# Assessing a Requirements Evolution Approach: Empirical Studies in the Air Traffic Management Domain

Fabio Massacci[a], Federica Paci[a], Le Minh Sang Tran[a], Alessandra Tedeschi[b]

[a] *University of Trento, Italy*
[b] *Deep Blue, Rome, Italy*

**Abstract**

Requirements evolution is still a challenging problem in engineering practices. In this paper, we report the results of the empirical evaluation of a novel approach for modeling and reasoning on evolving requirements. We evaluated the *effectiveness* of the approach in modeling requirements evolution by means of a series of empirical studies in the Air Traffic Management (ATM) domain. As we also wanted to assess whether the knowledge of the method and/or the application domain influences the effectiveness of the approach, the studies involved researchers, master students and domain experts with different level of knowledge of the approach and of the ATM domain. The participants have applied the approach to a real evolutionary scenario which focuses on the introduction of a new queue management tool, the Arrival MANager (AMAN) and a new network for information sharing (SWIM) connecting the main ATM actors. The results from the studies show that the modeling approach is effective in capturing evolution of complex systems. In addition, domain knowledge and method knowledge do not determine the effectiveness of the approach. Furthermore, the evaluation provided us useful insights on how to improve the modeling approach.

*Keywords:* requirements engineering, evolution, change management, user study, air traffic management domain

## 1. Introduction

The evolution of mission-critical requirements at enterprise level is known to be possible, but it is unknown whether it would happen: the *known unknowns* [1]. Unfortunately, large organizations cannot wait until the unknowns become known. The process of tendering and organizational restructuring requires a significant amount of time and planning. Decision makers at high-level must essentially bet on the final organizational solution and possibly minimize the risks that the solution turns out to be wrong. There is, thus, the need of approaches for evolving requirements that should

*Email addresses:* fabio.massacci@unitn.it (Fabio Massacci), paci@unitn.it (Federica Paci), leminhsang.tran@unitn.it (Le Minh Sang Tran), alessandra.tedeschi@dblue.it (Alessandra Tedeschi)

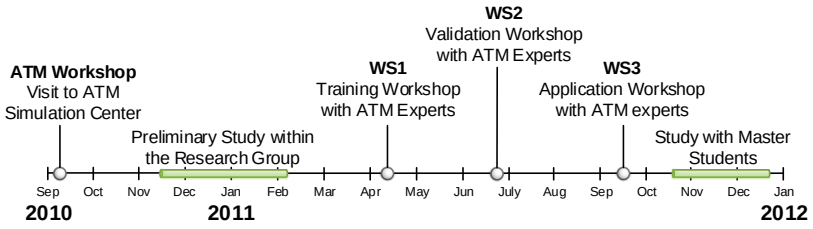



Fig. 1: Chronology of the family of empirical studies

help decision makers to select an optimal system design alternative that is resilient to requirements evolution.

In this paper we present the results of an empirical evaluation conducted on a requirements engineering approach to model and reason on requirements evolution (previously proposed in [1]). The evaluation aimed to assess the *effectiveness* of the approach in modeling requirements evolution and whether the effectiveness depends on the analyst's level of knowledge of the approach and of the application domain. To this end, we conducted three empirical studies with participants having different level of knowledge of the modeling approach and of the application domain. Fig. 1 summarizes how our empirical studies developed along a two-year horizon. First, we have conducted a study within the research team who have proposed the approach to model evolving requirements. Then, we have pushed the envelope further by carrying out a series of workshops with domain experts and industry practitioners as in [2]. Last, we conducted a study with MSc students.

As context for our evaluation, we have chosen the air traffic management (ATM) domain for three main reasons. First, ATM systems are complex and critical systems that are going through significant architectural, organizational, and operational changes as planned by the EU Single European Sky ATM Research (SESAR) Initiative [3]. Second, change management is a critical issue in the ATM domain. The need of system engineering techniques to support change management is well recognized [4]. Last but not least there is a significant body of research about empirical evaluations of requirements engineering approaches in the ATM domain [5, 6, 2]. For example, in [6], DMAN (Departure MANager), a system for managing departure of aircrafts, is used as context of evaluation. This makes it easier to benchmark our studies.

In our empirical evaluation, we have focused on changes associated with the introduction of a new decision supporting tool (the AMAN – Arrival MANager) and SWIM (System Wide Information Management) in the ATM domain.

The results from the studies show that the modeling approach is effective in capturing evolution of complex systems. In fact, the studies showed that it is reasonably possible for people different than the method's own inventor (such as students or domain experts) to build significantly large models, and identify possible ways for these models to evolve. Moreover, the studies have shown that domain knowledge and method knowledge do not have an effect on the effectiveness of the approach.

The paper is structured as follows. The next section introduces the context of our studies. Section 3 gives an overview of the approach to model requirements evolution being validated. We describe the research methodology in Section 4. Section 5 presents

the analysis of the data collected during the studies. Section 6 summarizes the main findings. Section 7 discusses the threats to validity. Section 8 gives an overview of related works while Section 9 concludes the paper with the lessons learned.

## 2. Application Scenarios

The context of our study is the evolution in air traffic management procedures planned by the SESAR (Single European Sky ATM Research) programme which is building the future European air traffic management system. The application scenarios (Table 1 provides a list of technical documents) were provided by Deep Blue Srl, an Italian consultancy company specialized in human factors, safety and validation of ATM concepts and systems, which actively participates to the SESAR Initiative. The scenarios focus on the introduction of a new queue management tool, AMAN, and the introduction of a new data transport infrastructure, SWIM, that will replace the current phone-communication lines.

Table 1: Technical documents of the scenario.

| Name | Document Title | Description |
|---|---|---|
| SC-D1.1[(1)] | Description of the scenarios and their requirements | describes in detail the requirements for the ATM scenario. Changes concerning to the introduction of AMAN are also elaborated. |
| SWIM-D1.2.1[(2)] | Information Content and Service Requirements | provides an overview of SWIM, ATM information content requirements and services requirements. |
| SWIM-D1.6.1[(2)] | SWIM Prototype Requirements for Iteration | describes the system context that the SWIM will face and support, including a set of usecases, scenarios where SWIM integrates with other systems. Requirements for the prototype iteration are also elaborated. |
| SWIM-D2.3.1[(2)] | SWIM-SUIT information models and services | describes existing ATM information systems, and future SESAR ATM system, as well as the role of SWIM network in the SESAR ATM architecture. Evolution of the SWIM services is also elaborated. |
| SWIM-TECH[(3)] | Segment 2 Technical Overview | describes in detail the functional architecture of SWIM, including architecture options, design solutions, and technologies. |

*Sources*:
[(1)] http://www.securechange.eu/content/deliverables
[(2)] http://www.swim-suit.aero/swimsuit/projdoc.php
[(3)] http://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techops/atc_comms_services/swim/documentation/media/Segment%202/SegmentTechnicalOverview_10709.pdf

Before the introduction of the AMAN, the flight arrival management operations are performed by the Sector Team composed by two controllers, the Tactical and Planning Controllers. This is done with the support of the CWP (Controller Working Position). The controllers have to compute the arrival sequence for the flights and give clearances for landing to the pilots flying in their sector on the basis of the information displayed by the CWP such as air traffic, radar data, weather condition, etc provided by different ATM actors. The communication among these actors takes places over a dedicated and secure communication line.

After the introduction of AMAN, the AMAN provides support to controllers by automatically generating the arrival sequence. The AMAN may also provide other functionalities, such as generation of advisories for aircrafts, or metering capabilities for a runway, or support runway allocation (at airports with multiple runway configurations). At the organizational level, the introduction of the AMAN requires the introduction of a new type of controller, namely, the Sequence Manager who will monitor and modify sequences generated by AMAN, and will provide information and updates to Sector Team. At the operational level, all ATM actors (including AMAN) communicate via SWIM, a new network for the management and sharing of information. This communication would provide authenticity, integrity and availability that should be comparable with the one provided by the dedicated communication lines (*e.g.,* phone) currently used by controllers.

## 3. The Validated Approach

This section gives an overview of an approach [1] to deal with requirements evolutions at design time. The ultimate objective of the approach is to help decision makers to select an optimal design solution so that the deployed system could be operational without (or with less) modification, while still keeping the development cost in budget. In this study, we aim to validate only the modeling aspect. Therefore, we do not discuss the reasoning part of the approach. Interested readers are referred to [1] for a detailed discussion.

Requirements evolutions refer to changes of requirements of a long-life software systems during their lifetime. Such changes could be either planned or unplanned. The unplanned changes are unexpected, but might happen due to external factors that are not under the control of the experts or the company who builds the system. These factors could be changes in the working environment, regulations, business processes, business agreements. Due to such changes, software systems might no longer be operational, and forces customers to modify the software systems (or develop new ones) to continue their business. To prevent major modifications to existing software systems that could suspend the business, requirements evolutions should be incorporated in the design phase of the software development process. Though unplanned changes are unexpected, they could be foreseen by experts with some levels of uncertainty about their occurrence. An example can be *"Which types of aircraft advisories will AMAN support?"*: some advisories might be very likely mandated, others might be unlikely supported but still on the table and others might be adopted by some states but not all. We call this uncertainty as *evolution probability*.

On the other hand, the planned changes are expected and they are usually incorporated into the system design (or at least the developers keep in mind such changes while developing). Still these changes might not be 100% certain because of some unexpected reasons and/or aforementioned factors. For instance we plan to implement a new function into SWIM next year, but then we might not do that due to some financial constraints. Consequently, planned changes are also associated with an evolution probability, thought this probability is usually high.

In the approach proposed in [1], both planned and unplanned changes are captured in terms of *observable rules* of the approach. An observable rule is a set of *evolution*

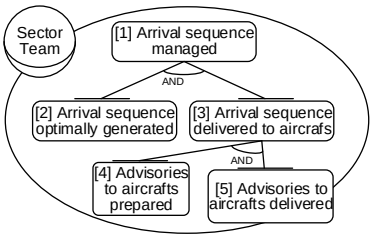



Fig. 2: An excerpt of the goal model for the Sector Team.

*possibilities* which are triplets of a *before*, an *after* model, and an evolution probability value. The *before* model is the requirements model when evolution has not happened yet. The *after* model is the one after evolution has taken place. The evolution probability is the expertise belief (*e.g.,* domain experts) that the *before* model evolves to the *after* model. An observable rule is formulated as follows:

$$r_o(\textit{Before}) \stackrel{\text{def}}{=} \left\{ \textit{Before} \xrightarrow{p_i} \textit{After}_i \left| \sum_{i=1}^{n} p_i = 1 \right. \right\} \tag{1}$$

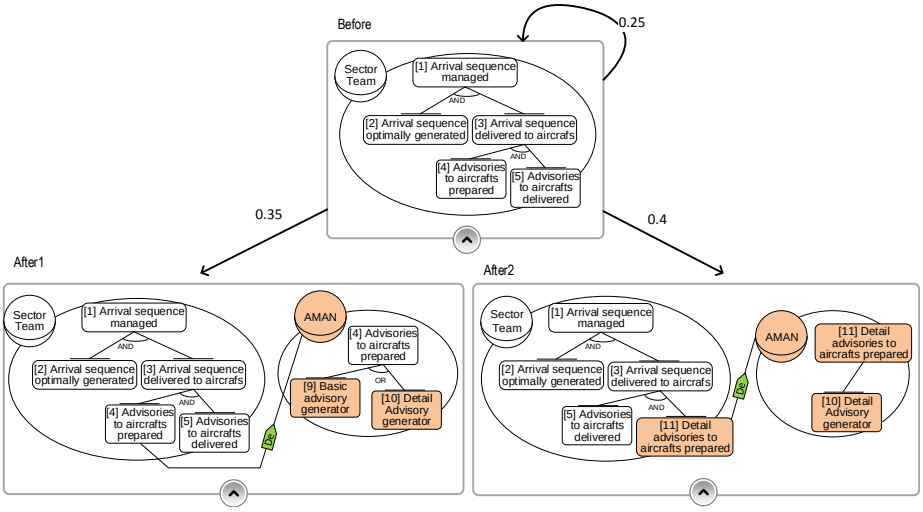
where $n$ is the number of possibilities and the *before* model might evolve. Not all the evolution possibilities will be implemented. An the end of the day, only one possibility (one *after* model) is materialized.

We capture different design alternatives (or implementation choice) of a requirement model in terms of *controllable rules*. A controllable rule $r_c$ is a set of tuples $\langle RM, RM_j \rangle$ that consists of a requirement model $RM$ and its possible design alternatives $RM_j$.

$$r_c(\textit{RM}) \stackrel{\text{def}}{=} \left\{ \textit{RM} \xrightarrow{p_i} \textit{RM}_j \,|j = 1..m \right\} \tag{2}$$

A controllable rule thus associates with a requirements model *RM* a set of design alternatives.

The evolution modeling approach discussed above is independent from any particular RE modeling language. In what follows we use the Si* language [7] to represent requirements models. Si* is founded on the concepts of *actor*, *goal*, *task*, *resource* and the relations *AND/OR decomposition*, *means-end*, and *delegation*. An *actor* is an active entity which models humans as well as software agents and organizations. A *goal* captures a strategic interest that actor wants to be achieved. A *task* represents a particular course of actions that produces a desired effect. It can be executed to satisfy a goal. A *resource* is an artifact produced/consumed by a goal or a task. *AND/OR decomposition* is used to refine a goal into sub-goals. The *AND-decomposition* means that the parent goal will be achieved if all its sub-goals are achieved or satisfied. The *OR-decomposition*, instead, means the parent goal will be achieved if at least one of its subgoals are achieved. The branches of a goal decomposition represent different design alternatives to fulfill the top goal. A *delegation* relation between two actors marks a formal passage of responsibility (*delegation execution*) or authority (*delegation permission*) from an actor (*delegator*) to the actor receiving the responsibility/authority (*delegatee*) to achieve a goal or to provide a resource.




The rectangles with label on top are the containers of *Before* and $After_i$ model. The label is the name of the contained model. Each container has a chevon at the bottom to collapse/expand its content. The arrows labeled with probability connecting two containers determines that the source model evolves to the target model.

Fig. 3: The graphical representation of the observable for goal $g_3$.

**Example 1 (*Before* Model)** Fig. 2 presents an excerpt of the goal model for the Sector Team before the introduction of AMAN. In the Sector Team's goal model we do not decompose the top goals into operational tasks to keep the model simple and easy to read. The top goal is $g_1$:*"Arrival sequence managed"* which is and-decomposed into $g_2$:*"Arrival sequence optimally generated"* and $g_3$:*"Arrival sequence delivered to aircrafts"*. The latter goal $g_3$ is further and-decomposed into $g_4$:*"Advisories to aircrafts prepared"* and $g_5$:*"Advisories to aircrafts delivered"*. ■

To represent an observable rule in a Si* model, we introduce a new graphical construct which is the container of the *before* and *after* models. When a model is large and complex, wrapping the entire model inside a container is sometimes hard to follow. Therefore, we recommend to analyze the evolution in sub parts of the models. However, if a big model is unavoidable, it requires the modeling tool to have a large-model support mechanism. Our prototype for modeling requirements evolution [8] addresses this problem by dividing a large model into several diagrams. A diagram then can link to others by using a special modeling construct.

Controllable rules are implicitly represented by *OR* decomposition, a native Si* graphical construct that allows designers to express alternative sub-goals to implement a parent goal. Therefore, in Si*, we represent controllable rules by means of *OR* decompositions.

**Example 2 (Evolution Rules)** We now illustrate how the Sector Team's goal model in Fig. 2 can evolve. The figure represents one observable rule and one controllable

rule. The observable rule

$$r_o(Before) = \left\{ Before \xrightarrow{0.4} After1, Before \xrightarrow{0.35} After2, Before \xrightarrow{0.25} Before \right\}$$

consists of three *evolution branches*: each branch corresponds to the arrow that links the before model *Before* to one of the after models *After1*, *After2* and *Before*. In the first evolution possibility, $g_4$ is delegated to AMAN. This dependency is presented by the line labeled with **De** connecting goal $g_4$ to the actor AMAN. The actor AMAN satisfies $g_4$ by either $g_9$:*"Basic advisory generator"*, or by $g_{10}$:*"Detail advisory generator"*. The probability that this possibility becomes true is $0.4$. In the second evolution possibility, *Before* might evolve to *After2* where a new goal $g_{11}$:*"Detail advisories to aircrafts prepared"* replaces $g_4$. The $g_{11}$ is also delegated to AMAN, and it is fulfilled by $g_{10}$:*"Detail advisory generator"*. The probability that this possibility occurs is 0.35. The third evolution possibility is that the model *Before* does not change with probability 0.25.

The controllable rule is represented by the or-decomposition of $g_4$ into goals $g_9$ and $g_10$ in *After1*. This rule has only two branches corresponding to the branches of the OR-decomposition. ■

## 4. Research Method

In this section we present our research questions and hypotheses, and the protocol followed to conduct our studies (Sec. 4.2).

### *4.1. Research Objectives*

Following the Goal-Question-Metric template [9], the goal of our studies is to assess if the method proposed in [1] is *effective* in capturing potential evolution of complex system requirements and whether effectiveness is influenced by knowledge of the domain or the method itself. Given the goal of our research, the main questions that we want to answer are:

$RQ_1$ *Is the approach effective in modeling requirements evolution of complex systems?*

$RQ_2$ *How does effectiveness of the approach is impacted by domain knowledge and method knowledge?*

We borrow the definition of effectiveness from the the Method Evaluation Model proposed by Moody [10] where the effectiveness of a method is defined as how well it achieves its objectives. Effectiveness can be measured by evaluating the quantity and/or quality of the results (output measures). Thus, to measure the effectiveness of the method we use the following variables that correspond to the main characteristics of evolution rules, the main outcome of the method's application:

- *size of baseline*. It is the number of unique model elements and interconnections in the before model of an observable rule.

- *size of change*. It is number of unique model elements and interconnections across all after models that are not in the before model or disappeared from the before model of an observable rule.
- *number of evolution rules.*
- *number of branches for evolution rules*.

**Example 3 (Counting Dependent Variables)** The dependent variables for the evolution rule described in Example 2 can be computed as follows:

- *size of baseline = 8* which includes 1 actor (Sector Team), 5 unique goals ($g_1$–$g_5$), 2 AND-decompositions ($g_1$ decomposes to $g_2, g_3$; and $g_3$ decomposes to $g_4, g_5$).
- *size of change =11* which includes 1 new actor (AMAN) + 1 deleted goal ($g_4$) + 3 new unique goals ($g_9, g_{10}, g_{11}$) + 2 De-dependency + 1 new OR-decomposition + 2 new AND-decompositions ($g_3$ decomposes to $g_5, g_{11}$; $g_{11}$ decompose to $g_{10}$) + 1 deleted AND-decomposition ($g_3$ decomposes to $g_4, g_5$).
- *number of evolution rules = 2* which includes 1 observable rule, and 1 controllable rule ($g_4$ OR-decomposes to $g_9$, $g_{10}$.)
- *number of branches for evolution rules*: the observable rule has 3 branches, and the controllable rule has 2 branches.

■

To investigate the second research question $RQ_2$, we use as control variables the method knowledge and the domain knowledge of subjects participating in our studies. We also defined the following set of null hypotheses $H_{n.m.0}$: $n$ denotes the research question to which the hypothesis is related, $m$ denotes the progressive hypothesis number, and $0$ denotes that it is a null hypothesis.

$H_{2.1.0}$ *There is no difference in the size of baseline identified by researchers, practitioners and master students.*

$H_{2.2.0}$ *There is no difference in the size of changes identified by researchers, practitioners and master students.*

$H_{2.3.0}$ *There is no difference in the number of evolution rules identified by researchers, practitioners and master students.*

$H_{2.4.0}$ *There is no difference in the number of branches for evolution rules identified by researchers, practitioners and master students.*

### 4.2. Experimental Design

The protocol consists of three main phases:

- *Training*.
  - Participants are administered a questionnaire to collect information about their level of expertise in requirement engineering, security and on other methods they may know.
  - Participants have to attend lectures about the modeling approach and on the ATM evolution scenarios depending on their expertise.
  - Participants are given a training material consisting of the slides used for introducing the modeling approach and documents describing the evolution scenarios and their requirements.
- *Application*.
  - Participants work alone or in groups and apply the modeling approach to the ATM evolution scenarios.
  - At the end of the application phase, participants have to deliver a report documenting the application of the method.
- *Evaluation*.
  - Participants are requested to evaluate the modeling approach through focus group interviews.
  - An ATM domain expert evaluates the report delivered by the participants to assess the quality of the models and the evolution rules drafted by them.

### 4.3. Experimental Procedure

We have conducted three studies with different kinds of participants. Following the terminology in [11], first, we run a preliminary study where the participants were the same researchers who have proposed the approach: the researchers have a good knowledge of the approach but are domain ignorant. Second, we have conducted a study with domain experts (a.k.a practitioners) who are novice to the approach, but are aware of the ATM domain. Third, we have conducted a study with master students who are method and domain ignorant (*i.e.* they have no prior knowledge of the approach and of the ATM domain). Table 2 summarizes the participants and their knowledge for each study. To conduct the studies, we have followed a mixed research approach that combines hypothesis testing with focus group interviews.

In what follows, for each study, we describe the participants, and the setting of the study.

#### 4.3.1. Study 1: Preliminary Study within the Research Group

*Participants.* Three researchers (the authors of this paper) have participated to the experiment. All of them had a background in requirement engineering and security, and were involved in the design of the approach to model and reason on requirements evolution.

Table 2: Participants' knowledge in the empirical studies.

| Study | Participants | Method Knowledge | Domain Knowledge |
|---|---|---|---|
| Study 1 | Researchers | Good | Limited |
| Study 2 | Domain Experts | Limited | Good |
| Study 3 | Master Students | Limited | Limited |

*Setting.* The researchers have first gained knowledge about the domain by attending half a day workshop about ATM procedures and tools, and safety and security issues in ATM organized by Deep Blue. Deep Blue also provided to the research team documentation about ATM process, AMAN and SWIM architecture and their functional and non functional requirements. After the training on the ATM domain, the researchers were engaged in three modeling sessions that took place at the University of Trento, each of the duration of half a day. During these sessions, the researchers worked independently and modeled several evolutionary scenarios following the approach to model requirements evolution. The scenarios considered include the introduction of the AMAN; the introduction of the ADS-B, a new surveillance tool used to determine aircrafts' positions, the introduction of the SWIM, and the introduction of the AMAN and SWIM to connect AMAN with queue management tools in other airports. For each of the evolutionary scenarios, the researchers have drawn an original model *Before* and identified an evolution possibility $After_i$. The *Before* model and the *After*$_i$ models have been modeled in the Si* language [7].

*4.3.2. Study 2: Workshops with ATM experts*

The study was organized into three separated workshops held in April 2011 (WS1), June 2011 (WS2), and September 2011 (WS3). The workshops involved both researchers and ATM experts. The role of researchers was to facilitate the workshop and make observations. The role of ATM experts was to apply the modeling approach and provide feedback about its effectiveness to model requirements changes.

- *Training workshop* (WS1) trained the participants on the modeling approach.
- *Validation workshop* (WS2) focused on the evaluation of the quality of the models and the evolution rules drawn by the researchers.
- *Application workshop* (WS3) asked ATM experts to apply the approach.

*4.3.3. WS1: The Training Workshop*

*Participants.* Seven ATM experts have participated to WS1: four of them are Deep Blue consultants with various background (*e.g.,* Computer Science, Human Factors, Safety and Security) who have worked in several projects related to the ATM domain. The other three ATM experts have been working for an European Air Navigation Service Provider with different roles and responsibilities: one is a system administrator, while the other two are air traffic controllers. The ATM experts have also extensive experience with the validation of new operational concepts [12] and are currently involved in various SESAR validations.

Fig. 4: Third ATM workshop.

*Setting.* The workshop started with a training session to introduce the experts to the requirements engineering domain and the modeling approach for evolving requirements. Then, ATM experts assessed the representation of changes (in terms of goals), the likelihood of particular change scenarios and the representation of such changes. Then, the research team held a focus group with the participants to identify possible evolution rules.

*4.3.4. WS2: The Validation Workshop*

*Participants.* Eight ATM experts participated to the second ATM workshop: seven participants were the same from the first workshop plus one additional participant who works as ATM manager.

*Setting.* During the workshop, the researchers have shown the original model and the possibility of evolution model $After_i$ they have drawn in Study 1. The quality of the requirements models and of the evolution rules has been discussed and the models have been revised with the domain experts. At the end of the workshop, the researchers conducted a semi-structured interview to collect preliminary feedbacks on the approach.

*4.3.5. WS3: The Application Workshop*

*Participants.* The third workshop had twelve participants: a security engineer from industry and eleven ATM experts. The ATM experts were the same as the other workshop plus two other Deep Blue consultants who have expertise in Security and Safety for ATM systems.

*Setting.* The workshop started with a brief presentation of the scenario to which the experts had to apply the modeling approach to requirements evolution and a summary of the steps they had to follow. The participants were divided into four heterogeneous groups (in terms of expertise). Each group had to draw an original model and one possibility of evolution model $After_i$ using the Si* tool. At the end of the workshop,

the research team engaged the participants into a focus group where the participants have provided additional feedback about the modeling approach. The application phase and the focus groups session have been audio-video recorded (see Fig. 4). Due to the limited time availability of the participants, the application phase did not terminate with the third workshop but continued remotely over a three months period going from September to November 2011.

#### *4.3.6. Study 3: Study with Master Students*

*Participants.* Eleven students enrolled in the Master in Computer Science at the University of Trento participated to the study. They had a background in Security Engineering and Information Systems.

*Setting.* Students were trained about the approach for evolving requirements during the Security Engineering course and they were introduced to the ATM case study. As additional material, they received three documents describing AMAN and SWIM users requirements, SWIM content and information services, and AMAN and SWIM core architecture. Then, the participants have been divided in four groups. Each group chose a possible scenario associated with the introduction of AMAN and SWIM network, and had to apply the approach for evolving requirements. After examining the scenarios, they drafted Si* models representing the requirements of the chosen scenario, and identified controllable and observable evolution rules. The participants were not observed during the application phase. Thus, to collect data about the application phase, students were asked to deliver a report describing in details the application of the approach and the generated models.

## 5. Quantitative Data Analysis

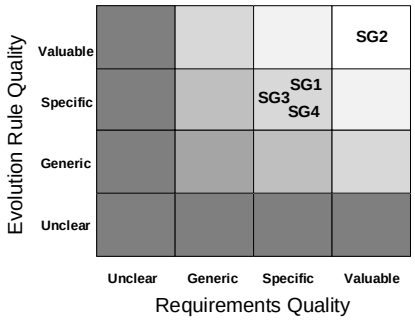
We collected the artifacts produced by researchers, domain experts and students as summarized in Table 3. The table reports for researchers, domain experts and students the mean and standard deviations of size of baseline, size of changes, and number of branches for controllable and observables rules.

To take into account the quality of the evolution rules and requirements models generated by students and researchers, we asked to a Deep Blue consultant who was expert in the ATM domain to assess the quality of the evolution rules and requirements models. The level of quality was evaluated on a four item scale: *Unclear* (1), *Generic* (2), *Specific* (3) and *Valuable* (4). Based on this scale, the groups who have got an assessment *Valuable* or *Specific* were classified as good groups because they have produced evolution rules and requirements models of good quality. On the contrary, the groups who were assessed *Generic* or *Unclear* were considered as not so good (bad) groups. Fig. 5 reports the final assessments in the matrix where the columns are the different quality levels of identified requirements models, and the rows are those of identified evolution rules. Among four groups of students, three have identified most of specific and important requirements and requirements evolution of the ATM scenarios. The last group was even better. They recognized valuable requirements and evolution rules from the scenarios. This implies that the artifacts produced by students

Table 3: Data about the Type of Participants and the Artifacts Generated.

Some standard deviation values for practitioners are not available because we have only one group of practitioners in our study.

| | **practitioner** | | **researcher** | | **student** | |
|---|---|---|---|---|---|---|
| **Effect** | mean | std.dev | mean | std.dev | mean | std.dev |
| Size of Baseline | 188.00 | 0.00 | 28.67 | 13.49 | 156.88 | 69.62 |
| Size of Change | 14.67 | 4.73 | 16.67 | 9.42 | 9.33 | 5.47 |
| Number of Observable Rules | 3.00 | – | 2.00 | 1.00 | 6.00 | 2.94 |
| Number. of Branches per Observable Rule | 2.00 | 0.00 | 2.17 | 0.41 | 3.42 | 0.83 |
| Number of Controllable Rules | 3.00 | – | 2.00 | 1.00 | 13.00 | 7.16 |
| Number of Branches per Controllable Rules | 2.00 | 0.00 | 2.00 | 0.00 | 2.27 | 0.49 |
| Total Number of Rules | 6.00 | – | 4.00 | 2.00 | 19.00 | 10.10 |




SG is the acronym for Student Group.

Fig. 5: The quality of requirements models and evolution rules produced by students.

are good enough for the purpose of this work. The quality of the models produced by the researchers was assessed by the ATM experts during the second workshop.

### *5.1. Preparation for an Analysis of Variance*

We wanted to determine the differences between the size of baseline, size of the change for evolution rules, the number of branches for evolution rule, and the number of evolution rules produced by researchers, practitioners, and students by means of the analysis of variance (ANOVA). In order to apply ANOVA, we first checked that its assumptions are satisfied. All the p-values in the following results are given under the assumption that the significance level $\alpha = 0.05$.

*Dependent Variables are Normally Distributed.* To check whether the dependent variables are normally distributed we used the Shapiro-Wilk test [13] for normality. For all dependent variables the p-value returned by Shapiro-Wilk test is lower than 0.05, and thus the variables are not normally distributed.

Table 4: Kruskal Wallis Summary

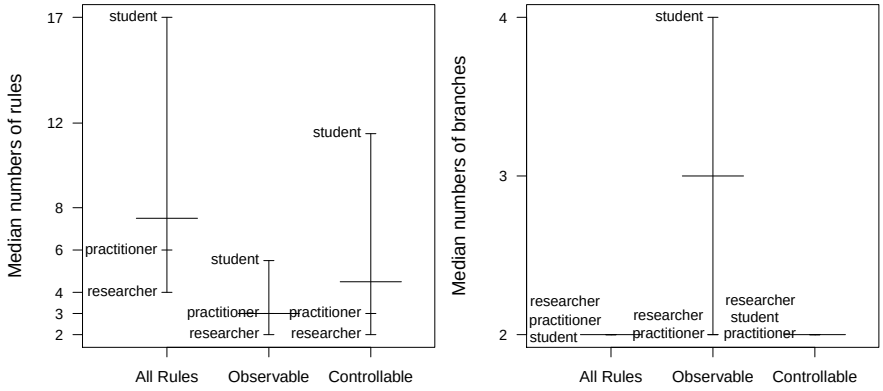
| Effect | Degree of Freedom | Kruskal Wallis $\chi^2$ | p-value |
|---|---|---|---|
| Size of Baseline | 2 | 15.842 | **0.000** |
| Size of Change | 2 | 6.342 | **0.042** |
| Number of Observable Rules | 2 | 4.652 | 0.098 |
| Number of Branches per Observable Rule | 2 | 12.786 | **0.002** |
| Number of Controllable Rules | 2 | 5.622 | 0.060 |
| Number of Branches per Controllable Rule | 2 | 2.801 | 0.246 |
| Total Number of Rules | 2 | 5.622 | 0.060 |

*Homogeneity of Variances.* We test the homogeneity of the variances with Flinger-Killen test. The test results are not significant with $p \geq 0.05$, except for the total number of branches. The assumption on homogeneity of variances is thus met for all the dependent variables except for the total number of branches.

*Observations Independence.* By design, the observations about the different types of participants are totally independent of each other.

### *5.2. Results*

Since the assumptions on normal distribution are not satisfied, we cannot use ANOVA. We need to apply Kruskal-Wallis test, which is the non-parametric alternative to ANOVA when ANOVA assumptions are not met.




(a) Number of Rules for Participants Type (b) Number of Branches for Participants Type

The short horizonal lines show the median of the total number of evolution rules (a), and median of the total number of branches per rules (b) for type of participants and type of evolution rule.

Fig. 6: Number of Rules and Branches for Participants Type.

First, we compared the difference in the number of evolution rules and number of branches for evolution rules across the different type of participants. Fig. 6(a) shows the median of the number of evolution rules in total and for type of rules (observable and controllable) produced by the researchers, practitioners and students, while

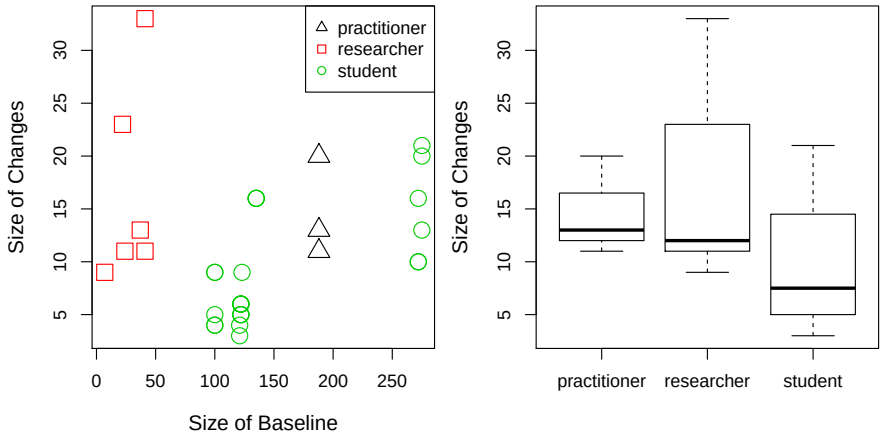
Table 5: Wilcoxon Rank-Sum Test Summary -Pairwise Comparison among Type of Participants.

Since we perform three comparisons per each effect, the Bonferroni-corrected significant level $\alpha$ is $^{0.05}/_{3} = 0.017$.

| **Effect** | **Pair of Type of Participants** | | **p-value** |
|---|---|---|---|
| Size of Baseline | practitioner | researcher | 0.025 |
| | practitioner | student | 0.171 |
| | researcher | student | **0.000** |
| Size of Change | practitioner | researcher | 1.000 |
| | practitioner | student | 0.111 |
| | researcher | student | 0.035 |
| Number of Observable Rules | practitioner | researcher | 0.637 |
| | practitioner | student | 0.468 |
| | researcher | student | 0.074 |
| Number of Branches per Observable Rule | practitioner | researcher | 0.637 |
| | practitioner | student | **0.016** |
| | researcher | student | **0.003** |
| Number of Controllable Rules | practitioner | researcher | 0.637 |
| | practitioner | student | 0.400 |
| | researcher | student | 0.057 |
| Number of Branches per Controllable Rule | practitioner | researcher | **0.000** |
| | practitioner | student | 0.340 |
| | researcher | student | 0.175 |
| Total Number of Rules | practitioner | researcher | 0.637 |
| | practitioner | student | 0.400 |
| | researcher | student | 0.057 |
| Total Number of Branches | practitioner | researcher | 0.556 |
| | practitioner | student | 0.051 |
| | researcher | student | 0.023 |

Fig. 6(b) reports the median of the number of branches. Students produced significantly more evolution rules than researchers and practitioners, who produced around the same number of rules. The same holds if we consider the number of controllable rules, but not the number of observable rules. In fact, researchers, practitioners and students have identified around the same number of observable rules. We checked with Kruskal-Wallis test if these results are statistically significant. The difference in the total number of rules (p-value = 0.06), the total number of observable rules (p-value = 0.098) and of controllable rules (p-value = 0.06) identified by researchers, practitioners and students is not statistically significant. These results are confirmed by the pairwise comparison that we have run with Wilcoxon rank-sum test as shown in Table 5. With respect to the total number of branches per rules and the total number of branches per controllable rules, there is no difference between students, practitioners and researchers. However, students perform better than researchers and practitioners with respect to the number of branches per observable rules. They produced a lower number of observable rules but with more branches than the controllable rules. The results of the Kruskal-Wallis test show that the difference in the number of branches per observable rules (p-value = 0.013) is statistically significant. This does not hold for the number of branches per controllable rules (p-value = 0.246). The pairwise comparison with Wilcoxon rank-sum test shows that the difference in the total number of branches

is not statistically significant for researcher and students (p-value = 0.023), while the total number of branches for observable rules is significant for practitioners and students (p-value = 0.016) and researchers and students (p-value = 0.003). The difference in the number of branches for controllable rules is statistically significant only for the pair practitioners-researchers (p-value = 0.000).




(a) Size of Change vs. Size of Baseline (b) Size of Change for Participants Type

Fig. 7: Size of Baseline and Size of Changes for Type of Participants

We then compared the size of baseline and size of changes identified by researchers, practitioners and students. Fig. 7(a) shows that researchers have sketched requirement models of lower size but have considered changes of small, medium and big size. Similarly, practitioners have produced a single big requirement model and changes of similar complexity of researchers. Students have produced two different kind of artefacts: some group of students produced small models and small changes; other groups identified one big model and changes of increasing complexity like practitioners. However, Fig. 7(b) shows that there is no difference in the size of changes identified by researchers, practitioners, and students. The results of Kruskal-Wallis test reported in Table 4 show that the results are statistically significant both for the size of baseline (p-value = 0.000) and the size of changes (p-value =0.042). However, a pairwise comparison conducted using Wilcoxon rank-sum test shows (see Table 5) that the difference between the size of baseline is statistically significant only for the pair researcher-student (p-value = 0.000.). Instead, the difference between the size of changes is not statistically significant for any pair of participant type.

## 6. Discussion

This section summarizes the main findings from the studies we conducted (see Table 6).

Table 6: Hypothesis testing results

| No | Hypothesis | Support |
|---|---|---|
| $H_{2.1.0}$ | No difference in the size of baseline sketched by researchers, practitioners and master students. | NO |
| $H_{2.2.0}$ | No difference in the size of changes identified by researchers, practitioners and master students | MAYBE |
| $H_{2.3.0}$ | No difference in the number of evolution rules identified by researchers, practitioners and master students. | YES |
| $H_{2.4.0}$ | No difference in the number of branches for evolution rules identified by researchers, practitioners and master students | NO (for the number of branches for observable rules) |

### *6.1. Method's Effectiveness*

The analysis of the experimental data revealed that our method is effective in modeling requirements evolution since researchers, practitioners and students were able to produce requirements models of medium size and identify several evolution possibilities for these models.

### *6.2. Impact of Domain and Method Knowledge*

*Impact on Size of Baseline.* The results of Kruskal-Wallis test and of the Wilcoxon rank-sum test show that the size of initial requirement models produced by students is higher than the one of researchers. Thus, null hypothesis $H_{2.1.0}$ can be rejected. We can also conclude that domain knowledge and method knowledge do not have an effect on size of baseline variable for two main reasons: a) students who are method ignorant have produced bigger models than researchers who are method aware; b) students have produced initial requirements models with size similar to the one of the models produced by practitioners who are domain aware.

*Impact on Size of Changes.* The Kruskal-Wallis test shows that there is a statistically significant difference in the size of changes identified by researchers, practitioners and students. However, this result is not supported by the Wilcoxon rank-sum test, which shows there is no statistically significant difference between any of the pairs of type of participants. Thus, we have not enough evidence to accept or reject hypothesis $H_{2.2.0}$. Based on the results of the Wilcoxon rank-sum test we may conclude that domain knowledge and method knowledge do not determine the size of changes.

*Impact on Number of Evolution Rules.* With respect to the total number of evolution rules, the Kruskal-Wallis test shows that there is no statistically significant difference among researchers, practitioners and students. Thus, $H_{2.3.0}$ can be accepted and we can conclude that domain knowledge and method knowledge do not have an effect on the number of evolution rules.

*Impact on Total Number of Branches per Rule.* The Kruskal-Wallis test shows that there is a statistically significant difference in the number of branches for observable evolution rules. In fact, students produced observable rules with significantly more branches than the one of researchers and practitioners. As result, $H_{2.4.0}$ is rejected.

Since we have evidence that the students performed better than researchers and practitioners with respect to the total number of branches per rule, we can conclude that domain knowledge and method knowledge do not help to identify an higher number of evolution scenarios.

### *6.3. Implications for the method*

During the focus group interviews, the ATM experts reported important aspects of the approach that require further investigation. They all pointed out that it is not possible to predict all the possible changes in advance especially for complex systems such as ATM systems:

> "*Sometimes, when you apply you discover a third change that is better than the one you have predicted*", ATM Manager
> "*The model may be good but when you switch from theory to practice you realize that there are many situations that you did not consider*", ATM Manager.
> "*We are talking about very complex systems. You don't know from the beginning all the actors involved in the process. There are always certain changes that you cannot predict due to the complexity of the system.*", Senior Deep Blue consultant.

The ATM experts went further and suggested that an incremental approach should be applied to identify all the possible evolution alternatives:

> "*It should be an iterative process* ", and "*you need to have more iterations if you want to reach 100%. You cannot foreseen everything at the beginning*", ATM expert.

The ATM experts also suggested that the graphical representation should be simplified because it does not scale very well for complex systems such as ATM systems. They remarked that an incremental approach should be used to draw the rules.

## 7. Threats to Validity

We discuss the four main types of threats to validity [14] in what follows.

*Conclusion validity.* Conclusion validity is concerned with issues that affect the ability to draw the correct conclusion about the relations between the treatment and the outcome of the experiment. The main threat to conclusion validity relevant for our studies is *low statistical power*. The sample size must be big enough to come to correct conclusions. We performed a post-hoc power analysis for the Kruskal-Wallis test and Wilcoxon rank-sum test for the three users' cohorts. The size of our sample is too small to have a power of 0.80. Therefore, it will be necessary to run the experiment again with more subjects for each user's cohorts - researchers, practitioners and students.

*Internal validity.* Internal validity is concerned with issues that may indicate a causal relationship between the treatment and the outcome, although there is none. A threat to internal validity can be the use of different application scenarios across the three study groups. Different scenarios may generate a bias in this experiment as effects might be due to (or canceled by) the varying difficulties of the scenarios. To mitigate this risk we have asked practitioners to identify scenarios that were close in complexity according to their opinion. An important reason behind the use of different scenarios was to avoid

a problem that emerged in previous pilot studies: when a scenario is repeatedly used (and therefore progressively refined) domain experts tend to focus on the adequacy of the requirements models at object level rather than at meta-level. For example, by reconsidering the same scenario twice or assessing the work of the students in comparison with their own, practitioners might have spent several minutes discussing whether the right sentence for the goal in Fig. 2 was "detailed advisory", and evaluate a model as inadequate because "specific advisory" should have been used in its place. By using different scenarios the evaluation focused on the forest rather than the trees. We believe that the advantages far outweigh the risks.

Another threat to internal validity is related to the use of Si* as requirement modeling language. The feedback provided by the ATM experts on the possible adoption of the graphical representation to model requirements evolution in the ATM domain can be biased by the fact that the requirements model were drawn in the Si* requirements language. Si* graphical notation tends to get very complex even for simple models and this aspect may have influenced the feedback of the ATM experts. We should organize another study using a different requirements language to evaluate whether the feedbacks depend on the use of Si*.

*Construct validity.* Construct validity concerns generalizing the result of the experiment to the concept and theory behind the experiment. The main threat to construct validity in our experiment was represented by a communication gap between the research team and the domain experts. Research team and domain experts might use same terms with different meanings and this can lead to misunderstandings; therefore, wrong or unrelated feedback might be provided. For example, the distinction between *goal* and *resource* was difficult to understand for the experts. A resource in the requirements engineering domain is an artifact produced/consumed by a goal, which captures a strategic interest of a stakeholder that is intended to be fulfilled. In the ATM domain, a goal has the same meaning that resource has in the requirements engineering domain and this lead to confusion. To mitigate this threat we have included a "*mediator*" who occasionally reformulated questions of the research team for the domain experts and reformulated domain experts' feedback for the researchers. The mediator role in this experiment was played by a member of Deep Blue who was familiar with our approach.

*External validity.* External validity concerns the ability to generalize experiment results outside the experiment settings. External validity is thus affected by the objects and the subjects chosen to conduct the experiment. We reduced the threats to external validity by making the experimental environment as realistic as possible. In fact, as object of our experiment we have chosen a real evolutionary application scenario proposed by Deep Blue, a consulting company which is actively involved in SESAR Initiative.

However, a threat to external validity of our results is represented by the use of Si* as requirements modeling language. To generalize our results beyond our studies, we should run other controlled experiments where different requirements modeling languages - problem frames, natural language, tables - are used by subjects to draw the evolution rules.

Table 7: Overview of Research Methodology

| Validation Method | Description |
|---|---|
| Case Study | Monitor a phenomenon in its real context |
| Experiment | Investigate a testable hypothesis |
| Survey | Collect standardized information from a specific population |
| Ethnography | Study a community and community's members social interactions |
| Action Research | Influence or change some aspect of the focus of the research |
| Assertion | Use ad-hoc validation techniques |
| Lessons Learned | Examine qualitative data from complete projects |
| Benchmarking | Test performance running several tests |
| Screening | Feature-based evaluation done by a single individual |
| Effects Analysis | Use expert opinion to assess the quantitative effects of methods/tools |
| Project Monitoring | collect and store data during project development |
| Field Study | Monitor and collect data about different projects simultaneously |
| Literature Research | Analyze papers and other documents publicly available |
| Legacy Data | Examine data from completed projects trying to identify trends |

## 8. Related Work

In order to set the scope for the type of empirical studies we present in this paper, we first overview the existing empirical research methodologies, and we introduce a set of terms in the field of empirical research that we use throught the paper. Then, we discuss the works reporting empirical studies on requirements evolution and the one that use the ATM domain as evaluation context, and we categorize them based on the empirical research methodology they adopt.

### *8.1. Empirical Research Methodologies*

Different taxonomies [15, 16, 17, 18, 19] have been proposed to classify empirical research methodologies in software engineering. In Table 7 we summarize the major research methodologies from the taxonomies in [15, 16, 17, 18, 19].

Runeson et al. [15] identify four classes of empirical research methods: *case study*, which is an empirical inquiry that investigates a contemporary phenomenon within a real-life context; *survey* that is a collection of standardized information from a specific population by means of a questionnaire or interview; *experiment* is an investigation of a testable hypothesis when one or more independent variables are manipulated to measure their effect on one or more dependent variables; *action research* aims to solve a real-world problem while simultaneously studying the experience of solving the problem. Easterbrook et al. [16] also count *ethnographic studies* among the major research methodologies. Ethnographic research studies based on field observations a community of people to understand how the members of that community make sense of their social interactions. Kitchenham [17] also considers case study, experiment and survey as classes of empirical research methodologies, but she also identifies *screening*, *effects analysis*, and *benchmarking* as classes of research methods. Screening is a feature-based evaluation done by a single individual who not only determines the features to be assessed and their rating scale but also does the assessment. We have used this method to evaluate the correctness of requirements produced by students (in

Table 8: Related Work

| Work | Method | Our method |
|---|---|---|
| Kamsties et al. (1998) | Case Study | Mixed Method (screening of output models) |
| Villela et al. (2010) | Experiment | Mixed Method (procedure for researcher) |
| Maiden et al. (2004, 2005, 2007) | Case Study | Mixed Method (procedure for ATM experts) |
| McGee et al. (2011) | Case Study | Mixed Method (reference of the research method) |
| Herrmann et al. (2009) | Case Study | Mixed Method (delta requirements) |

Sec. 4.3.6). Effects analysis is a method that uses expert opinion to assess the quantitative effects of different methods and tools. Benchmarking is a process of running a number of standard tests/trials using a number of alternative tools/methods (usually tools) and assessing the relative performance of the tools in those tests. Zelkowitz et al. [18] enrich the taxonomies proposed in [15, 16, 17] with new classes of empirical research methodologies: *project monitoring*, *assertion*, and *field study*, *literature research*, *legacy data*, and *lessons learned*. Project monitoring focuses on the collection and storage of data that occurs during project development. An assertion is an experiment where the designer of a new technology is both experimenter and subject of study[1]. A field study monitors and collects data about different projects simultaneously. Literature research analyzes papers and other documents publicly available. Legacy data is a method that examines data from completed projects trying to identify trends. Lessons learned examine data from complete projects to identify qualitative aspects that can be used to improve further developments.

We have further based our approach on a broader set of guidelines proposed by Runeson et al. [15] about how to conduct qualitative research. Following those guidelines we have involved researchers, students, practitioners (*i.e.* domain experts) with different expertise and used semi-structured interviews, questionnaires and audio-video recordings to collect data in order to perform data triangulation. Furthermore, we also employ the *screening* and *effects analysis* to process and analyze the collected data.

### *8.2. Empirical Studies on Requirements Evolution*

Below we briefly review studies that are closely related to ours. From these studies, we adopt methods and settings in our study (see Table 8).

One of the first studies on the topic has been performed by Kamsties et al. [20]. They summarized the results of a case study on requirements interdependencies held with practitioners from ten small and medium enterprises. They showed that new requirements implementation can cause unpredictable interactions with requirements already implemented. In their case study, they employed the screening method [17] to perform the inspection of artifacts. We also apply the same method to evaluate the output artifact (*i.e.* i* models) during our studies.

With the aim of planning for changes beforehand, Villela et al. [21] proposed a method based on software evolution model to address adaptation needs and potential

[1] A subject of study is an agent that is studied and collected data on.

changes in all levels of software abstraction. The proposed model was later validated in another work [22] by the model proponents. In [22], Villela et al. presented a quasi-experiment in the field of Ambient Assisted Living to characterize the adequacy and feasibility of their proposed model. We apply a setting similar to theirs to conduct our study with researchers (see Sec. 4.3.1).

McGee et al. [23] followed the guideline in [15], which was also adopted in our study, to conduct a study on requirements change taxonomy. It focused on how change classification helps designers to understand the impact of change, why and when it happens. This study investigated changes recorded during development cycles of an industrial project. Change data were collected on spreadsheet, and validated using observer triangulation (customer, project manager, and analyst).

Another study on requirements evolution by Herrmann et al. [24] was one of the pioneer in specifying the delta requirements without having to describe complete system in detail. Herrmann et al. investigated the applicability of TORE, a requirements engineering approach to identify delta requirements for a engineering tool.The study measured improvements in the as-is-analysis, the to-be-analysis, and the prioritization of requirements. In our quantitative analysis (see Sec. 5), we employ the idea of delta requirements to study the complexity evolution rules where we focus on how big the change is in each evolution rule identified.

Maiden et al. [5, 6], have presented several case studies in the ATM domain to validate RESCURE, a scenario-driven requirements engineering process. In [6], they exploited the DMAN system of ATM to understand the scalability of research-based techniques (*e.g.,* i*, KAOS) employed by RESCURE against large socio-technical systems. These techniques are found scalable, tractable and useful with appropriate tool support, however a long-term commitment were required from researchers. In the sequel study [5], the authors ran several workshops with different roles of ATM experts to study the use of RESCURE to discover stakeholder and system requirements. The workshops were organized in three main phases: a *training* phase about RESCUE, a *brainstorming* phase, and then an *application* phase where the experts applied RESCUE to discover requirements for different ATM tools (*e.g.,* DMAN, CORA-2, and MSP). During the workshops, color-coded idea cards, post-it notes, A3 papers etc have been used to collect the results. We adopt the organization structure of these workshops to conduct our study with ATM experts (see Sec. 4.3.2).

In another study [2], Maiden et al. focused on the different techniques in RESCURE to generate requirements: brainstorming, scenario walkthrough, and usage of i* model. Their study revealed that: the brainstorming technique generated more requirements that were more general in nature and provided an overview of the system objectives; the scenario walkthrough generated more operational-specific requirements; the usage of i* models could generate requirements that were not covered by the above techniques. We employ all these techniques for the study with ATM experts, and the latter two for studies with researchers and students (see Sec. 4.3.1, Sec. 4.3.6).

## 9. Conclusions and Lessons Learnt

In this paper we reported the results of three studies that we have conducted in the ATM domain to evaluate the effectiveness of an approach to model requirements

evolution and the impact that domain knowledge and method knowledge have on effectiveness.

The main findings from the studies are that the modeling approach is effective in capturing evolution of complex systems. In fact, the studies showed that since researchers, practitioners and students were able to produce significantly big requirements models, and identify possible ways for these models to evolve. Moreover, domain knowledge and method knowledge do not have an effect on the effectiveness of the approach, since there is no statistically significant difference between the artifacts produced by students who are domain and method ignorant, and researchers who are method aware and practitioners who are domain aware.

The studies also highlighted a number of aspects that we should take into account to continue our research.

- *Subjects' Selection*. The selection of domain experts strongly influences the relevance of feedback collected and the satisfaction of the success criteria chosen for the case studies. In our case studies, the selected domain experts had a different background and so we were able to collect feedback about the approach to requirements evolution from different perspectives. However, an issue of the domain is the separation between ATM organizations and IT suppliers. They have different and often competing stakes. In future studies, we think that one should validate the approach separately with two groups of ATM organizations and IT supplier, and identify methods to *firewall feedback* by different groups. This might highlight competitive advantages that one group might gain over the other by adopting the method.

- *Language Gaps*. Another interesting lesson concerns the foreign language gap. The level of engagement of the domain experts depends on two main factors: the means to provide feedback, and the language in which such feedback needs to be provided. Our workshop sessions included Hungarians, Indians, Italians, Norwegians, and Vietnameses; juggling between languages made our meetings lively. Albeit obvious in hindsight, this was not mentioned in the previous work by N. Maiden and others [5, 6, 2] because their studies were clearly English-to-English. A possible solution is that the domain experts can discuss in their mother tongue language and then provide summary feedback in English, but this hampers the immediacy of the feedback, and "minority opinions" might not be reported (we noticed this phenomenon during the workshops). The mediator was a useful tool to mitigate the internal validity threats also in this setting.

- *Determinants of Users' Technology Acceptance*. A major factor in the level of engagement of domain experts is the *perceived compliance* with the practice in industry. In the ATM domain, the discussion was facilitated by the existence of a model representation (influence diagrams) which was very close to goal models. When we proposed the same approach to another company, a show-stopper in the discussion with a practitioner was simply "We use DOORS" (and therefore cannot use and should not waste time evaluating requirements models in format different than DOORS). This was purely a syntactical limitation, not a semantical or methodological one: we could have perfectly used DOORS to link

requirements expressed by goal models, but our tool simply did not do it, as we thought this was a matter of engineering. This is indeed true if we considered limiting our validation to an experiment (as noted in [25] this is what the vast majority of RE papers report). Being able to syntactically interface with these tools (even for just gathering requirements IDs to label goals), is essential to obtain better perceived compliance and thus a better engagement and case-study based validation. As future work, we would like to organize a controlled experiment to evaluate if perceived compliance is a motivating factor that leads to an individual's intention to use a methodology.

- *Evolution Probability Setting.* An aspect of the approach that deserves further investigation concerns the definition of a systematic process to obtain evolution probabilities. Typically, decision-makers are not able to provide probability that an event occurs but just the frequency with which the event happens [26, Chap. 10.2.1]. Even when they are able to provide probabilities, they are subjective and contain a high degree of personal bias. In fact, the probability that an event occurs differs from person to person. We will explore the use of game theory *e.g.,* Clarke-Tax mechanism to assess evolution probabilities based on the probabilities of an event specified by different decision makers.

**Acknowledgement**

This work is partly supported by EU-FP7-IST-NoE-NESSOS, EU-EIT-ICTLabs, EU-FP7-SEC-SECONOMICS, PAT-TRISE.

**References**

[1] L. M. S. Tran, F. Massacci, Dealing with known unknowns: Towards a game-theoretic foundation for software requirement evolution, in: Proceeding of the 23rd International Conference on Advanced Information Systems Engineering (CAiSE'11), Springer-Verlag, 2011, pp. 62–76.

[2] C. Ncube, J. Lockerbie, N. Maiden, Automatically generating requirements from i* models: Experiences with a complex airport operations system, in: Proceeding of the 13th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ'07), Vol. 4542 of LNCS, Springer Verlag, 2007, pp. 33–47.

[3] EUROCONTROL, ATM Strategy for the Years 2000+ (2003).

[4] R. Graham, N. Pilon, H. Koelman, P. Ravenhill, Performance framework and influence model in ATM, in: Proceeding of the 28th Digital Avionics Systems Conference (DASC'09). IEEE/AIAA, 2009, pp. 2.A.5–1 – 2.A.5–11.

[5] N. Maiden, S. Robertson, Integrating creativity into requirements processes: Experiences with an air traffic management system, in: Proceeding of the 13th IEEE International Requirements Engineering Conference (RE'05), IEEE Press, 2005, pp. 105–116.

[6] N. Maiden, S. Jones, S. Manning, J. Greenwood, L. Renou, Model-driven requirements engineering: Synchronising models in an air traffic management case study, in: Proceeding of the 16th International Conference on Advanced Information Systems Engineering (CAiSE'04), Vol. 3084 of LNCS, Springer Verlag, 2004, pp. 3–21.

[7] F. Massacci, J. Mylopoulos, N. Zannone, Security requirements engineering: The SI* modeling language and the secure tropos methodology, in: Z. Ras, L.-S. Tsay (Eds.), Advances in Intelligent Information Systems, Vol. 265 of Studies in Computational Intelligence, Springer Berlin / Heidelberg, 2010, pp. 147–174.

[8] L. M. S. Tran, F. Massacci, Unicorn: A tool for modeling and reasoning on the uncertainty of requirements evolution, in: CAiSE Forum, 2013, pp. 161–168.

[9] V. Basili, H. Rombach, The tame project: towards improvement-oriented software environments, IEEE Transactions on Software Engineering 14 (6) (1988) 758–773. doi:http://doi.ieeecomputersociety.org/10.1109/32.6156.

[10] D. L. Moody, The method evaluation model: a theoretical model for validating information systems design methods, in: ECIS, 2003, pp. 1327–1336.

[11] A. Niknafs, D. Berry, The impact of domain knowledge on the effectiveness of requirements idea generation during requirements elicitation, in: Proceeding of the 12th IEEE International Requirements Engineering Conference (RE'12), 2012, pp. 181–190.

[12] EUROCONTROL, European Operational Concept Validation Methodology, E-OCVM Version 3.0 (2010).

[13] N. M. Razali, Y. B. Wah, Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests, Journal of Statistical Modeling and Analytics 2 (1) (2011) 21–33.

[14] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, Experimentation in Software Engineering, Springer, 2012.

[15] P. Runeson, M. Host, Guidelines for conducting and reporting case study research in software engineering, Empirical Software Engineering 14 (2) (2009) 131–164.

[16] S. Easterbrook, J. Singer, M. Storey, D. Damian, Selecting Empirical Methods for Software Engineering Research, Springer, 2007.

[17] B. A. Kitchenham, Evaluating software engineering methods and tool part 1: The evaluation context and evaluation methods, ACM SIGSOFT Software Engineering Notes 21 (1) (1996) 11–14.

[18] M. Zelkowitz, D. Wallace, Experimental models for validating technology, Computer 31 (5) (1998) 23–31.

[19] V. R. Basili, R. W. Selby, D. H. Hutchens, Experimentation in software engineering, IEEE Transactions on Software Engineering 12 (7) (1986) 733–743.

[20] E. Kamsties, K. Hörnmann, M. Schlich, Requirements engineering in small and medium enterprises, Requirements Engineering 3 (2) (1998) 84–90.

[21] K. Villela, J. Dörr, A. Gross, Proactively managing the evolution of embedded system requirements, in: Proceeding of the 16th IEEE International Requirements Engineering Conference (RE'08), IEEE Computer Society, 2008, pp. 13–22.

[22] K. Villela, J. Dörr, I. John, Evaluation of a method for proactively managing the evolving scope of a software product line., in: Proceeding of the 16th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ'10), Vol. 6182 of LNCS, Springer, 2010, pp. 113–127.

[23] S. McGee, D. Greer, Software requirements change taxonomy: Evaluation by case study, in: Proceeding of the 19th IEEE International Requirements Engineering Conference (RE'11), 2011, pp. 25–34.

[24] A. Herrmann, A. Wallnöfer, B. Paech, Specifying changes only — a case study on delta requirements, in: Proceeding of the 15th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ'09), Springer-Verlag, pp. 45–58.

[25] N. Condori-Fernández, M. Daneva, K. Sikkel, R. Wieringa, O. Dieste, O. Pastor, Research findings on empirical evaluation of requirements specifications approaches, in: Proceeding of the 12th Workshop on Requirements Engineering (WER'09), Valparaiso University Press, 2009, pp. 121–128.

[26] M. S. Lund, B. Solhaug, K. Stølen, Model-Driven Risk Analysis: The CORAS Approach., Springer, 2011.