EasyChair Preprint

№ 762

Towards a Meta-Model for Requirements-Driven Information for Internal Stakeholders

Ibtehal Noorwali, Nazim Madhavji, Darlan Arruda and Remo Ferrari

January 31, 2019

Towards a Meta-Model for Requirements-Driven Information for Internal Stakeholders

Ibtehal Noorwali[1], Nazim H. Madhavji[1], Darlan Arruda[1], Remo Ferrari[2]

[1] Department of Computer Science, University of Western Ontario, London, ON, Canada
inoorwal@uwo.ca, madhavji@gmail.com, darruda3@uwo.ca
[2] Siemens Mobility, New York, USA
remo.ferrari@siemens.com

Abstract. [Context & Motivation] Providing requirements-driven information (e.g., requirements volatility measures, requirements-design coverage information, requirements growth rates, etc.) falls within the realm of the requirements management process. The requirements engineer must derive and present the appropriate requirements information to the right internal stakeholders (IS) in the project. **[Question / Problem]** This process is made complex due to project-related factors such as numerous types of ISs, varying stakeholder concerns with regard to requirements, project sizes, a plethora of software artifacts, and many affected processes. However, there is little guidance in practice as to how these factors come into play together in providing the described information to the ISs. **[Principle ideas/results]** Based on analyzed data from an action research (AR) study we conducted in a large systems project in the rail-automation domain, we propose a meta-model that consists of the main entities and relationships involved in providing requirements-driven information to internal stakeholders within the context of a large systems project. The meta-model consists of five main entities and nine relationships that are further decomposed into three abstraction levels. We validated the meta-model in three phases by researchers and practitioners. [**Benefits/Contribution]** The meta-model is anticipated to facilitate: (i) control and management of process and resources for providing requirement-driven information to stakeholders and (ii) communication among internal stakeholders.

Keywords: Requirements engineering, requirements management, requirements metrics, meta-model, internal stakeholders, empirical study.

1 Introduction

Context. The requirements engineering (RE) process and resultant requirements usually inform and interact with downstream (e.g., design and testing), upstream (e.g., contract management), and side-stream (e.g., project and quality management) processes in various ways. Each of these processes involves numerous internal stakeholders (e.g., managers, developers, architects, etc.) who, in turn, have different concerns with regard to the impact of requirements on their respective processes. In

other words, the various stakeholders need different types of requirements information in order for them to manage, control, and track their respective process activities (e.g., requirements engineer: measures that track and monitor requirements growth; architect: requirement-design coverage information; systems manager: percentage of requirements dropped per release; etc.) [1–4]. The burden of providing this information (hereon, "requirements-driven information"), generally falls within the realm of the requirements management process [5, 6].

Problem. To this end, we conducted an action research (AR) study in a large systems project in the rail-automation domain to derive requirements-driven information that can be used by the project's internal stakeholders (IS) (see section 3.2). However, we found it difficult for requirements engineers to derive and provide the various internal stakeholders with the correct requirement-driven information that addresses their various concerns due to a lack of understanding of: i) the type of information that can be generated from system requirements, ii) who the ISs are that would benefit from information generated from system requirements, iii) the concerns of ISs which can be addressed by providing requirement-driven information, iv) how the ISs use that information to address their various concerns, and v) the type of artifacts needed to derive the requirements-driven information. This problem is also mirrored in the scientific literature (discussed in more detail in Section 2).

Principle Idea. To address the problem we experienced in industry we ask the following research questions: **RQ1.** What are the types of entities involved in the process of providing requirements-driven information to ISs in a large systems project? **RQ2:** What are the relationships that exist among the entities involved in the process of providing requirements-driven information to ISs in a large systems project? To answer the research questions, we performed a post-analysis on the data gathered from the AR study we conducted in industry (see section 3.1). The result of the post-analysis is a meta-model that maps out the entities and relationships involved in providing requirements-driven information to ISs. The anticipated benefits of using the meta-model include i) control and management of processes and resources involved in providing requirement-driven information to ISs and ii) communication among ISs.

Contributions. The key contributions of this paper are: (i) descriptions of the entities involved in providing requirements-driven information to ISs, (ii) descriptions of the relationships among the identified entities, (iii) an empirically derived meta-model that combines the identified entities and relationships, and (iv) a discussion of the meta-model and its implications on industry and research.

Paper Structure. Section 2 describes related work; Section 3 describes the research methods; Section 4 presents the meta-model with a detailed description; Section 5 discusses the validation procedures and threats to validity; Section 6 discusses implications of the meta-model, and Section 7 concludes the paper and describes future work.

2 Related Work

This section focuses on three key issues in RE meta-models: (a) ISs and their concerns, (b) requirements-driven information, and (c) relationships among the

preceding two items. With respect to IS, the literature lacks a comprehensive understanding of the types of ISs that can exist in large systems engineering projects and their concerns regarding requirements. Though the term "stakeholders" is well-known in RE, in-depth research has focused on external stakeholders (i.e., client/customer, business) and their concerns, which are usually translated into new requirements [7], while the concerns of ISs (e.g., project managers, architects, etc.) are rarely addressed [8]. In the rare cases in which ISs are addressed, the problem is two-fold: 1) they focus on developer concerns only (e.g., source code defect analytics) [8], or 2) the stakeholders are roughly divided into generic notions of "developer" and "manager" [1]. However, our observation is that ISs and their concerns exist at a finer granularity (e.g., different types of managers, technical stakeholders, and concerns).

In addition, requirements-driven information is usually limited to requirements quality metrics (e.g. use case completeness metrics) [9] and basic progress metrics (e.g., number of 'complete' requirements) [6, 10] that do not specifically address the concerns of the spectrum of internal stakeholders within a project.

Finally, the relationships amongst: (i) internal stakeholders, (ii) stakeholder concerns, and (iii) requirements-driven information have not yet received significant research attention. Thus, managing these elements to derive requirements-driven information from the correct sources and providing it to the correct ISs becomes a tedious task in practice.

From the above analysis, to our knowledge, a model to support the requirements management task of deriving and reporting requirements-driven information to internal stakeholders is currently lacking. The remainder of this paper addresses this gap.

3 Research Method

The meta-model presented in this paper is a result of a post analysis performed on data gathered from an AR study we conducted in industry. Fig. 1 provides an overview of the research methods and data used in this study. The following subsections discuss the data gathering and data analysis stages in detail.

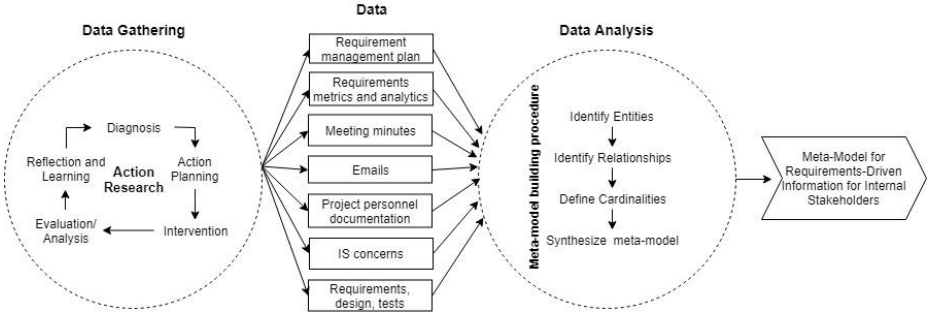

Fig. 1. Overview of Study Research Methods and Data

3.1 Data Gathering: Action Research Study

Action research (AR) is an iterative process involving researchers and practitioners acting together on a particular cycle of activities, including problem diagnosis, action planning, intervention/action taking, evaluation, and reflection/ learning [11], where researchers identify problems through close involvement with industrial projects, and create and evaluate solutions in an almost indivisible research activity. We note that, because the goal of the AR study was to derive requirements-driven information (not reported in this paper) to be used by the ISs, we limit our description of the AR procedure to details relevant to the meta-model and its underlying constructs.

Table 1. Software Artifact Breakdown per Product

Product	**# of Req. Spec. Docs.**	**# of Reqs.**	**# of Design Docs.**	**# of Design Objects**	**# of Test Cases**
Product 1	40	59335	23	8373	1770
Product 2	13	25502	3	1199	960
Product 3	37	50051	28	24618	827
Total	90	134888	54	34190	3557

Our AR study, which followed the described approach [11], was conducted in a large-scale rail automation project in a multi-national company in the United States. The overall project (i.e., program) consisted of three sub-projects, each sub-project consisted of a product that had its own set of requirements, architecture design, test cases, and engineering team. Table 1 shows a breakdown of the software artifacts, number of requirements, design objects, and test cases per product that the first author worked with. Other official project documents that were analyzed included: requirements and change management plans and project personnel documentation that describe the roles and responsibilities of the ISs involved in the projects. The project adopted a waterfall software development approach. The internal project stakeholders included: systems manager, R&D managers, test mangers, developers, architects, testers, project managers, program managers, safety managers, quality mangers, financial managers, and project operations managers.

The AR study began in February 2017. The primary researcher (1st author) was onsite full-time for ten months and worked with the primary industrial partner (4th author) and secondary industrial participants (internal project stakeholders) in consultation with a senior researcher (2nd author).

In the diagnosis phase, the primary researcher, primary industrial partner and senior researcher, through a series of unstructured interviews, found that a central problem in the projects' RE process is a difficulty in tracking, monitoring, and managing requirements-driven information such as requirement growth (e.g., how many requirements so far), volatility (e.g., number of changed requirements over releases), and coverage (e.g., number of requirements that have been covered by test and design) and a difficulty in in accessing this information by ISs. To solve this problem, the industrial partner and researcher conducted several meetings, as part of the action planning phase, and decided to derive, define, and validate a set of requirements metrics

and analytics that can be provided to ISs and used within the requirements management and software development processes. The requirements-driven information would include: measures on requirements size, growth, coverage, volatility, safety requirements distribution.

In the intervention phase, the primary researcher, with continuous feedback from the primary industrial partner, conducted a document analysis on the requirements, design, and test documents in which the meta-data were gathered in spreadsheets and the completeness and consistency of the data were ensured (see Table 1). The researcher then used the gathered meta-data to define a set of metrics using GQM+ [12] (not reported in this paper). The measures for the different products (see Table 1) were calculated and organized in spreadsheets and graphs. To familiarize the ISs with the derived information and to gather feedback from them, three iterations of focus groups were held. IS feedback included suggestions for new metrics and addition of descriptive information (e.g., dates) to the tables and graphs. After the three rounds of focus groups, the researcher provided the updated requirements-driven information to the ISs individually and upon request. Thus, the researcher received continuous feedback through direct engagement with the internal stakeholders and observation of the stakeholders' use of the information. Once the requirements metrics were inserted into the requirements management process, the primary researcher and industrial partner decided to add the requirements 'analytics' element by proposing a 'traffic-light' system that would provide insight into the projects' health. Such a system would utilize the derived requirements metrics in conjunction with other project artifacts such as project schedules, budget, resources, etc. The researcher evaluated the intervention effects of the derived metrics on the requirements management and system development processes through informal discussions with the primary and secondary industrial participants and observations of the processes. Issues such as improved requirement-design coverage, improved planning of time and effort per release, etc. were noted.

As part of the reflection and learning phase of the AR study, the primary researcher took on the task of eliciting the challenges and lessons learned during the study, which resulted in the identification of the problems in Section 1 (i.e., lack of understanding of: the types of requirements-driven information, the ISs and their concerns with regard to the requirements-driven information, IS usage of the information, the project artifacts needed to derive the information). This, in turn, led to the research questions posed in Section 1. In an attempt to answer these questions, and given the availability of data from the AR study, a post-analysis was conducted to construct the meta-model, which we discuss in the following subsection.

3.2 Data Analysis: Meta-Model Building Procedure

To answer the research question posed in Section 1, we adopted the model construction process by Berenbach et al. [5] as we found it to be comprehensive. Berenbach states that a holistic understanding of the domain of interest is a prerequisite before commencing a model-construction process[5]. Our AR study allowed us to gain first-hand and in-depth knowledge of the overall context of the requirements engineering

and software development processes in the project under study. Moreover, our continuous collaboration with our industrial partner allowed for live feedback throughout the AR study and model-construction process, thus supporting incremental validation of the resultant meta-model. The key steps of the model construction process are:

(i) *Identify entities (**RQ1**):* The entities were incrementally identified and added to the meta-model by analyzing the data gathered from the AR study. First, the primary researcher extracted the metrics and the IS concerns they addressed from the metric spreadsheets and GQM+ document that was used to define the metrics during the AR study. The ISs were identified from the project's personnel documents and from meeting minutes that were gathered from the focus groups that were conducted during the AR. The project processes were extracted from the project's requirements management and change plans. We note that, up until this point of the entity identification process, the entities were concrete project data. We then began creating abstractions of the identified entities. For example, stakeholder categories in light of their requirements-related information needs (i.e., primary technical stakeholders, regular technical stakeholders, mid-level managers, high-level managers) were identified through analyzing the ISs' feedback and the primary researcher's correspondences with the ISs during the AR study. Specifically, the level of detail of the requirements-driven information requested by the ISs and the frequency with which they requested it were the main factors in determining these categories (see Fig. 1).

(ii) *Identify relationships among entities (**RQ2**):* We identified the relationships among the entities based on organizational rules such as the relationships between software artifacts and processes and their constituents. Other relationships were identified based on the metrics derived from the AR study such as the relationship between requirements metrics and their types. Finally, some relationships were identified based on our observations of the process and interactions between various elements in the project such as the relationship between ISs and their concerns.

(iii) *Synthesize the meta-model:* the identification of the entities and their relationships occurred iteratively and in parallel. Therefore, meta-model synthesis was an ongoing process since the beginning of the meta-model building procedure. For example, when we identified three main entities at the beginning of the process (i.e., requirements metrics, ISs, and IS concerns), we added the relationships between them and further entities and relationships were iteratively added as we gained better understanding of the entities and relationships involved. Moreover, the meta-model was incrementally updated in tandem with the feedback received from the reviews by the industrial partner, senior researcher and junior researcher, which resulted in the first version of the meta-model that did not include abstraction levels. After further evaluation and feedback at a workshop session [13] (see Section 5 for validation details), the abstraction levels were added, and the entities and relationships were updated accordingly.

We adopted Berenbach's [5] notation, for familiarity by the sponsor's organization, to depict the meta-model elements. An entity is represented by a rectangular box with the name of the entity. A relationship is represented by a line connecting two elements with a label to indicate the type of relationship between the elements.

4 A Meta-Model for Requirements-Driven Information for Internal Stakeholders

The meta-model is intended to complement the organization's requirements engineering process, specifically, the requirements management process. The company's requirements engineering process consists of: requirements elicitation, analysis, validation and management. The requirements management process includes a number of activities: tracing, managing requirements workflow states, managing requirements change, deriving and reporting requirements measures and other relevant requirements-driven information; the meta-model is intended to support this latter activity.

The current version of the meta-model for requirements-driven information for ISs consists of entities and relationships organized across three abstraction levels as proposed by [14]. In this section we will discuss the entities, relationships among the entities, and abstraction levels. Figure 2[1] depicts the synthesized meta-model.

Entities. The meta-model consists of five main entities that are pertinent to the process of deriving and providing requirements-driven information to ISs: *requirements-driven information* consists of information mainly derived from requirements and requirements meta-data and that may be supported with other artifact data; *ISs* who are involved in the system development and use the requirements-driven information; *concerns* that the ISs have with regard to the requirements-driven information and that are addressed by that information*; artifacts* from which the requirements driven information is derived; and *processes* in which the IS are involved in. These entities are represented at abstraction **Level 1**, the highest level of abstraction in the meta-model. Entities and relationships at level 1 are abstract and generalizable enough to be applicable to any context regardless of domain, software development process, or organizational structure.

The decomposed entities constitute abstraction **Level 2** of the meta-model. Entities at Level 2 are also intended to be generalizable to different contexts. However, its applicability may differ from one context to another. For example, while in a large systems project, such as ours, the distinctions between managerial and technical ISs are well defined, the differences may not be so evident in a smaller, more agile project.

Thus, it is up to the project stakeholders to decide which ISs fall into which entity type. Table 2 consists of the entity descriptions at abstraction level 2. Due to space limitations, we restrict our discussion to entities that are not deemed self-explanatory.

The entities at abstraction level 2 are further decomposed into entities at abstraction **Level 3**. Level 3 is the project specific level in which the entities are tailored to represent the environment of a given project in a specific domain and development process.

[1] High resolution images of Figures 2 and 3 can be found at: http://publish.uwo.ca/~inoorwal/Uploads/Meta-Model_publish.pdf

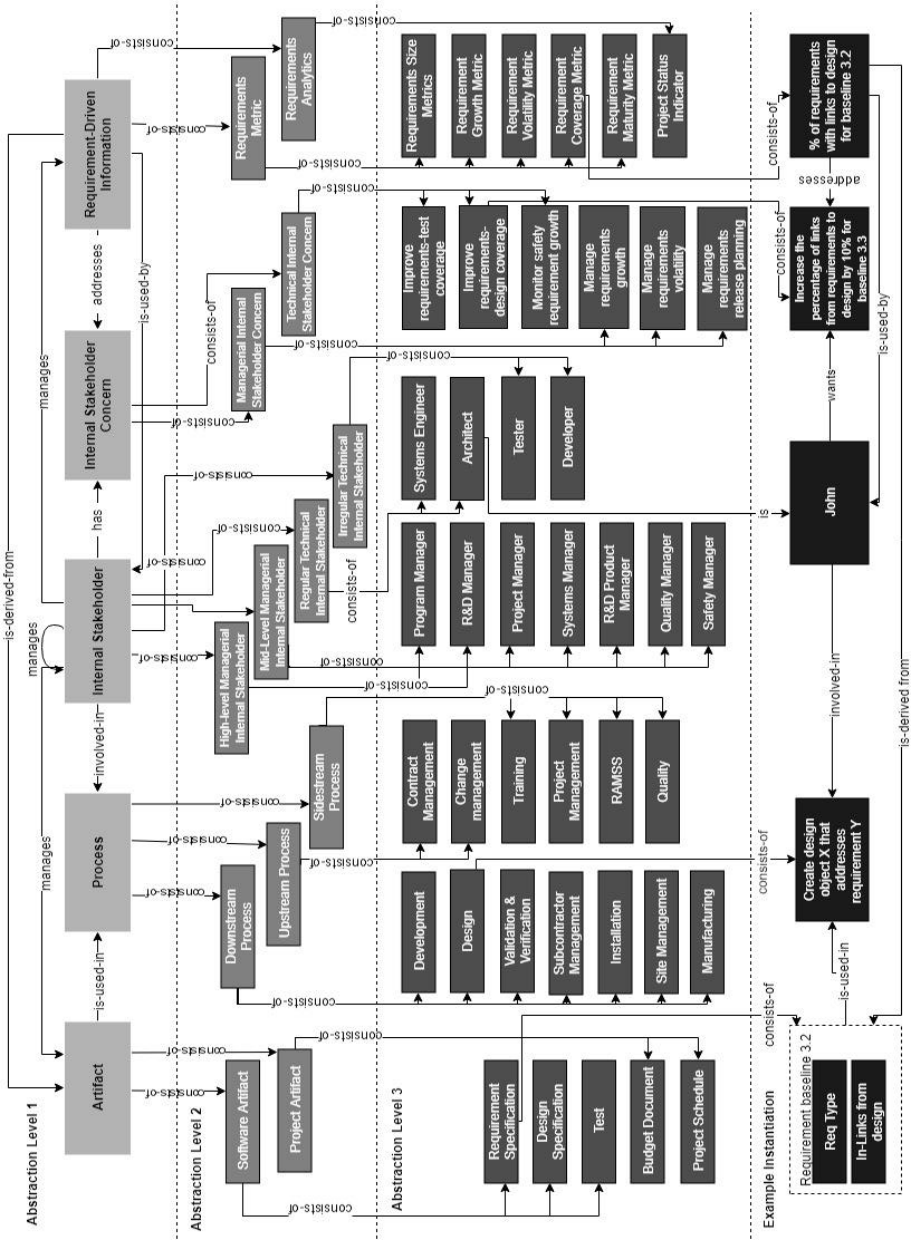

Fig. 2. Meta-Model for Requirements-Driven Information for Internal Stakeholders

For example, requirement metrics in our study consisted of size, growth, volatility, coverage, and maturity metrics. Another project's requirements metrics may include only volatility metrics. The same applies to other entities. Because entities at level 3 are specific to our project, we did not include a detailed description of them. However, they

can be seen in Figure 1 and they illustrate how the meta-model can be applied within a large systems project.

Table 2. Descriptions of meta-model entities at abstraction level 2

Entity	Description
Requirements Metric	A measurement derived from requirements to provide a quantitative assessment of certain requirements attributes
Requirement Analytics	Analytics on requirements data in conjunction with other software artifacts (e.g., design, code, budget and schedule documents, etc.) that aims to gain insight about the state of the project from a requirements perspective
High-level Managerial IS	Managerial stakeholders who manage at the project level or higher (i.e., program or regional levels) such as the program or regional R&D manager, etc.
Mid-Level Managerial IS	Managerial stakeholders who manage at the project level or lower (i.e., product level) such as test manager, product quality manager, etc.
Regular Technical IS	Technical ISs who use requirement-driven information regularly such as architects and requirements engineers
Irregular Technical IS	Technical internal stakeholders who use requirement-driven information less frequently such as developers and testers
Managerial IS Concern	Managerial issues that ISs care about in relation to the requirements such as estimating time and effort for a software release
Technical IS Concern	Technical issues that ISs care about in relation to the requirements such as increasing requirement-design coverage
Downstream Process	Activities involved in system development and initiated after the requirements engineering process such as development, design, testing, etc.
Upstream Process	Activities that are involved in system development and are initiated before the RE process such as contract/client management
Sidestream Process	Activities involved in system development and initiated and executed alongside the RE process such as quality and project management, etc.

Relationships. The following relationships are represented in the model: (1) is-used-by: represents the relationship when an inanimate entity (e.g., requirements metrics) is used by an animate entity (e.g., IS) to aid in technical or managerial tasks. (2) is-used-in: represents the relationship between entities when an inanimate entity (e.g., artifact) is used in another inanimate entity (e.g., process) to support the definition, execution, or management of the inanimate entity it is being used in. (3) addresses: represents the relationship between requirements-driven information and IS concerns. (4) consists-of: this relationship is used when an entity (e.g., requirements driven information) is composed of one or more of the related entities (e.g., requirements metrics and analytics). (5) is-derived-from: indicates that one entity (e.g., requirements size metrics) can be defined and specified from another entity (e.g., requirements specifications). (6) manages: indicates that an entity (e.g., ISs) can create, add, remove, modify the related entity (e.g., software artifacts). (7) involved-in: indicates that an entity (e.g., IS) actively participates in the related entity (e.g., processes). The participation can be in the form of execution, management, support etc. (8) has: indicates that an entity (e.g., ISs) possesses one or more of the related entities (e.g., IS concerns).

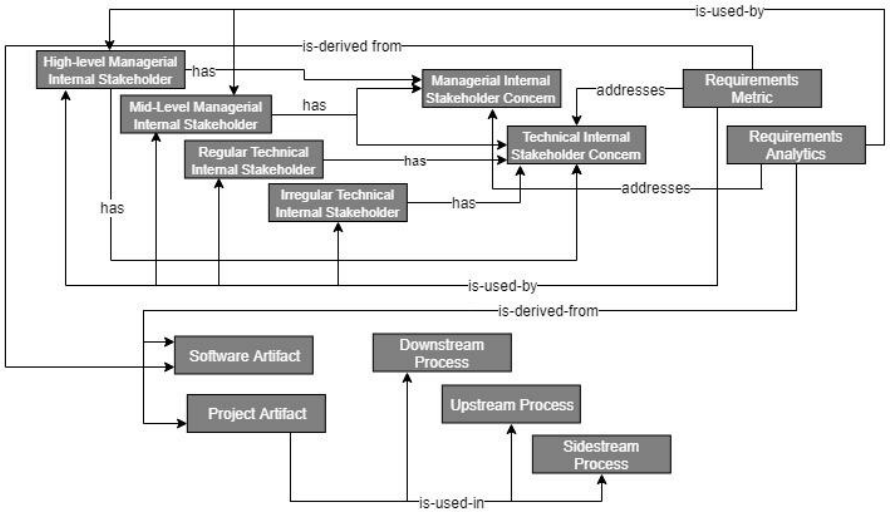

Fig. 3. A detailed overview of the relationships at abstraction level 2

The number of relationships among the entities increase as we go lower in abstraction level. This provides a more detailed picture of how the decomposed entities relate to one another [14] in different ways. For example, at **Level 1** there is one 'addresses' relationship between requirements-driven information and IS concerns. The 'addresses' relationships among the decomposed entities at **Level 2** increase in number and are more nuanced: requirements metrics 'addresses' managerial and technical IS concerns while requirement analytics 'addresses' managerial IS concerns only. Fig. 3 shows the expansion of **Level 2** relationships. Similarly, the number of relationships among the decomposed entities at Level 3 increase in comparison to the relationships among the entities at Level 2. The relationships at Level 3 are project specific and thus can be tailored to project and organization rules. Due to space limitations and to preserve the readability of the model, we did not include project-specific relationships at level 3. Finally, the relationships that cross over the abstraction boundaries are '*consist-of'* relationships that connect the higher-level entities with their lower-level constituents.

Rationale. We believe that several entity and relationship choices in the meta-model warrant a discussion of their rationale. The identification of categories of ISs and IS concerns is based on their needs regarding requirements-driven information and therefore a discussion is warranted. The meta-model separates managerial and technical internal stakeholders because they have different *concerns* regarding requirements measures and information, and, therefore, require different types of requirements-driven information. For example, an architect is concerned with tracking and improving requirements-architecture coverage and, thus, needs to know the number of requirements with and without links to architecture. On the other hand, a R&D product manager is concerned with estimating time and effort for a product release. Therefore, s/he needs the number of allocated requirements for a specific release. However, our experience with a large–scale systems project revealed that managerial ISs may also

have technical concerns. Then, how does the separation between managerial and technical ISs affect the generated requirements measures? We observed that even in the case when a technical and managerial IS share the same technical concern, the separation between managerial and technical ISs affected the level of detail of the relevant requirements-driven information. For example, both architect and a R&D manager may want to gain insight into the state of requirements-architecture coverage. However, while the architect is interested in detailed measures (e.g., the number of requirements that do not have links to architecture per feature, per release, and per requirements baseline), the R&D manager is interested in more big-picture measures (e.g., the overall percentage of requirements that have links to architecture per requirements baseline).

As for the separation between regular and irregular technical ISs, we observed that regular technical ISs need to be frequently updated with requirements-driven information while irregular technical ISs require the relevant information less frequently. For example, the architect requires a monthly report of requirement-architecture coverage measures and in detail. On the other hand, a tester requires requirement-test coverage measures only before a product release. Similarly, the separation between high-level and mid-level managerial ISs dictates both the frequency and level of detail of the relevant requirements information they need. These categorizations can aid the requirements engineer in knowing: *what* measures and information to generate and report from the requirements, to *whom* they should be reported, *how* to report it (i.e., level of detail), and *when* (i.e., how frequently), which, in turn, will facilitate the requirements management task of generating and reporting requirements relevant information.

Finally, the rationale for separating the meta-model into abstraction levels is to facilitate the tailoring of the meta-model to different contexts, and, thus, improving its generalizability.

Example Scenario. Fig. 2 depicts an instantiation of the model based on our project data. For example, the measure '% of requirements with links to design for requirements baseline 3.2' is derived from the project's requirements specification and uses the attributes 'REQ Type' and 'In-links from design' in the requirements database to calculate the measure. The requirements measure is used in 'creating design objects that address the system requirements' that 'John' (architect) is involved in and who wants to 'increase requirements-design coverage by 10% for baseline 3.3'. Knowing that 'John' is a regular technical IS, the measure will be reported to him in detail, which includes the percentage and absolute value of requirements-design coverage for baseline 3.2. and a list of the requirements that do not have links to design is also provided.

5 Meta-Model Validation

In [15], Shaw states that the form of validation in software engineering must be appropriate for the type of research result. For a qualitative model, validation through evaluation demonstrates that the study results (i.e., meta-model) describes the

phenomena of interest adequately [15] and validation through experience shows evidence of its usefulness. Thus, the objectives of our validation are to: (i) ensure that the meta-model adheres to the scientific principles of model building, (ii) identify missing, superfluous, and/or incorrect entities and relationships, (iii) ensure that constructs (i.e., entities and relationships) represent their correct real-world meaning, and iv) show preliminary evidence of its usefulness in practice.

Table 3. Meta-model Validation Phases

Validation Phase	Type of Validation	Involved Validators	Method	Output
Phase 1	Evaluation	V1, V2, V3	Expert opinion	Version 1 of the model (not included in paper)
Phase 2	Evaluation	V1, V4, V5, V6	Live study at workshop	Version 2 of the model (included in paper)
Phase 3	Evaluation, Experience	V7, V8	Expert opinion	Evidence of meta-model usefulness

Table 4. Profile of Meta-Model Validators

Validator	Research Experience	Industry Experience	Involved in Studied Project?
V1 Researcher	40 years	33 years of industry collaboration	No
V2 Practitioner	6 years	7 years	Yes
V3 Researcher	5 years	4 years	No
V4 Researcher	16 years	10 years of industry collaboration	No
V5 Researcher	44 years	30 years of industry collaboration	No
V6 Researcher	25 years	11 years	No
V7 Practitioner	2 years	17 years	Yes
V8 Practitioner	9 years	8 years	No

To this end, the meta-model went through three phases of validation (see Table 3) by eight validators (see Table 4). The validators' areas of expertise include empirical software engineering, requirements engineering, quality and architecture, testing, software ecosystems, global and cross-organizational software development, agile methods, agent-oriented analysis, modeling, simulation, and prototyping of complex sociotechnical systems.

Phase 1. **V1** reviewed the meta-model for the soundness of its entities and relationships. He also brought to our attention the notion of 'change' in the meta-model. That is, who makes the changes to requirements metrics, software artifacts and stakeholders? This is in line with Berenbach's approach in which he states that the following questions must be asked when building a meta-model [5]: Who creates the entities? Who modifies them? How do they become obsolete? This feedback from V1 resulted in the addition of the '*manages*' relationship between: ISs and metrics, ISs and software artifacts, ISs and ISs (see Section 4 and Fig. 2). **V2** is the main requirements management figure in the project that we conducted our AR study. He manages the RE processes for all the products in the rail-automation project. He, therefore, is the most

knowledgeable internal stakeholder on the RE processes. His validation consisted of feedback on the soundness of the meta-model constructs (i.e., entities and relationships) and ensured that the entities and relationships represented the project accurately. **V3** also reviewed the technical aspects of the meta-model to ensure the correctness of the meta-model. He also aided the first author in identifying proper relationship labels and reviewing the semantics of the meta-model.

Phase 2 consisted of a collaborative, live study at EmpiRE'18 [13] in which the meta-model from phase 1 was presented and explained to the audience. The participants were given questions to validate the meta-model, and then asked to write their answers on post-it notes that were pinned to their designated areas on the wall. 27 answers were provided in total and were used to enhance the meta-model. The main piece of feedback from the live study was the suggestion to divide the meta-model into abstraction levels.

Phase 3 is ongoing and consists of validating the meta-model for its usefulness in practice. To this end, we have sent out the meta-model to practitioners to gather their feedback on its usefulness. So far, we have received feedback from two practitioners (V7 and V8), who both asserted that the meta-model would be useful in practice with some modifications. V7, who has managed the project's quality management processes and is involved in the system architecture, says the meta-model would be very useful in managing the requirement-driven information that can be generated and disseminated among ISs. However, he suggests that "*this information get captured in modeling tools and thus tied to the system structure as opposed to chapters in a document*" for increased usability. V8 is from an external organization and states that "*I think the key are stakeholders. So taking the perspective of "WHO does/needs/provides WHAT?", this model would be a great way to elaborate what the stakeholder descriptions/roles are (for the internal stakeholders, and secondarily for the customer / upper management). In that respect, this model is a mental model that is used after having done stakeholder discovery (e.g., with the onion model) and gives some tools while documenting the stakeholder roles (e.g., when determining the importance & influence).*" Thus, phase three provides preliminary evidence for the anticipated practical benefits discussed in Section 1. V8 also suggested the replacement of the monochrome color scheme with different colors to facilitate reading and comprehension of the meta-model.

5.1 Threats to Validity

We discuss the study validity threats and how we mitigated them according to Runeson and Host's guidelines [16].

Internal Validity is concerned with the validity of causal relationships, typically in scientific experiments. Given that our study objective does not include investigation of causal relationships, this threat is not relevant to our study.

External Validity is concerned with the generalizability of the results to other contexts. The meta-model is based on the AR study conducted within the safety-critical, transportation domain, which may limit the meta-model's generalizability. Thus, readers must interpret and reuse the results in other contexts with caution. Despite this limitation, the results constitute an important data-point for making scientific progress.

Further validation of the meta-model in different domains and project sizes is encouraged in order to improve its generalizability.

Construct Validity concerns the operationalized constructs of the study in that whether or not they accurately represent the real-world phenomena. It is possible that some meta-model entities (e.g., stakeholder concerns, metrics, etc.) might not have been captured accurately by the researcher. In order to minimize this threat, we validated the model constructs with our industrial partner and analyzed them against official project documentation to ensure that the constructs accurately reflect their real-world counterparts. In addition, given that the meta-model was not the main goal of the AR study, there is a risk that important data is missing from the meta-model. This risk was mitigated by obtaining feedback from a variety of sources on the meta-model entities and relationships during the workshop (see Section 5 for workshop details).

Reliability: is concerned with the degree of repeatability of the study. The AR study followed AR principles for software engineering [17] to ensure rigor during the study. In addition, the AR and meta-model creation processes were documented to ensure traceability and analysis. Although a level of subjectivity is inevitable during the meta-model development process, our continuous involvement with our industrial partners and researchers inside and outside of the study helps to mitigate this threat.

6 Implications

Implications for Practice The meta-model can aid in aligning internal stakeholder concerns with requirements-driven information that can be generated within the project [18]. It can also be an effective tool for enabling effective communication as well as controlling project complexity [18]. In our case, the complexity is the network of numerous internal stakeholders, stakeholder concerns, requirement metrics and analytics, downstream, upstream and side-stream processes, and a web of interactions amongst them. Therefore, mapping out the numerous elements and the relationships amongst them will equip requirement engineers with the understanding needed to effectively control and manage the requirements-driven information they are required to provide [18] and communicate to the right people (see Phase 3 of validation in Section 5). The meta-model could also aid incoming personnel (e.g., new requirements engineers) in understanding this complex web of interactions, which, in turn, will help them in their requirements management tasks.

The meta-model can also serve as a stepping-stone toward operationalizing the entities and relationships in the meta-model in the form of a tool (e.g., dashboard) that could aid practitioners in the requirements management process by implementing features inspired by the meta-model (see Phase 3 of validation in Section 5).

Implications for Research The importance of requirements-driven information for internal stakeholders has been recognized by researchers [2, 8]. Some research efforts have targeted architects' and testers' information needs in relation to requirements and requirements specifications [2, 19] by proposing view-based solutions that would allow testers and architects to view the requirements specification in a format that will provide them with the requirements-based information they need. We take this work further by

attempting to explicate the types of stakeholders in light of their needs with regard to requirements-driven information. Further research can be conducted to explore further questions addressing IS information needs with regard to requirements. Such questions could include: what are the types of ISs in an agile environment? What are their information needs with regard to requirements in an agile environment? In addition to requirements metrics and analytics, what other types of information can be generated from requirements and that can benefit internal stakeholders in their processes?

7 Conclusions and Future Work

Requirements is an information-rich software artifact that has the potential to provide ISs with information that can guide their respective processes. However, little is known about the types of ISs in light of their requirements-information needs, the information that can be generated from requirements, and how this information is used by ISs, all of which complicates the requirements management process. Based on empirical data that we gathered and analyzed from an AR study conducted in a large-scale rail automation project, we identified the main entities and relationships involved in providing requirement-driven information, which we assembled into a meta-model. The empirically derived meta-model depicts the internal stakeholders, internal stakeholder concerns, requirements-driven information, artifacts, processes, and relationships among them at three abstraction levels.

Our preliminary validation shows that the meta-model aids in understanding the complex network of entities and relationships involved in providing requirements-driven information to internal stakeholders. More specifically, the explicit identification of the types of internal stakeholders and their needs in relation to requirement-driven information (see Section 3) could facilitate: (i) communication among internal stakeholders and (ii) proper identification and presentation of requirement-driven information for the correct internal stakeholders (see Section 4.1).

For future work, we intend to extend the meta-model to include cardinalities, which will provide a more accurate representation of a project's rules and policies. For example, only one IS (i.e., requirements engineer) manages the requirements-driven information. This cardinality is a representation of the current project practices. Therefore, upon reading the meta-model, one would know that one person is in charge of managing the various requirements-driven information and so appropriate interpretation is facilitated. We also plan to incorporate the meta-model into the organization's requirements management plan to validate it empirically for its practicality, usefulness, and benefits within the project.

Acknowledgements. We thank Philipp Hullmann and Eduard Groen for their valuable feedback. This work is supported by the Ministry of Education of Saudi Arabia.

References

1. Buse, R.P.L., Zimmermann, T.: Information Needs for Software Development Analytics. In: International Conference on Software Engineering. pp. 987–996. , Zurich, Switzerland (2012).
2. Gross, A., Doerr, J.: What do Software Architects Expect from Requirements Specifications? Results of Initial Explorative Studies. In: 1st IEEE International Workshop on the Twin Peaks of Requirements and Architecture. pp. 41–45. IEEE, Chicago, Illinois (2012).
3. Hess, A., DIebold, P., Seyff, N.: Towards requirements communication and documentation guidelines for agile teams. Proc. - 2017 IEEE 25th Int. Requir. Eng. Conf. Work. REW 2017. 415–418 (2017).
4. Doerr, J., Paech, B., Koehler, M.: Requirements engineering process improvement based on an information model. Proc. IEEE Int. Conf. Requir. Eng. 70–79 (2004).
5. Berenbach, B., Paulish, D.J., Kazmeier, J., Rudorfer, A.: Software and Systems Requirements Engineering in Practice. McGraw Hill (2009).
6. Wiegers, K.E.: More about Software Requirements: Thorny Issues and Practical Advice. Microsoft Press, Redmond, Washington (2006).
7. Sarkar, P.K., Cybulski, J.L.: Aligning System Requirements with Stakeholder Concerns: Use of Case Studies and Patterns to Capture Domain Expertise. In: Australian Workshop on Requirements Engineering. pp. 67–82 (2002).
8. Hassan, A.E., Hindle, A., Runeson, P., Shepperd, M., Devanbu, P., Kim, S.: Roundtable: What's next in software analytics. IEEE Softw. 30, 53–56 (2013).
9. Costello, R.J., Liu, D.-B.: Metrics for requirements engineering. J. Syst. Softw. 29, 39–63 (1995).
10. Berenbach, B., Borotto, G.: Metrics for model driven requirements development. In: Proceedings of the 28th International Conference on Software Engineering - ICSE '06. pp. 445–451. , Shanghai, China (2006).
11. Susman, G., Evered, R.D.: An Assessment of the Scientific Merits of Action Research. Adm. Sci. Q. 23, 582–603 (1978).
12. Basili, V., Heidrich, J., Lindvall, M., Munch, J., Regardie, M., Trendowicz, a.: GQM^+ Strategies -- Aligning Business Strategies with Software Measurement. First Int. Symp. Empir. Softw. Eng. Meas. (ESEM 2007). 488–490 (2007).
13. Noorwali, I., Madhavji, N.H.: A Domain Model for Requirements-Driven Insight for Internal Stakeholders A Proposal for an Exploratory Interactive Study. 2018 IEEE 7th Int. Work. Empir. Requir. Eng. 32–36 (2018).
14. Monperrus, M., Beugnard, A., Champeau, J.: A definition of "abstraction level" for metamodels. Proc. Int. Symp. Work. Eng. Comput. Based Syst. 315–320 (2009).
15. Shaw, M.: Writing good software engineering research papers. Int'l Conf. Softw. Eng. 6, 726–736 (2003).
16. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. 14, 131–164 (2009).
17. Santos, P.S.M. dos, Travassos, G.H.: Action Research Can Swing the Balance in Experimental Software Engineering. Adv. Comput. 83, 205–276 (2011).
18. Humphrey, W.S., Kellner, M.I.: Software Process Modeling: Principles of Entity Process Models. In: Proceedings of the 11th International Conference on Software Engineering. pp.

331–342. ACM (1989).
19. Hess, A., Doerr, J., Seyff, N.: How to make use of empirical knowledge about testers' information needs. In: IEEE 25th International Requirements Engineering Conference Workshops. pp. 327–330. IEEE Computer Society (2017).