the automation in product production so that costs are lower ...*", "*[...] produce a very complete product line of freeware games at very little cost.*", "*[...] reduce the number of staff needed to produce a product.*"). The **reduced time to market** of products theme is observed in, for instance, Arcade (e.g., "*The benefits of using a product line [...] shorter time to market.*"), Smart Home (e.g., "*...the need for short time-to-market and saving of costs drive the need for a Smart Home product line...*").

3.1.2. Strategies for Goal Achievement

The second major feature of economic sustainability is the **business strategy** to be employed to achieve economic goals. Alongside with **standardising processes** (primarily discussed in Arcade study) this feature sub-divides into two further parts: the **definition** of the SPL business model and that model's **implementation**. For the **definition**, the case studies frequently refer to such techniques as commonality-variability analysis (e.g. Arcade: "*... The architecture captures and exploits those commonalities and variabilities....*"), and cost-benefit analysis (e.g., Arcade: "*...the costs and benefits for each approach across the increments is rated.*"). These align with well known analysis and definition methods for the design of a software product line that is focussed yet flexible. These definition strategies go hand in hand with the implementation of the SPL Business model where features such as tooling, infrastructure, and reusable assets are identified, as are strategies for adoption and practice of SPL model (pro-active, reactive, etc.).

Overall, from these case studies, it is clear that economic sustainability is a major consideration in the adoption of an SPL based approach. Indeed, in its core, the software product line philosophy has always intended to streamline the creation of families of software products with the focus on reuse. As such, the SPL worldview directly contributes to the sustainability of software companies by increasing their productivity and reducing their costs. Interestingly, in our case studies we found hardly any consideration for the *sustainability of the SPL itself* as opposed to the sustainability of

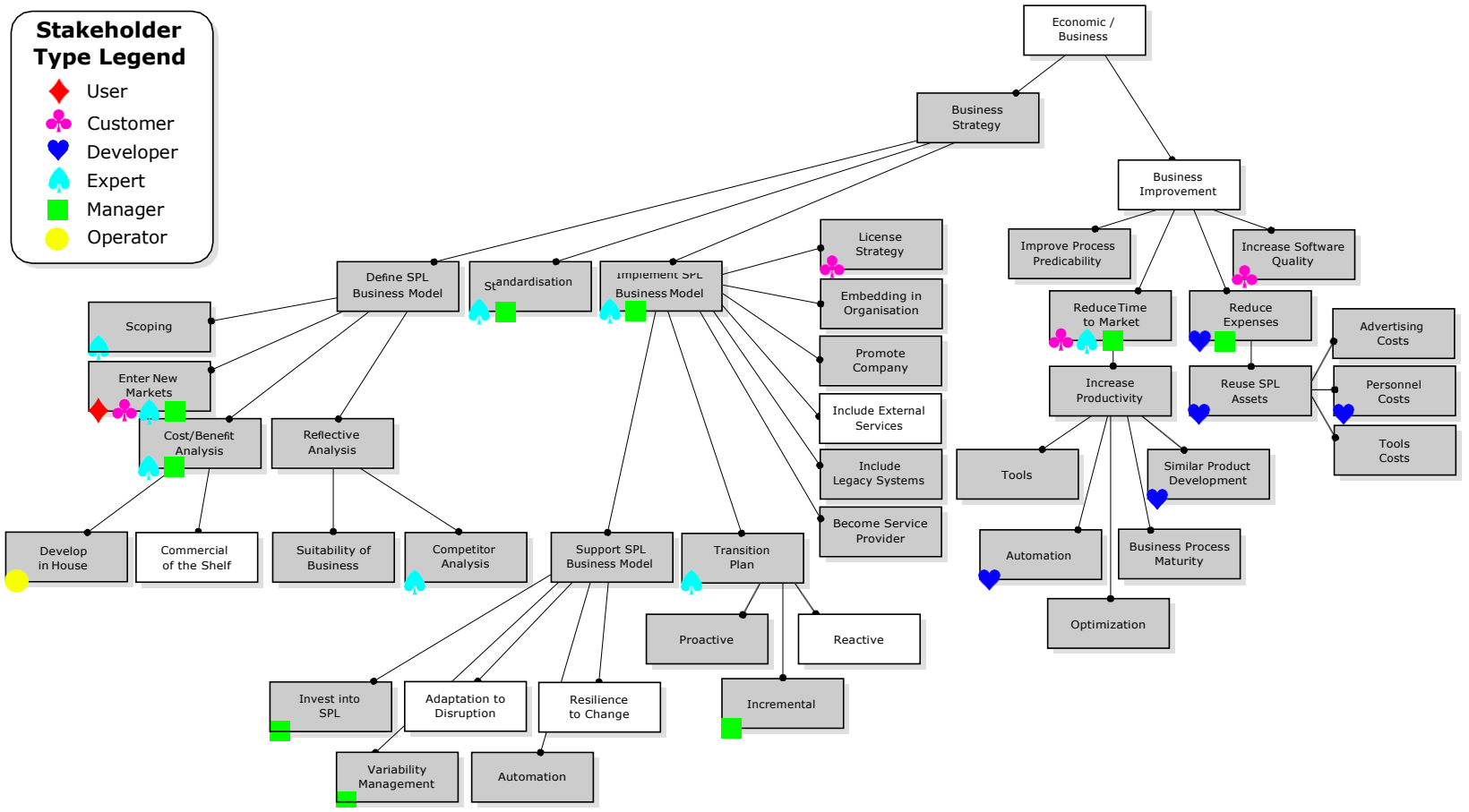

Figure 1. Derived economic sustainability feature tree

the *organisation adopting it*. The economic viability of SPLs generally is considered over a specific **timeframe** rather than in terms of prolonging its longevity.

Let us review the dynamic process of economically sustaining an SPL business, the static snapshot of which has clearly emerged from our case study analysis:

1. One or more goals are defined to improve the economic return of the business (as shown by the Business Improvement feature branch in Figure 1)
2. Then the SPL business model is defined based on costs, etc.
3. The defined model is implemented through a number of means, such as Licensing (e.g. BMD: "*...At the supplier level, BMD has structured the software into seven solutions which can be individually licensed and composed into five main products covering major markets...* "), service provision (e.g. Smart Home: "*...Extensibility: The inclusion of new services and extension of already available services should be easy...*"), etc.
4. The implemented model is supported through investment, automation, etc. (e.g. Arcade: " *The strategy calls for a migration from manual production to automatic production...*",) until such time (or event) as the set goal is met or expected to be reviewed
5. Thereafter step 1 is taken again.

To summarise, within the SPLE context, economic sustainability is:

- defined in terms of **goals**,
- it is aimed at sustaining **profitability of the business** and not the longevity of its products,
- it is defined and implemented based on **costs and benefits** expected of a specific SPL solution;
- it is set for a **particular timeframe**, and reviewed periodically.

3.2. Social Sustainability

Within the SPLE domain, social sustainability pertains to the "keeping up" of the business organisation, its employees, and the relationships between the business and the employees in order to provide a supportive environment for productive work. We note, that in the studied texts, the topics related to personal sustainability exclusively pertain to the role of an individual as an employee, with no notions related to broader human aspect of the individuals (such as job satisfaction, self-worth etc.) discussed. Consequently, the personal sustainability code is integrated under the social sustainability umbrella. It is easy to notice that the feature tree depicting the social sustainability concerns (see Figure 2) is sharply divided into two major branches, one for the personal and one for the organisational sustainability concerns.

3.2.1. Personal Sustainability

The first branch - Personal Sustainability - depicts how SPL pertains to well being of the business employees. This feature if further divided into concerns related to **time pressure** (e.g., MSS "*[ensuring correctness of MSS is] especially hard under time pressure...*"), **personalisation** (e.g., MSS: "*...certain MSS editions are used by domain experts such as metallurgists while other editions are mainly targeted at software engineers...*"; Smart Home: "*A user should be able to define her/his preferences*"), and **skills level** (e.g., Arcade refers to "*...sustaining the expertise...*", VLSSS notes that "*highly skilled specialists with very specific knowledge are required*").

With 45 recorded references, the skills level topic is the predominant one in our texts with respect to personal sustainability. The focus here is on **training personnel** to ensure the required level of skills is obtained (as noted in Arcade and VLSSS), **convening domain knowledge** to the engineers, (as noted in VLSS), as well as ensuring that the required **experts are available** and **supported**. Within the personal sustainability branch the user is also explicitly identified as part of social sustainability, for example through means of **human machine interaction** and **product quality**.

We note that personalisation and time pressure are barely mentioned in the text (with 4 and 1 references respectively). This, however, does not reflect the true scope of relevance of these topics, as they are both indirectly referred via codes in other categories. Thus, personalisation is clearly

related to the technical concern of *usability* of the tools and techniques, while time pressure is tightly related to the goal of reducing *time to market* discussed under the economic sustainability code set. We will discuss relationships between features across the different feature models in more detail in Section 3.5.

3.2.2. Organisational Sustainability

For organisational sustainability, **communication** between stakeholders and the **business culture** of the organisation are deemed very important, as per our texts.

Increasing communication is discussed, e.g., in VLSSS case study in terms of external communication, i.e., interacting with the customer. To increase communication, an SPL business would **provide documentation**, **introduce formal channels**, and **support traceability**. For example, Arcade case study aims to "*... ensure clear communication...*" , states that "*any new processes must be coordinated via ... existing process roadmap*" and uses a plan that "*... describes the specific set of actions that will achieve ... goals*".

Establishing a **business culture**, on the other hand, is a much more intricate task. This is tackled via such avenues as:

1. **establishing a community of practice** (e.g., Arcade: "*...functional team members who are assigned to different building teams share experiences at functional team meetings.*". Here it is suggested that projects should be supported by dedicated teams, and these teams should enjoy relative autonomy.
2. **reducing complexity of the organisation** by defining clear roles and boundaries of responsibilities (e.g., Arcade: "*organize around role.*", VLSSS; "*each component has a dedicated team with engineers responsible for development, testing, and maintenance*".)
3. **supporting an established process and transition plan** when intending to adopt SPL for the first time. For example, Arcade notes that "*the morale of personnel will be damaged if the product line initiation effort is not successful*" and risks are mitigated by adopting SPL in "*layers*", i.e., gradually for various tasks, not in one go for the whole business.

We must note that the number of references to the topics of organisational sustainability is much more significant than that of personal sustainability. The focal point here is the topic of organisational complexity (with 79 references), followed by the issues of transitioning to SPL (with 54 references). Both of these topics are important indeed, as they pertain to the stability of the organisation itself, which can be threatened due to improper SPL adoption attempts. As Tainter demonstrates [48], complexity builds up in an attempt to solve problems; in case of SPL, the business model and process complexity increases to solve the productivity and reuse problems. However, increased complexity will eventually lead to organisational collapse [48], unless roles, responsibilities, and group membership of employees is simplified. It is this simplification and stability that the organisational sustainability features (see Figure 2) aim to furnish.

To summarise, within the SPLE context in our analysed documents, social sustainability:

- is defined in terms of **organisation** and its **employees**,
- is aimed at sustaining **stability of the business** and **skillset** enhancement of its employees,
- aims to ensure **clear roles and responsibilities** for the employees and **clear communication channels** across and between them and their clients,
- defines the structure of the organisation (e.g., via team assignments) on **per-project** or **per-product** basis and it is reviewed periodically.

3.3. Environmental Sustainability

The environmental part of the feature model is extremely sparse (see Figure 3), with a single sub-feature that deals with **reduction of resource** use, where "resource" implies energy. This code is used only in two case studies (4 times altogether). In the context of IEC study it is noted that an "*energy management application is to manage and control the distribution of electrical energy*

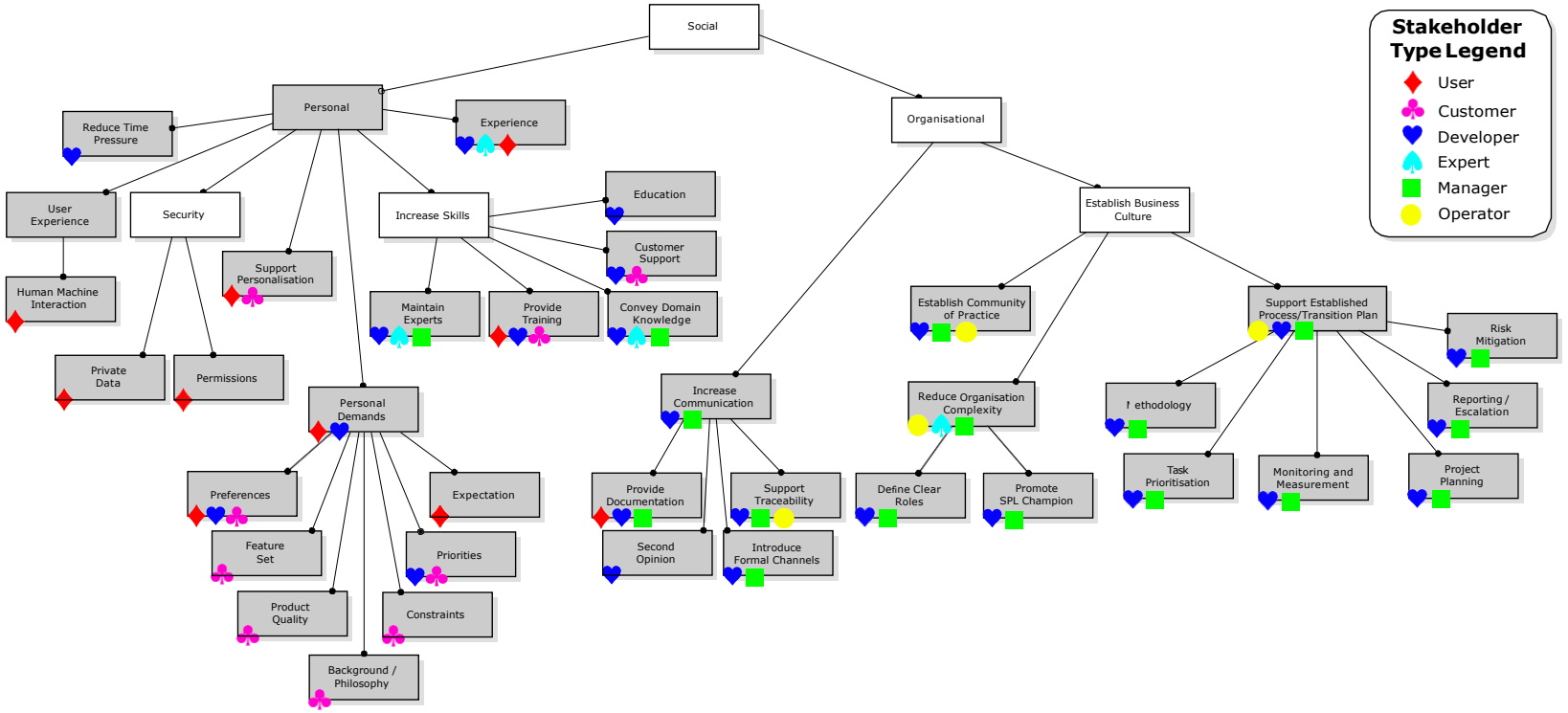

Figure 2. Derived social sustainability feature tree

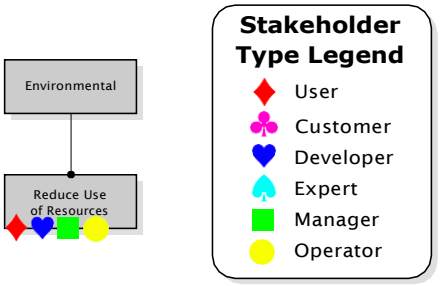

Figure 3. Derived environmental sustainability feature tree

[observing] priorities of consumers or environmental constraints for producers". In the Smart Home case, the devices in the home must "*spend as little energy as possible*".

A number of the other identified factors (technical, social, economic) could have potentially substantial environmental impact. For instance, increase in productivity and efficiency will often imply reduced resource use; improved technical assets will require less energy and human resources use for their maintenance, better tool support will also quicken development of new systems again reducing energy and other resource use for a given project. These, however, are all indirect effects from codes identified under other categories (as further discussed in Section3.5). From this feature model, it is quite obvious that environmental concerns have not been considered as a matter of any significance to the SPL studies used in our analysis.

To summarise, within the SPLE context in our analysed documents, environmental sustainability is defined in terms of **reduction of resource (i.e., energy) use** only.

3.4. Technical Sustainability

Here technical sustainability relates to the ability of the business to "keep up" its technical assets and their characteristics which ensure useful service of these assets in present and future. Yet, we note that the useful service of the assets is meaningful if the organisation that operates these assets also persists into the future. As one of the largest topics within the reviewed documents, technical sustainability has produced 48 codes, aggregated into the Figure4. The key topics within technical sustainability area are centred around SPL technical management (23 codes), quality factors (such as adaptability, traceability, scalability) that the SPL is concerned about (18 codes), and tooling (7 codes).

3.4.1. SPL Technical Management

SPL Technical Management topic covers the issues of SPL **analysis**, **modelling**, **implementation**, **configuration management**, and **maintenance**. These topics mirror the software development life cycle activities that take place within the PL software.

The **analysis** concerns relate to *domain analysis* and *scoping*, analysis of feasibility for SPL use as well as *change impact* due to requirements change/evolution, and *variability* and *commonality* of SPL assets. We note that some types of analysis are explicitly discussed in most studies (e.g., variability and commonality analysis are discussed in Arcade, Steel Plant, IEC, MSS, BMD, and DiVA) , while others are discussed only in one or two cases. For instance, *change impact analysis* (e.g., Smart Home: " *[..] know what artefacts to change when requirements change (e.g. where new variations have to be built) [...]* ") is mentioned only once overall; *domain scoping* (e.g., "*The PLM [...] examines a proposed product to determine whether it is within the scope of the product line.*") and *feasibility analysis* ("*[...] PLM charters a feasibility study. [...] determines whether to accept the new product into the product line or to reject the product.*") are discussed only in Arcade. Yet, implicitly, these scarcely mentioned types of analysis would have to be completed in a production-ready SPL, where domain analysis will be essential for an SPL construction in the first place and impact of requirements change will have to be observed and implemented for a new release of a product.

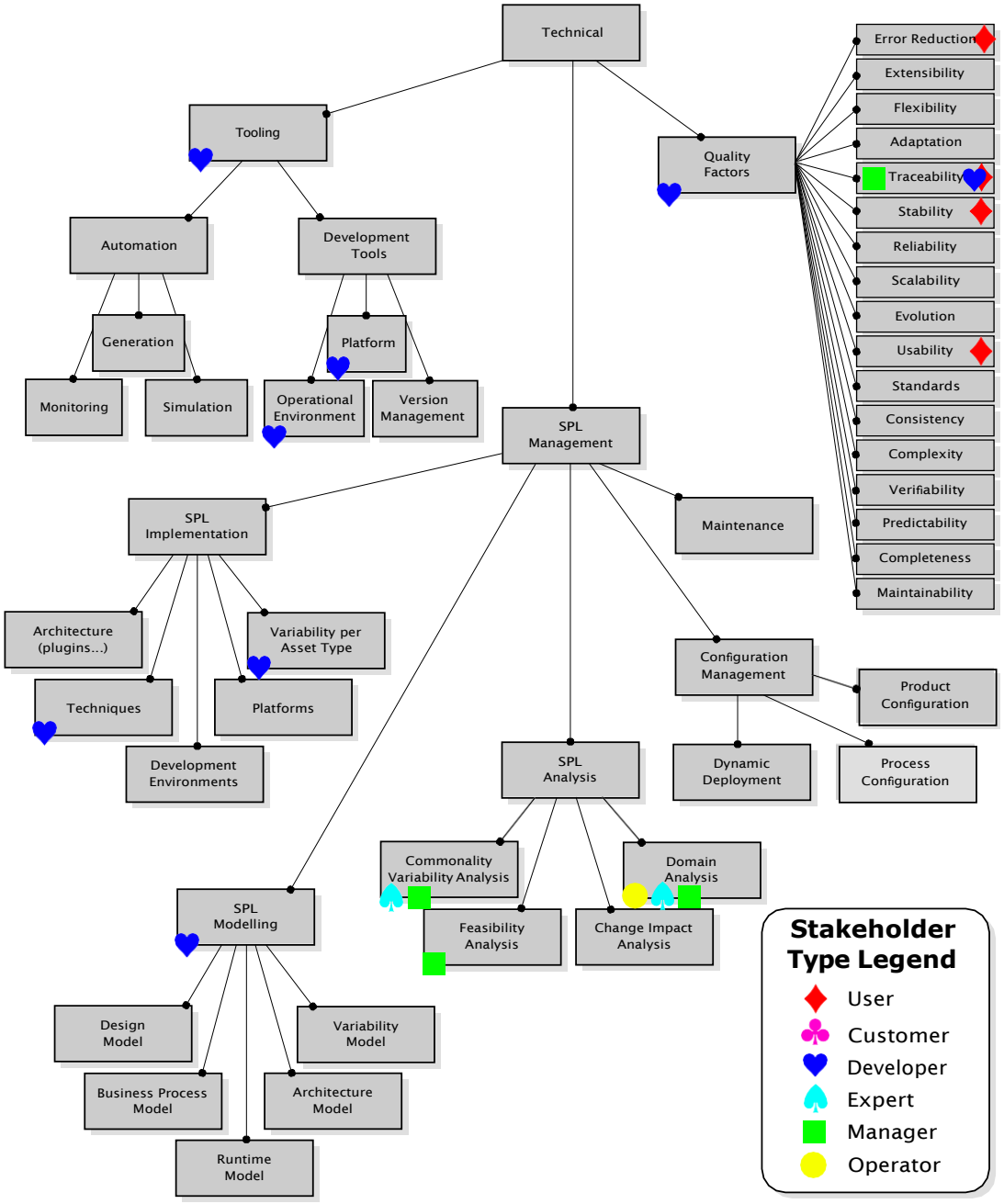

Figure 4. Derived technical sustainability feature tree

The **modelling** categories identified for technical sustainability have highlighted that models are used for all parts of SPL product and process lifecycle. The architecture model is "*[...] perhaps the most significant core asset...*" (Arcade), as this is the medium around which various products are instantiated and the quality characteristics of the product or process are defined. This is complemented with design modes which represent the *instances* of the PL components, at different *granularity levels*. These models often use *domain specific languages* and are elaborated through *configuration setting models*. *Variability* models are used to reason about decisions on variations and product core, while *runtime* models are used to reason about deployment parameters for a given product.

The SPL **implementation** codes are centred around the topics of realisation of SPL through software and hardware, where *variability per asset type* is realised. Here the notion of *core software assets* plays a notable role, as these assets are implemented for further reuse and formation of the product line. Implementation techniques for the *architecture* and software components, *development environments*, and *platforms* are also central to this category.

The **configuration management** activity is particularly relevant to the SPL-based companies, as they must configure each product using their core assets, and/or develop core assets for a new family of products, if no previous assets exist. Here the management practices relate to version and configuration management *for each product over time* and within its lifecycle (including *dynamic deployment* practices), as well as management and adaptation of the *SPL business process configuration*.

Finally, the issue of **maintenance** and upgrade of SPL software and hardware was identified to belong to the SPL Management sub-category.

3.4.2. Quality Factors

Quality factors constitute a large set of topics: in our analysed documents we have identified 16 such factors, such as *traceability, reliability, scalability* (e.g., DiVA: "*Does the reconfiguration technology support technology heterogeneity and distribution?*"), *evolvability* (e.g., DiVA: "*Specification of adaptation policies: Identifying variability and modelling it efficiently is a key concern when engineering large and complex systems*"), *maintainability* - to name a few. From the technical perspective, the SPL practice tends to be adopted within a company in order to improve quality of the software products and processes. Thus, it is not surprising that such a large set of relevant quality attributes has emerged.

3.4.3. Tools and Automation

Since software is the core part of the SPLs studied in this work, the notion of software engineering tools is also critical here. Tools are categorised into **automation** and **development** groups. In the automation group (software) tools are used to produce new (software) tools (e.g., DiVA: "... *the tools and methodology [...] enable and ease the design of complex adaptive systems*...") and for *simulating* possible product configurations as well as their runtime behaviour, and/or *monitoring* the processes or product performance (e.g. DiVA *"[Tool] ... to weave configurations from a simulation result [..], thus verifying what will be deployed once a reconfiguration context has been validated."*). The development tools, on the other side, are used for the traditional development activities, such as supporting the *operational environment*, provision of *development platforms*, and *version management*.

To summarise, within the SPLE context in our analysed documents, technical sustainability:

- is defined in terms of **technical management** activities of SPL assets and its **quality attributes** that the SPL aims to achieve,
- is aimed at sustaining a set of core assets by **analysing, modelling, implementing, and maintaining** them out of which products are **configured** that meet pre-set target **quality attributes**, such as usability, scalability, predictability, etc.
- the **maintenance** activities are to ensure that the products and assets continue to meet the expected qualities, and
- a set of **tools** are employed to support this cycle.

3.5. Relationships across Feature Models

3.5.1. Cross-Dimensional Dependencies

While analysing the coded text and constructing the above feature trees, we observed that a number of codes were allocated to the same textual fragment. Such co-located codes indicate that there is a potential cross-feature interdependence between various areas of sustainability, as summarised in Figure5. Lago and colleagues [28] have previously noted that the explicit demarcation of dependencies between dimensions of sustainability is of particular value during trade-off analysis

of software requirements. In our work the cross-dimensional dependencies between features could be used to undertake multi-dimensional sustainability assessment of SPLE products, as the explicit effects in change of a seemingly localised feature could have substantial repercussions. Below we discuss a few examples.

Effect of the Environmental Concerns: First and foremost, it is clear that, although there is only one code for environmental sustainability – *environment/reduce use of resources* – identified in our sample document set, if included into a product line configuration, it will be a key driver for configuration composition. This is because to reduce resource use, a business must either:

1. increase its productivity, which is a key part of the economic side of sustainability, as depicted by the *economic/reduce time to market and increase productivity* code;
2. change its applied technology, which pretty much affects the whole of the technical sustainability feature model as shown in Figure4, including the tools used in technology production (*technical/tooling* feature branch), and/or its technical management practices (*technical/spl management* feature branch), and/or;
3. achieve better technical quality outcomes for the given technology, as depicted with the *technical/quality factors* branch of the technical feature model;
4. invest into skills and training of own personnel, as depicted by the *social/personal/increase skills* branch of the social feature model (see Figure2), and/or,
5. improve the collaboration within the organisation, as per the *social/organisational* branch of the social feature model.

Thus, each and every part of sustainability model is affected by the environmental considerations. Yet, this is not a one-sided influence.

Effect of the Technical Concerns: Technical concerns are either a powerful enabler or detriment to the social, economic and environmental concerns. For instance, such technical quality features as *standards* (which relates to documentation, clear meta-model, availability of editors) as well as *usability* etc. are aimed at the longer-term persistence and utility of technical assets. Yet they also help the employees to learn tools easier, contributing to employee social sustainability (e.g., via *social/personal/reduce time pressure* code). At the same time they increase productivity, contributing to economic sustainability (e.g., via *economic/reduce time to market and increase productivity* code). For instance, the IEC study notes that "This is where variability models help they can be used to automate the manual, error-prone approach of instantiating and (re-)wiring the function blocks". Here the the economic threat is mitigated by a technical solution.

However, provision of these very technical quality concerns (e.g., *scalability*, *error reduction*, *maintainability*, etc.) will impose pressure of new skills acquisition upon the workforce, thus undermining employee well-being. These could also require economic investment in terms of staff time and effort (e.g., additional testing for error reduction), as well as tooling and hardware purchase or development, thus negatively affecting economic sustainability.

Effect of the Social Concerns: The seemingly soft notions of establishing organisational culture and supporting skills, as shown in the social sustainability feature model, and confidence of employees have long proven to be amongst the decisive factors of an organisational success [18]. These factors affect productivity (e.g., via *economic/reduce time to market and increase productivity* code) of employees, and so the amount of resources used for a given product (as per *environment/reduce use of resources* code). This is very well demonstrated by the statement from the AGM source that "Training is always a long-term investment but a short-term threat", where social feature of training is interlinked with the economic costs. These also affect the technical quality of the products: for instance, the VLSS study notes that "These automation systems differ significantly regarding their architectures and technologies and the developers require a wide range of technical skills and domain knowledge". Furthermore, the social features have a close interaction with the technical dimension via *technical/quality factors* branch. It has even been shown that [33], after a time, the software architecture reflects the structure of the organisation that produced it.

Effect of the Economic Concerns: We have already noted that SPLE is first and foremost a business model, concerned with the economic sustainability of the host organisation. Thus,

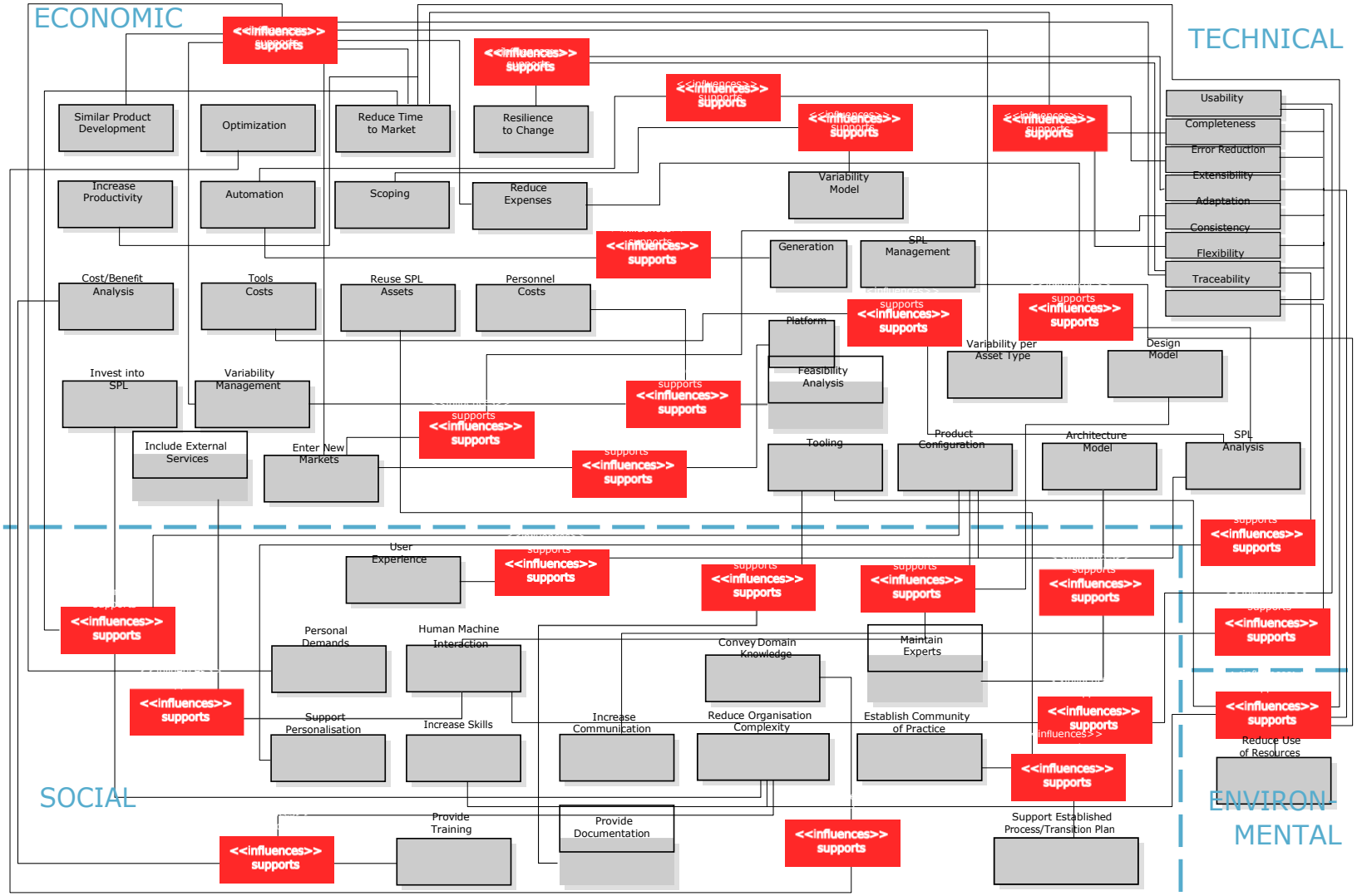

Figure 5. Cross-dimensional dependencies between sustainability features

economic considerations define which (sustainability) features to consider relevant for each given case.

While setting out to adapt an SPL model, the business chooses to implement the *economic/business strategy/transition plan* feature. Which immediately necessitates a social dependency via the *social/establish business culture/transition plan* feature. At the same time, to manage the SPL commonality and variability concerns, the economic feature *economic/business strategy/variability management* necessitates use of technical features of *spl analysis*, *spl modelling*, *spl implementation*, at the very least. All of these, in turn, directly affect the pattern of resource use by the given organisation, i.e., its environmental sustainability. For instance, the Smart Home study notes that "Varying types of houses, different customer demands, the need for short time-to-market and saving of costs drive the need for a Smart Home product line, which is characterized by a wide range of variants." where the *economic/business improvement/reduce expenses* economic feature affects the *technical/spl management/spl implementation/variability per asset type* technical concern.

To summarise, it is clear that there is strong and inseparable coupling between all four above discussed façades of sustainability, and none of these can be taken in isolation, but rather should always be considered within the context of its impact on the others.

3.5.2. Stakeholders and Sustainability Concerns

In addition to cross-dimensional dependencies, the influence and relationships between stakeholders add complexity the interaction between various sustainability features. While it is typically true that a given feature originates from and is owned by a specific stakeholder, the decisions taken with regard to a feature and their consequences would normally affect a larger group of stakeholders. To capture such stakeholder-based interactions, we analysed the case studies for references to stakeholders in sustainability-related context. As a result of this analysis, the sustainability dimensions feature models, presented in Figures 1,2,3and4are annotated with symbols of stakeholders identified within the text of each feature. For this analysis we used the stakeholder classification model proposed by [4].

A number of patterns emerged from this analysis. In the economic sustainability features the upper level management and day to day operators were mentioned predominantly, with a few features also specifically referring to customers or developers. It stands to reason that, from an economic perspective, the major interactions will happen between managers and operators. An inverted but similar pattern can be seen for the technical sustainability features where the developers and users have strong involvement with most features. Managers and operators are in particular involved in analysis tasks that directly affect economic sustainability. Social sustainability features, on the other hand, have the most complex interplay of stakeholders with the organisational branch. This highlights frequent interaction between developers, managers and experts.

It also became apparent during the analysis, that for the majority of the identified features more than one relevant stakeholder could be found. This was to be expected, as sustainability concerns are broad in nature and would affect interests of a number of parties at once.

The value of this information lies in the awareness of potential interactions, especially in combination with the cross-dimensional dependencies described in the previous section. Through the understanding of these dependencies stakeholder interactions and conflicts can be anticipated, explained, and resolved across sustainability dimensions. For example in the AGM case study one of the text fragments stated *"AGM expects that using the software product line approach will allow us to gain market share by offering custom products quicker and at lower cost than the competition. In particular, we expect to be able to have entry level, lower cost, employees build products from the core assets. This will reduce the cost of each product. Eventually we expect to automate production fully so that marketing personnel can build products further lowering costs."*. At first glance this statement carries a clear conflict between the manager and developer stakeholder roles. It seems the primary interest of the manager is to reduce costs by minimizing personnel and expertise required, and from the **Reduce Expenses** feature in Figure1it can indeed be seen that this feature typically relates to managers and developers. However, in this statement the **Reduce Time to Market** feature

Table III. Code validation case studies

Main Focus and Summary
1. *Train Control Case Study (TC)* **Domain:** Train signalling; **Type:** Technology project Here a software product line has been created by applying the Common Variability Language (CVL) on the Train Control Language (TCL). TCL is a domain-specific language to generate code for computers controlling train stations. The case study contains details about the technical realisation of the product line using CVL as well as considerations about how the transition from project engineering to product line engineering can be best performed in this case. Safety issues relevant in the train signalling domain are also discussed. **Reference:** Svendsen et al [47].
2. *Celsius Tech Case Study (CT)* **Domain:** Ship control; **Type:** Technical and a managerial project This is a retrospective analysis of the successes and problems that Celsius Tech (a Swedish company that builds large and complex shipboard command-and-control systems) have experienced during their transfer to an SPL oriented business model. The case study covers a variety of viewpoints - from technical to managerial- and evaluates the results against the goals defined before the SPL approach was implemented. The study has a special focus on economic viability and the market place and organisational benefits the SPL approach has yielded for Celsius Tech. **Reference:** http://www.sei.cmu.edu/productlines/casestudies/celsiustech/

is also relevant which according to Figure5directly relates to the **Increase Skills** feature in Figure 2. This highlights to the planners of AGM that while developer effort can be reduced from product creation, their expertise likely is needed in other areas and skill training might be required. The Increase Skills feature in Figure2further suggests the involvement of an external expert stakeholder to aid in resolving this conflict.

4. VALIDATING THE MODEL

We now consider whether the feature models constructed in Section3can be considered adequately representative of other published SPLE work for sustainability code set, and discuss concerns that could threaten the validity of these models.

4.1. Validating Code Set

As discussed in Section2, in accordance with the analysis methodology adopted for this paper, we continued coding until saturation of the set of emergent codes. As shown in TableII, saturation seemed to be achieved by completing coding for six case studies. The last two studies, SPA and IEC, provided only one and zero new codes respectively. At this no substantially new information emerged. Hence we stopped the coding of new texts and completed the analysis discussed in Section 3.

We now set out to validate the code set and categories through the analysis of two additional case studies outlined in TableIII. We selected these case studies for validation using the previously discussed principles: they should be operating in different domains (train signalling and ship control) than the case studies used for the initial analysis, and be publicly available. Both these studies were again coded and analysed as per previously outlined process (see section2.2).

TableIVpresents codes that have been added during the study of the TC and CT case studies.

Table IV. Code validation case studies

	Train Control	**Celsius Tech**
Economic	2 economic/business: + business improvement/reduce time to market/increase productivity/optimization; + business strategy/define SPL business model/scoping;	1 economic/business: + business improvement/improve process predictability;
Technical	3 technical/quality factors: + completeness; + techniques/validation; + techniques/verification;	1 technical/quality factor: + predictability;
Total	**5**	**2**

In coding of the **Train Control** study, 106 codes were assigned in total of which five were not yet part of our model. Two new codes were added to the economic code sub-set at levels three and four respectively. These relate to the optimisations to increase productivity (business improvement/reduce time to market/increase productivity/optimization) and scoping as a way to define the SPL business model (business strategy/define spl business model/scoping). Three new codes were added in the technical category refining the set of quality factors (*completeness*, *validation*, and *verification*).

Examples of existing categories assigned from our feature model include, for instance, technical categories that contain automation and generation issues, variability modelling, domain-specific languages, product line analysis and implementation, and different quality factors. Example of assigned economic categories include reducing time to market by increasing productivity and similar product development, business process maturity, and reducing costs by lowering development time. Social issue codes used in this case include maintaining experts and reducing organisation complexity by defining clear roles. Environmental sustainability was not considered as an issue in the Train Control case study.

In coding of the **Celsius Tech** study, 69 codes were assigned in total of which two were not yet part of our model. One new code was added to the economic part, which focused on business improvement through moving the company towards SPL practice (business improvement/improve process predictability, see TableIII). In addition a code was added to level three of the technical qualities feature to improve *predictability* of the development process, which correlates to the code introduced in the economic feature as well. Example existing categories that were used in this model include such technical categories as SPL analysis and commonality-variability analysis, economic codes to reduce time to market, lower costs, increase software quality and entering new markets. Social issues codes used here include personal transition plan, SPL champions and the definition of clear roles. Again, environmental aspects have not been discussed in the Celsius Tech case study.

In summary, as per TableIV, we see that as part of this validation exercise, a total of seven new codes were introduced, which were either at level three, four or five of the feature models (sub-categories). Thus, the new codes only added further refinement to our feature models. Hence, though we do not claim in any way that these feature models are complete, we are satisfied that the code set (embodied in the above feature models) is sufficiently representative of the current state of how the sustainability topics are addressed within the SPL domain.

4.2. Threats to Validity

As with any model that is derived from data, there are a number of factors that can affect its quality and validity [21]. For the above discussed analysis, we now discuss how four such factors - *construct validity*, *internal validity*, *external validity*, and *reliability* - have been addressed.

Construct validity is a serious threat factor for the validity of this type of text analysis, as data may be misinterpreted by the analyst. To mitigate this threat we ensured that the work of each coder was cross-validated by two other analysts. Coding of the first three case studies was conducted with all three coders participating. The following case studies were coded individually by one of the analysts, then cross-validated by the two others. Furthermore, to ensure a common baseline for the relevant concepts and terms, we referred to background literature and definitions.

Confounding factors influencing the analysis are a major threat to **internal validity**. To mitigate this threat, we followed a clear research process (documented in Section 2) and used commonly accepted, scientific methods of qualitative content analysis, as detailed in [35,23]. Additionally, we have selected the documents analysed in this study with an aim to ensure that they come from different domains and have different focus, thereby contributing to a wider spread of content and lower chance of confounding factors influencing the end result. Yet, we are aware that, despite these efforts, confounding factors cannot be ruled out completely.

Considering **external validity**, the cases presented here are not claimed to be statistically representative. This is a qualitative study, and statistical generalisation is not claimed or aimed for. Instead, we are concerned with analytical generalisation [50]. Our explorative qualitative study is helping us to identify current concerns relevant to sustainability that are present in today's published SPL practice and research. By selecting documents from different application domains, countries, and companies as well as different types of case studies from technical studies to interview studies and pedagogical studies, we aimed to collect a rich data set.

To mitigate threats to **reliability due to interpretation** in qualitative analysis, coding was done first in a team (for first 3 studies) and then (for remaining studies) cross-validated by two additional analysts. Any mapping disagreements were discussed until consensus was reached. To ensure that the study is reproducible by other researchers, we ensured that all used materials are publicly available from reputable sources, described each case study, and also included the references to all the case studies used in our analysis.

5. DISCUSSION

The analysis described in this paper has lead to a first systematic representation of sustainability concerns in SPL as covered in Section 3. The core concepts and their relations are captured in feature models depicted in Figures 1,2,3,4,5. The models provide a hierarchical representation of the identified sustainability concerns. Here we discuss several additional issues related to the structure of our models, their practical utility, and relevance of sustainability to SPLE.

5.1. Practicality of the SPL Sustainability Feature Models

Today in SPLE community there still is a prevalent opinion that sustainability is only relevant to customers, and "If the customers do not ask for sustainability, we do not need to do anything about it". This opinion persists primarily due to poor understanding of what sustainability is, and how it matters to SPLE. Since the sustainability models presented in this paper emerged from analysis of recent, real SPL projects, they demonstrate that not only are various dimensions of sustainability practically relevant, but they are already key drivers of SPLE practice and process:

- economic sustainability is the motivator of adoption of the SPLE as a business model;
- technical sustainability is the motivator of adoption of the SPL architecture;
- social sustainability is the motivator of research and practice on the SPL business process, organisational culture, and alike.
- finally, environmental sustainability, while currently largely overlooked, has a massive impact on all other areas of the SPL business.

While, by construction, the presented models are firmly rooted in the SPLE practical data, further feedback and evaluation of their utility and completeness is certainly desirable. Qualitative and

quantitative evaluation of these models with the SPLE practitioners and sustainability experts will thus form our immediate future work.

Meanwhile, to companies currently practising or considering adoption of SPLE, these models could serve as a starting point for reflection upon own priorities on sustainability concerns. Similar to traditional feature-based product configuration, any SPL project could consider how these models would be instantiated and expanded in its intended socio-technical setting. These models could:

- serve as a guide on the current state of the sustainability concepts in SPL;
- act as the minimum target to achieve towards a specific façade of sustainability, to be on par with the current practice;
- stimulate identification and treatment of additional (currently missed out of the model) sustainability concepts that expand the model, while also expanding own practice of sustainability.
- aid in the identification and analysis of stakeholder concerns within and across sustainability dimensions.

5.2. Sustainability as a topic of interest in SPL: Metrics Review

Here we further review the question of relevance of sustainability topic to the SPL community from the metrics perspective. We observe that if an issue is considered to be of relevance to the SPL community, the community will have metrics developed for it, as we only measure what we are interested in. Thus, we (very briefly) review the metrics currently in use in SPL and note how they relate to the sustainability concerns. Through this exercise we also identify areas of sustainability that have not been considered as key topics in SPL so far, or, at least, have not been interesting enough to have them measured and monitored. We then relate the findings from the metrics review to our feature trees.

5.2.1. Metrics Related to Technical Sustainability

Within SPL development traditional software metrics are used to assess the quality of the software [22,3], such as (i) number of lines of code, (ii) cyclomatic complexity, (iii) depth of inheritance. These are complemented with metrics specifically tailored for measuring the technical ability of the SPL to support the production process, such as: (i) core asset utility, (ii) percent reuse, (iii) specialised SPL maintainability metrics [5].

All these metrics clearly relate to the ability of the technical assets to "keep up" and provide useful service, i.e., their technical sustainability. With respect to our feature models (as per Figures 1,2,3and4), the metric of cyclomatic complexity directly measures the *Complexity* concern; lines of code measures *Implementation*, and depth of inheritance relates to *Maintainability*; moreover, all these closely relate to *Scalability* and *Evolution*.

5.2.2. Metrics Related to Economic Sustainability

The main goal of SPL business model lies in reducing the business cost and increasing its productivity. This goal does, indeed, coincide with that of the Economic Sustainability. The currently used metrics that focus on SPL performance include [51], for instance (i) total product development cost, (ii) time to market, (iii) market feature coverage. Other metrics measure a more indirect contribution focusing on how the development process itself has been streamlined. The underlying assumption for these metrics is that an improved production process will drive production costs for individual software products down, which in turns repays the investment for creating the streamlined production process. These include, e.g., (i) effort to produce core assets, (ii) core asset utility, (iii) percent reuse.

All these metrics directly relate to the topics of *Optimisation*, *Increase Productivity*, and *Increase Software Quality*, as depicted under the Economic Sustainability concern in Figure1.

5.2.3. Metrics Related to Social Sustainability

We review the metrics related to organisational and personal aspects of SPLs separately. The SPL metrics related to the organisational aspect seem to focus on such issues as process compliance, or

revert back to issues closely lined with the financial gain [51]. On the one hand, this correlation is logical as a process support is essential for a smooth operation of an organisation, and reduced costs in software development translate to continuity of the organisational existence. The topics that have emerged from our study as relevant to the organisation's sustainability are, for instance, *Increase Communication* (e.g., support traceability, provide documentation) and *Establishing a Community of Practice* (see Figure2). Such topics relate to the process compliance metrics of the SPL.

Turning to the metrics related to the Personal sustainability, we observe that these are often centred around the challenges faced by the developers in utilising the SPL infrastructure. This same observation emerged from our study, where the individuals were considered only as *employees* and related to such topics as *Increase Skills* or *Support Tool Personalisation*. In truth, such metrics and topics fall into either the economic or technical areas and are focused on making human capital or an enterprise more productive. The human side of the employees is missed, with such issues (to name a few) as personal job satisfaction, self worth, or employee equality missed out entirely.

Thus, as the topics related to the human side of the organisation and employees, are missed both in all our study results and in this brief metrics review, we suggest that the entire notion of human sustainability is an open question to be addressed by SPL community.

5.2.4. Metrics Related to Environmental Sustainability

The most notable conclusions from our analysis is that the area of environmental sustainability has received very little attention within the SPL community. Our analysis identified only one code directly related to the environmental concerns in this study. Nevertheless, as noted above, SPL does have direct impact on the environment, e.g., through resource consumption, or through work process change. When examining metrics that specifically address environmental sustainability of software development, such issues as energy efficiency of software [24] or impact of software architectural choice on resource consumption and emissions [15] have been considered. It is clear that similar issues would directly relate to the SPL community, and should be further researched on within the SPL context.

In summary, as reflected in the above discussion, we find that the review of the metrics currently use in the SPL community conveys the same message as we have observed form the feature tree construction process. That is: economic, technical, and organisational aspects of sustainability are considered relevant, and are measured, and monitored through dedicated metrics, with human and environmental aspects left largely unattended.

6. RELATED WORK

6.1. Sustainability and SPL

Currently, there is relatively little work on how sustainability and software product line engineering fit together. In previous work [13] we have started a discussion within the SPLE community on what sustainability means for SPL, and presented very initial ideas on identification of sustainability concerns within the SPL [14] using only the case of DiVA study. The present paper extends our initial work [14] through analysis of a much more substantial corpus of (10) case studies, which allows for us to arrived at a relatively saturated codeset that represents our sustainability feature models.

Lutz et al. [32] adopt techniques from SPLE for the design and operation of long-lived, sustainable systems (LSS). LSS have an extended lifetime, make efficient use of resources, and are highly adaptable. This work deals with knowledge preservation during system changes with the help of SPLE. Voyager Spacecraft is used as an example of an LSS. Lutz et al. focus mostly on technical sustainability.

Savolainen et al. [44] discusses how SPLs can be built in a sustainable way. They propose a model of planned staged investments with two phases (investment and harvesting) that ensures long-living product lines. Savolainen et al. mainly focus on evolution and technical sustainability of product lines but also discuss how the organization can deal with switching between the two phases.

6.2. *Sustainability and SE*

While the concept of sustainability is a relatively new occurrence in SPL development, in recent years, it has received more notable attention in the wider field of software engineering [38,37]. According to a systematic literature review by Penzenstadler et al [38], the first publications in this area started to surface from 2006 onwards with a steady increase in recent years. The areas were sustainability approaches have been proposed, include, for instance, green decision making processes [25,44,43,6], requirements engineering [11,34,8], architectural patterns [29], and use of resources [31,36,27].

Becker et al. [6] argue that it is the requirements engineering (RE) stage of SE that has the most substantial contribution in ensuring that software supports sustainability of its situated communities, as well as its own sustainability. This is because it is in RE that the decisions are taken about what matters to, and thus should be implemented in, a given software system.

Durdik et al. [20] present a catalogue of software sustainability guidelines for reaching the goal of economic sustainability during system evolution. Seacord et al. [45] present measures to evaluate sustainability of software systems. Sustainability in this case focuses on software maintenance and evolvability. Koziolek [26] assesses the capabilities of existing architecture evaluation methods with respect to their support for measuring the sustainability of a software architecture. Chitchyan et al. [12] discuss how programming languages can be designed to support technical sustainability of software systems. Again, sustainability here is limited to evolvability, maintenance, and long-life of software systems.

A recent study [10] discusses the current perception of sustainability within the requirements engineering practitioners. It is interesting to note that here the practitioners generally perceive sustainability as mainly pertaining to environmental and economic concerns in private life, but do not directly link these notions to their work as software practitioners. This perceived environmental disconnect from software systems correlates with our above discussed findings of largely ignored significance of environmental concerns in SPLE.

In [38] an integrated definition of sustainability in and for software engineering has been suggested, that highlights four distinct aspects of sustainability during system development, maintenance, production, and use.

All the above discussed notions strongly correlate with the findings in our models, even while it decomposes sustainability from a software developer perspective. For example development and maintenance process of [38] directly correlate to the technical branch of our model, as well as elements of our economic branch regarding maintenance cost, delivery time, etc. Additionally, our model introduces specific SPLE sustainability concerns and activities, such as the economic and technical impact of reusable SPL components, the requirements to support a sustainable SPL infrastructure, and the social implications for SPL sustainability, such as team roles and training. On the other hand the environmental branch of our model contains considerably less concepts and detail than the current state of the art in sustainability in software engineering, highlighting the lack of attention this topic has received in SPL development. It is notable that the human concerns, where humans are considered as fully formed individuals, rather than only employees, is also missing in the work on SE in general.

7. CONCLUSIONS

Sustainability, i.e. living within the bounds of the available environmental, social, and economic resources, is becoming an increasingly more prominent topic in present-day social and scientific discussions. Within the field of software engineering this increased attention has lead to such initiatives and approaches as more energy- and other resource-efficient software, and better insights into sustainable software development from a social and economic perspectives.

It can be argued that Software Product Lines (SPL) were initially conceived from a desire to increase the economic sustainability of software development. By utilising structured reuse and extensibility through SPL, the cost of production and time to market are contained, leading

to leaner and more flexible software process and products. However, sustainability concerns in SPL engineering can and need to be considered beyond this traditional economic context. To further the practice of sustainability in all its facets with regards to SPLE, a more in depth understanding is required of how sustainability is currently perceived, which concepts are currently under-represented, and how this relates to the wider field of research on sustainability in software engineering.

In this paper we have undertaken an in-depth analysis of a set of Software Product Line studies to identify which sustainability related concepts are currently considered as part of SPLE. We have applied text analysis, to create an inventory of sustainability concerns using eight previously published SPL studies which resulted in a hierarchy of sustainability concerns represented by four feature trees, each corresponding to one of the four major sustainability concerns (economic, social, technical and environmental sustainability). Two further SPLE studies were used to validate how well these feature trees cover the sustainability concepts currently handled in SPLE. The results of this validation illustrated that our SPLE sustainability feature models can be considered sufficiently representative. In addition we have identified and visualised dependencies across sustainability dimension that highlight the interaction between sustainability features as well as stakeholders from which those features originate. These models can aid in the exploration and identification of conflicting sustainability and stakeholder concerns as well as their resolution.

Thus, we have noted that the SPL community has long engaged with topics directly focused on economic, technical, and organisational sustainability, and even some (very limited topics on) environmental sustainability concerns. This is also confirmed by our brief review on the wider SPL metrics, where we observe existence of a number of metrics relevant to economic, technical, and organisational sustainability. However, all such metrics are used in silos, not considered or measuring sustainability as a whole. Since sustainability dimensions are deeply interdependent (as discussed in this paper), such segregated view on and optimisation of one aspect of sustainability inevitably undermine one or more of the other aspects. Indeed, topics related to environmental and human sustainability have, so far, been largely left out of the SPLE domain; and these are the very aspects of sustainability that, up to now, software in general and product lines in particular, have also disregarded (and often undermined). These, then, are the topics that we hope the SPLE community will research on further.

Hence, the set of sustainability feature models presented in this paper provide a first step towards the systematic consideration of sustainability in SPL development. Continuing on with this work we plan to perform a qualitative evaluation of the accuracy and usefulness of the models together with experts from industry and academia, as well as extending and completing the models with a specific focus on the environmental dimension of sustainability. However most importantly with this work we aim to raise awareness and make sustainability a more visible and prominent concern in SPLE development helping to inform the conversation on social, environmental, and economic concern manifestations and interdependencies within SPLE and, consequently, the discipline of Software Engineering as a whole. With the increased attention and consideration of sustainability concerns in this field, our models can evolve and become a guide for developers in the specific and unique challenges that come with building sustainable Software Product Lines.

REFERENCES

1. Data and analysis files, 2016. Online resource: http://seg.cmp.uea.ac.uk/files/study-files.zip (retrieved 15-8-2016).
2. Intergovernmental panel on climate change publications and data, 2016. Online resource: https://www.ipcc.ch/publications_and_data/publications_and_data.shtml/ (retrieved 20-12-2016).
3. Z. Ahmed. Towards performance measurement and metrics based analysis of pla applications. *arXiv preprint arXiv:1007.5127*, 2010.
4. I. Alexander. A taxonomy of stakeholders, human roles in system development. *Issues and Trends in Technology and Human Interaction*, pages 25–71, 2006.
5. E. Bagheri and D. Gasevic. Assessing the maintainability of software product line feature models using structural metrics. *Software Quality Journal*, 19(3):579–612, 2011.
6. C. Becker, S. Betz, R. Chitchyan, L. Duboc, S. M. Easterbrook, B. Penzenstadler, N. Seyff, and C. C. Venters. Requirements: The key to sustainability. *IEEE Software*, 33(1):56–65, 2016.

7.C. Becker, R. Chitchyan, L. Duboc, S. Easterbrook, B. Penzenstadler, N. Seyff, and C. C. Venters. Sustainability design and software: The karlskrona manifesto. In *37th IEEE/ACM International Conference on Software Engineering, ICSE 2015, Florence, Italy, May 16-24, 2015, Volume 2*, pages 467–476, 2015.
8.C. Bonfim, W. Nunes, L. Duboc, and M. Schots. Modelling sustainability in a procurement system: An experience report. In *IEEE 22nd International Requirements Engineering Conference, RE 2014, Karlskrona, Sweden, August 25-29, 2014*, pages 402–411, 2014.
9.J. Cabot, S. Easterbrook, J. Horkoff, L. Lessard, S. Liaskos, and J. N. Mazon. Integrating sustainability in decision-making processes: A modelling strategy. In *Software Engineering - Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference on*, pages 207–210, 2009.
10.R. Chitchyan, C. Becker, S. Betz, L. Duboc, B. Penzenstadler, N. Seyff, and C. C. Venters. Sustainability design in requirements engineering: State of practice. In *38th IEEE/ACM International Conference on Software Engineering, ICSE 2016, Austin, USA, May 14-222, 2016, Volume 2*, page to appear, 2016.
11.R. Chitchyan, S. Betz, L. Duboc, B. Penzenstadler, S. Easterbrook, C. Ponsard, and C. Venters. Evidencing sustainability design through examples. 2015. Fourth International Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy).
12.R. Chitchyan, W. Cazzola, and A. Rashid. Engineering sustainability through language. In *Proceedings of the 37th International Conference on Software Engineering-Volume 2*, pages 501–504. IEEE Press, 2015.
13.R. Chitchyan, J. Noppen, and I. Groher. Sustainability in software product lines. In *Proceedings of the 18th International Software Product Line Conference - Volume 1*, SPLC '14, pages 367–367, New York, NY, USA, 2014. ACM.
14.R. Chitchyan, J. Noppen, and I. Groher. What can software engineering do for sustainability: case of software product lines. In *Proceedings of the Fifth International Workshop on Product LinE Approaches in Software Engineering*, pages 11–14. IEEE Press, 2015.
15.R. Chitchyan, A. Obeid, and H. Janicke. Study of architectural impact on software sustainability. 2014. Third International Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy).
16.R. Chitchyan, A. Rashid, P. Rayson, and R. Waters. Semantics-based composition for aspect-oriented requirements engineering. In *Proceedings of the 6th International Conference on Aspect-Oriented Software Development, AOSD 2007, Vancouver, British Columbia, Canada, March 12-16, 2007*, pages 36–48, 2007.
17.P. Clements and L. Northrop. *Software Product Lines: Practices and Patterns*. Addison-Wesley Professional, 2001.
18.A. De Jong, K. De Ruyter, and M. Wetzels. Linking employee confidence to performance: A study of self-managing service teams. *Journal of the Academy of Marketing Science*, 34(4):576–587, 2006.
19.D. Dhungana, P. Gr ünbacher, and R. Rabiser. The DOPLER meta-tool for decision-oriented variability modeling: a multiple case study. *Autom. Softw. Eng.*, 18(1):77–114, 2011.
20.Z. Durdik, B. Klatt, H. Koziolek, K. Krogmann, J. Stammel, and R. Weiss. Sustainability guidelines for long-living software systems. In *Software Maintenance (ICSM), 2012 28th IEEE International Conference on*, pages 517–526. IEEE, 2012.
21.S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian. *Guide to Advanced Empirical Software Engineering*, chapter Selecting Empirical Methods for Software Engineering Research, pages 285–311. Springer London, London, 2008.
22.N. E. Fenton. *Software Metrics: A Rigorous Approach*. Chapman and Hall, 1991.
23.B. G. Glaser and A. L. Strauss. *The discovery of grounded theory: Strategies for qualitative research*. Transaction Publishers, 2009.
24.F. Heliot, M. A. Imran, and R. Tafazolli. Near-optimal energy-efficient joint resource allocation for multi-hop mimo-af systems. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on*, pages 943–948. IEEE, 2013.
25.F.-H. Huang, Y.-L. Lee, and S.-L. Hwang. E-shopping behavior and user-web interaction for developing a useful green website. In *Human-Computer Interaction. New Trends*, pages 446–454. Springer, 2009.
26.H. Koziolek. Sustainability evaluation of software architectures: a systematic review. In *Proceedings of the joint ACM SIGSOFT conference–QoSA and ACM SIGSOFT symposium–ISARCS on Quality of software architectures–QoSA and architecting critical systems–ISARCS*, pages 3–12. ACM, 2011.
27.P. Lago, R. Kazman, N. Meyer, M. Morisio, H. A. M üller, and F. Paulisch. Exploring initial challenges for green software engineering: Summary of the first greens workshop, at icse 2012. *SIGSOFT Softw. Eng. Notes*, 38(1):31–33, Jan. 2013.
28.P. Lago, S. A. Koçak, I. Crnkovic, and B. Penzenstadler. Framing sustainability as a property of software quality. *Commun. ACM*, 58(10):70–78, Sept. 2015.
29.G. A. Lewis and P. Lago. Architectural tactics for cyber-foraging: Results of a systematic literature review. *Journal of Systems and Software*, 107:158–186, 2015.
30.F. J. v. d. Linden, K. Schmid, and E. Rommes. *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.
31.J. Liu, T. Feng, and X. Yang. The energy requirements and carbon dioxide emissions of tourism industry of western china: A case of chengdu city. *Renewable and Sustainable Energy Reviews*, 15(6):2887–2894, 2011.
32.R. Lutz, D. Weiss, S. Krishnan, and J. Yang. Software product line engineering for long-lived, sustainable systems. In *Software Product Lines: Going Beyond*, pages 430–434. Springer, 2010.
33.A. MacCormack, C. Baldwin, and J. Rusnak. Exploring the duality between product and organizational architectures: A test of the mirroring hypothesis. *Research Policy*, 41(8):1309–1324, 2012.
34.M. Mahaux, P. Heymans, and G. Saval. Discovering sustainability requirements: an experience report. In *Requirements engineering: foundation for software quality*, pages 19–33. Springer, 2011.
35.P. Mayring. Qualitative content analysis. *Qualitative Social Research*, 1(2), 2000. (online journal).
36.S. Naumann, M. Dick, E. Kern, and T. Johann. The greensoft model: A reference model for green and sustainable software and its engineering. *Sustainable Computing: Informatics and Systems*, 1(4):294–304, 2011.

37.B. Penzenstadler. Towards a definition of sustainability in and for software engineering. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pages 1183–1185, New York, NY, USA, 2013. ACM.
38.B. Penzenstadler, V. Bauer, C. Calero, and X. Franch. Sustainability in software engineering: A systematic literature review. In *Evaluation Assessment in Software Engineering (EASE 2012), 16th International Conference on*, pages 32–41, May 2012.
39.B. Penzenstadler and H. Femmer. A generic model for sustainability with process- and product-specific instances. In *Proceedings of the 2013 Workshop on Green in/by Software Engineering*, GIBSE '13, pages 3–8, New York, NY, USA, 2013. ACM.
40.K. Pohl, G. B öckle, and F. J. v. d. Linden. *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.
41.A. Rashid, A. M. D. Moreira, and J. Ara újo. Modularisation and composition of aspectual requirements. In *AOSD*, pages 11–20, 2003.
42.A. Rashid, J.-C. Royer, and A. Rummler. *Aspect-Oriented, Model-Driven Software Product Lines: The AMPLE Way*. Cambridge University Press, New York, NY, USA, 2011.
43.K.-H. Rob èrt, B. Schmidt-Bleek, J. A. De Larderel, G. Basile, J. L. Jansen, R. Kuehr, P. P. Thomas, M. Suzuki, P. Hawken, and M. Wackernagel. Strategic sustainable developmentselection, design and synergies of applied tools. *Journal of cleaner production*, 10(3):197–214, 2002.
44.J. Savolainen, N. Niu, T. Mikkonen, and T. Fogdal. Long-term product line sustainability with planned staged investments. *Software, IEEE*, 30(6):63–69, 2013.
45.R. C. Seacord, J. Elm, W. Goethert, G. A. Lewis, D. Plakosh, J. Robert, L. Wrage, and M. Lindvall. Measuring software sustainability. In *null*, page 450. IEEE, 2003.
46.R. Suddaby. From the editor: What grounded theory is not. *Academy of Management Journal*, 49(4):633–642, 2006.
47.A. Svendsen, X. Zhang, R. Lind-Tviberg, F. Fleurey, Ø. Haugen, B. Møller-Pedersen, and G. K. Olsen. *Software Product Lines: Going Beyond: 14th International Conference, SPLC 2010, Jeju Island, South Korea, September 13-17, 2010. Proceedings*, chapter Developing a Software Product Line for Train Control: A Case Study of CVL, pages 106–120. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
48.J. A. Tainter. Social Complexity and Sustainability. *Ecological Complexity*, 3:91–103, Feb. 2006.
49.M. Vierhauser, R. Rabiser, and P. Gr ünbacher. A case study on testing, commissioning, and operation of very-large-scale software systems. In *Companion Proceedings of the 36th International Conference on Software Engineering*, ICSE Companion 2014, pages 125–134, New York, NY, USA, 2014. ACM.
50.R. K. Yin. *Case study research: Design and methods*. Sage publications, 2013.
51.D. Zubrow and G. Chaslek. Measures for software product lines. Technical report, Software Engineering Institute, 2003. CMU/SEI-2003-TN-031.