**Identifying Common and Persistent Errors Made By Novice Analysts When Modeling Business Processes: Utilizing a Hierarchical Error Classification**

**Abstract**

Accurate process modeling is critical to the successful design of information systems. Therefore, learning to design correct, complete, and irredundant process models is an important part of training for systems analysts, yet is very challenging, especially for novice analysts. In order to teach high-quality modeling skills, it is essential to identify the common difficulties encountered in designing process models. Motivated by this insight, we formulated two research objectives: (1) identify the errors made by novices during process modeling, and analyze and classify them in light of three quality criteria – completeness, irredundancy, and correctness; (2) identify the most common errors, particularly the most persistent ones, that is, those most resistant to training. To this end, we analyzed 525 models built by 181 students (two or three models per student) during an academic course. We employed (1) a qualitative content analysis based on the principles of the modeling language, and (2) a frequency analysis, wherein we counted the prevalence of each error type. Our analysis produced a four-layer hierarchical classification of errors with 52 elements, including 38 error categories, subcategories, and irreducible types. We also identified the most common and most persistent error categories, both of which pertained mainly to difficulties in abstracting from a given scenario. This hierarchical classification plays an important role in establishing ways to improve the quality of process models designed by systems analysts, especially novices. Moreover, identifying persistent errors and "cracking" them is an essential step in designing a learning methodology that will help novice analysts to recognize such errors and, indeed, avoid them in the first place.

**Keywords:** Business processes modeling, UML activity diagram, process modeling errors, modeling quality criteria, requirements engineering education

# Introduction

Business process management (BPM) is the branch of knowledge and practice that deals with the operations of an organization in pursuing its business goals. Process models serve as blueprints of these organizational processes and capture, in some graphical and/or textual notation, the tasks, events, states, procedures, and business rules that constitute these processes [‎25]. Done correctly, process modeling can improve operational efficiency, enable cost reductions, increase compliance, and improve the fit between organizational processes and information technology (IT) systems [‎21],[‎25],[‎35].

Mastering process modeling is a cornerstone of the effective design of complex information systems (IS). Hence, learning to build high-quality process models is an important part of training for systems analysts. However, novices often find process modeling (and modeling in general) to be especially challenging, particularly in complex, dynamic environments. This is because business modeling is essentially a cognitive design process that relies on an individual's past experience, detailed knowledge of the object under consideration, and understanding of a rich set of production rules [‎26]. In addition, novices tend to lack the perceptual expertise required to efficiently grasp a client's requirements, make sense of the associated model, draw appropriate inferences, and avoid redundancies.

To enable novices to acquire high-quality modeling skills, it is crucial to identify the difficulties they encounter when designing process models, by discovering their typical modeling errors. There is now an established stream of research dealing with the practice of process modeling and the overall quality of conceptual models [7]. However, while several studies in IS have addressed modeling errors, they tend to classify them in broad terms, and therefore fall short of helping either novices or experts understand what types of flaws or specific content areas need most attention [7]. Further, although there is consensus on the need for a shift from error detection to error prevention, a sufficiently detailed classification of error domains to enable such a shift is still outstanding.

Motivated by the above, we formulated two research goals: (1) to identify modeling errors made by novices during process modeling, and analyze and classify them in light of three quality criteria – *completeness*, *irredundancy*, and *correctness*; (2) to identify those errors which are most common, and also those that are most persistent, that is, errors that prove resistant to attempted correction despite training, practice, repeated reminders, and so on.

To identify modeling error types and track their persistence, we tested students' modeling abilities in two tasks over a semester-long course in Information Systems Analysis and Design. The first task was completed as part of a midterm test after students had studied modeling in theory but had yet to gain practical experience; the second task was administered as part of an end-of-semester final exam, after participants had undergone some practical, as well as theoretical, training. In both tasks, the students were presented with hypothetical scenarios and were required to model the organizational processes depicted in them using a UML (Unified Modeling Language) Activity Diagram (UAD) [‎30]. The scenarios were taken from the healthcare management field, because of its dynamism and relative complexity [‎16],[‎17].

The sample comprised 181 students, of whom 163 performed all three of the scenario-modeling assignments included in the two tasks, and 18 performed two scenario-modeling assignments, resulting in a total of **525** models. Subsequent analysis of these models yielded **5910** individual errors.

We employed the following research methods: (1) a qualitative content analysis [‎10] based on the principles of the modeling language, and (2) a frequency analysis, wherein we counted the prevalence of each error type and identified both the most common error categories and those whose frequency persisted despite training (i.e., the most persistent error categories). As noted, models were analyzed in light of three quality criteria: correctness, irredundancy, and completeness[‎13],[‎22]. *Completeness*is defined by the inclusion of *all* necessary constructs in a model. It is achieved when no requirement from a given set is absent. *Irredundancy*is defined by the inclusion in a model of *only* the necessary constructs. It is achieved when the given set of requirements is met with the minimum of constructs. *Correctness* is defined by the correct use of all constructs in the model. It is achieved when all constructs and relationships conform to the modeling rules, that is, when they accurately interpret and represent the given set of requirements.

The analysis yielded four outcomes. First, a **four-layer classification** with 52 elements, consisting of the three quality criteria, which served as the roots of the classification, 20 error categories and subcategories, and 29 irreducible error types (see Figure 3). The three criteria in which the classification is rooted are commonly used when examining models in terms of quality. However, the branches of the classification were oriented toward business-process modeling errors. Second, the **most common** errors occurred in three error categories: a semantic-correctness category accounted for 1528 of the errors; a "completeness of nodes" category covered 1354 errors, and a "redundancy of actions" category accounted for 1050 errors. The third outcome was that, overall, we found a significant improvement from the first task to the second in terms of errors associated with the completeness and correctness criteria. However, despite the improvement in correctness-related errors, some subcategories seemed to **persist over time**, while the large numbers of semantic errors implies capacity to further enhance the quality of models in this context. There was also improvement from the first task to the second in relation to irredundancy errors, but it was not statistically significant and the "redundancy of nodes" error category, in particular, appeared resistant to learning. Finally, both the common and the persistent error categories point to difficulties experienced by novices in **abstracting** from a given scenario – for example, distinguishing between necessary and unnecessary concepts or correctly mapping essential concepts into appropriate and required model constructs [‎25].

The remainder of this paper is organized as follows: Section 2 elaborates on the scientific background and specifically the modeling language chosen for the current work; Section 3 provides an overview of related works, while Section 4 presents our research objectives. Section 5 describes the research design and setting, and Section 6 presents the study results. Finally, in Section 7, we discuss our results, summarize our conclusions, and outline our plans for future work.

# Scientific Background: Business Process Modeling Notation

A business process model describes the business activities that must be carried out to achieve a business’s goals. Various notations can be used for business process modeling, examples of which include UML 2.0 activity diagrams [OMG, 2005], Business Process Modeling Notation (BPMN) [OMG, 2006], and event-driven process chains (EPCs) [Scheer, 1999].

The different notations all support the core aspects of business process modeling, including activities and roles, decision points, and flows. Notations differ in, among other things, their target users. For example, BPMN is oriented more toward business users, while UML activity diagrams are oriented toward software developers [].

In order to identify core modeling errors rather than those dependent on the nuances of the specific notation in use, we focused on the core elements of business process modeling. Namely, activities, flows, decision nodes, and roles, which feature in all of the notations. Notations such as BPMN support events and complex gateways, but these are not supported by many of the others []. Therefore, we chose the UML Activity Diagram (UAD) modeling language because it is known for its simplicity and broad variety of concepts, and is widely employed by system designers and academic institutions [‎12],[‎14].

In Table 1, we review basic modeling constructs supported by the UAD modeling language (The definitions are adapted from https://www.lucidchart.com/pages/uml-activity-diagram). Examples of process modeling using UAD notation can be seen in Figures 1 and 2.

Table 1. UML Activity Diagram constructs for process modeling

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Visualization** |
| **Start node** | Symbolizes the beginning of an activity. The start node is designated by a black circle. |  |
| End node | Symbolizes the final step in an activity. The end node is designated by an outlined black circle. A process can have one or more end nodes. |  |
| Control flows | Represent the process flow. Connectors (incoming and outgoing arrows) show the flow between steps of an activity. |  |
| Action | A step in the activity wherein the user or software performs a given task. |  |
| Role | Identify who is responsible for performing each action. The roles visualization is performed via a construct called swim lanes. Each lane is designed to one role. Each action is assigned to one swim lane. |  |
| Decision node (split node) | Represents a decision between two or more possible states or options. A decision node always has at least two paths branching out. Paths are accompanied by text describing the possible conditions or options. |  |
| Merge node | Represents a point where two or more flows merge. The symbol for a merge node is the same as the symbol for a decision node/ split node (a diamond). A merge node can be combined with a decision node if two or more paths all lead to the same decision point. In such cases the diamond symbol acts as a frame or container where different paths merge, and then branch again into different flows. |  |
| Fork node | Splits a single activity flow into two concurrent activities. Symbolized by multiple arrows emanating from a join. |  |
| Join node | Used to combine or synchronize two or more concurrent activities. A join node and fork node are both represented by a thick vertical or horizontal bar. A join node can be combined with a fork node if two or more concurrent activities all combine and then split into a different set of concurrent activities. |  |

# Related Works

In this section, we provide an overview of related works dealing with quality standards and error measurement in the realm of process modeling. First (section 3.1), we review quality standards suggested in previous works to evaluate conceptual models and present the quality criteria we chose to use in this study. Then, in section 3.2 we review studies that explore errors made during process modeling.

## Criteria for Quality Evaluation of Conceptual Models

Genero and his colleagues [‎13] reviewed over 266 papers which deal with quality aspects of UML models. They collected various quality factors, and then mapped them onto three main criteria suggested earlier by Lindland et al. [‎19]: Syntactic, Semantic, and Pragmatic. According to [‎19], the Syntactic quality criterion refers to how well the model adheres to the rules of the language; the Semantic criterion refers to how faithfully the modeled system is represented; and the Pragmatic criterion refers to how easy the model is to understand. However, while the classification of [‎13] and [‎19] is insightful, it presents Correctness as only a secondary factor, assigned to the Syntactic and Semantic criteria; and it leaves out other factors, such as Irredundancy, entirely.

Moody and Shanks [‎22] proposed five criteria to assess the quality of conceptual models: Completeness, Correctness, Simplicity, Understandability, and Implementability. In their classification, Completeness is defined as a state wherein the model contains all user requirements and is measured by calculating the number of requirements missing from the model. Correctness is defined as whether the model conforms to the rules of the modeling technique (i.e., whether it is a valid data model), and is evaluated in terms of the number of errors in use of the model constructs. Simplicity is achieved when all the client's requirements are met with minimum constructs; it is measured by counting the number of constructs in the model. Understandability is achieved if the concepts and structures in the data model are coherent. Finally, Implementability relates to the relative ease by which the model fits the organizational environment and serves its intended purposes. Moody and Shanks [‎22] relate to (Ir)redundancy as an additional quality factor. Irredundancy is achieved when each fact in the model is represented in a single place. They included the redundancy aspect under the rubric of Correctness, but they acknowledged that given its importance in modeling, it could have been defined as a quality factor in its own right.

Many researchers have focused on the Completeness, Correctness, and (Ir)redundancy criteria, on the grounds that these three factors together cover most aspects of quality in relation to conceptual models. Table 2 presents relevant studies for these criteria.

Table 2: Uses of Completeness, Correctness and Irredundancy in the literature

|  |  |  |
| --- | --- | --- |
| Criterion | Relevant papers | Example |
| *Completeness* | [‎1],[‎5],[‎9], [‎13],[‎23],[‎27],[‎32] | Anda and colleagues [‎1] used the Completeness criterion to measure whether any single use case diagram includes all relevant actors. |
| *Correctness* | [‎5],[‎9],[‎13],[‎23],[‎27],[‎31],[‎32] | Beimel and Kedmi-Shahar [‎5] used the Correctness criterion (among others) when examining UC diagrams before and after creating a navigation map. |
| *(Ir)redundancy* | [‎5],[‎9] | Dahan and colleagues [‎9] classified possible types of errors into (among others) redundant characteristics of a construct. |

Despite the importance and value of Moody and Shanks's [‎22] criteria, by themselves they are too broad to be of much use in improving either the teaching or practice of process modeling by identifying specific aspects of model quality that need improvement. Yet thus far, only a few studies have adequately refined these criteria into categories and subcategories. Sadowska [‎31], for instance, in his proposed evaluation metamodel for business processes from 2015, refined Correctness into Syntactic and Semanticqualities (as in [‎13] and [‎32]). Yet in our view this work remains insufficient. We argue that even further refinement is needed at the level of the individual conceptual model type. To achieve this, generic quality criteria, which are relevant to all conceptual models, should first be formulated. Then, these cross-model criteria should be uniquely interpreted for each conceptual model type and should be refined into categories and subcategories according to the model's rules and logic. This approach will help designers create more correct, complete, and accurate models, and also will help guide the teaching of modeling by enabling identification of common and persistent error types.

After reviewing the literature, and inspired by Moody and Shanks [‎22], we chose the three quality criteria delineated in Table 2 (Completeness, Irredundancy, and Correctness), because we believe they reflect the quality required in the UAD. We then refined these criteria into a **3-layer** hierarchical classification consisting of categories and subcategories of quality factors, all oriented to a particular modeling framework, the UML Activity Diagram (see Figure 3).

## Errors Made during Process Modeling

As noted above, a small number of works have begun to examine patterns of errors in process models made by systems analysts. In an empirical study, Roy and colleagues [‎28] explored error frequencies in industrial business process models, with samples from domains such as banking and capital markets. They focused on detecting only syntactic and control flow-related errors, since they examined given models without their context. Unlike the present study, they did not examine the work of novices, nor did they ask which types of errors are more persistent and resistant to learning.

Other studies involve modeling quality criteria (see section ‎3.1) while exploring errors made during process modeling. Figl et al. [‎11] carried out a questionnaire-based experiment among 154 IS students to test the influence of routing symbol design on process model comprehension in terms of accuracy, efficiency, and perceived difficulty.In this study, participants were not required to produce a process model but to answer questions related to business process models. Their findings suggest that "semantic transparency and aesthetic design of symbols lower the perceived difficulty of comprehension."

Recker and colleagues [‎25] examined business process modeling designs by novice analysts. However, they were interested in how different ways of visualizing the model (e.g., via textual, hybrid, or graphic forms of representation) affect the novices' performance and the quality of their models. They reported that "the quality of the process designs decreases with the increased use of graphics and that hybrid designs featuring appropriate text labels and abstract graphical forms appear well-suited to describe business processes."

Other researchers, such as [‎18] and [‎29], explored modeling errors by students, as in the present work. Katz and Shmallo [‎18] examined the effectiveness of an error-based approach when teaching database design, while Rozman et al. [‎29] explored errors made by students when modeling processes via BMPN. Both studies produced several error categories that were mapped onto common quality criteria, as described earlier.

As noted, (subsection 3.1), in our study, we refer to three quality criteria: Completeness, Correctness, and Redundancy (similar to [‎29]). Recall, Completeness criterion concerns whether the subject has identified a required concept and respectively added the correct construct to the model; Redundancy criterion concerns whether the subject added needless constructs to the model; and Correctness criterion concerns to the correct use the subject has made of all constructs in the model. We identified **11** main error categories mapped onto the three quality criteria. However, in the present study, the syntacticandsemantic categories (mapped onto the Correctness criterion) have undergone a thorough refinement that is unique to process modeling research. Each of these categories was fine-tuned into a 3-layer hierarchical classification of process-modeling error types. The total number of categories and subcategories assigned to the Correctness criterion is **40**. Moreover, after examining participants' individual-level performance, we identified persistent error types that were resistant to training. These findings will be discussed in detail in section 6.2.2.

# Research Objectives

Modeling business processes is a significant task when planning an information system. The task falls on the shoulders of analysts and is perceived as challenging. To help analysts (especially novices) produce high-quality models, we need to understand the difficulties they experience. Motivated by the need for this understanding, we formulated the following study objectives:

1. *Identify errors made by novices during process modeling, analyze them in light of the three chosen quality criteria (Completeness, Irredundancy, and Correctness), and classify them into categories and subcategories.*
2. *Identify the most common categories, particularly those that showed persistence (i.e., in which there was no significant improvement during learning).*

The outcomes of this study will establish ways to improve the quality of process models designed by systems analysts, especially novices. In addition, characterizing persistent errors can play an important role when it comes to designing a learning methodology that helps novice analysts identify and avoid them.

# Research Design and Setting

To address the research objectives, we conducted (1) a qualitative content analysis [‎10], derived from the principles and outline of the UAD modeling language, and (2) a frequency analysis, wherein we counted the prevalence of each error type and identified both the most common error categories and those whose frequency persisted despite training.

This design and setting were chosen because they allowed monitoring of participants' abilities and skills over an extended period of time as the course progressed. An advantage of the setting is that there was little attrition among participants, as students were required to complete the defined tasks to gain course credit. In the end, 163 students modeled all three of the study scenarios (described below), while 18 students modeled only the two scenarios in the first task.

## Participants and Procedure

Our study participants were 181 college students from two academic departments: the Department of Industrial Engineering and Management, and the Department of Business Administration (BA). The BA students are specializing in information systems. All the students were in their second year of the program. The study took place during a required course, *Information Systems Analysis and Design*, which was taught in three sections (class groups) by two different lecturers, who are among the study authors. Participants comprised all the students in two classes of 20 and 71 students, respectively, during the second semester of 2017, and in one class of 72 students during the first semester of that year.

For all students, the topic of business processes is covered at the beginning of the course as part of the IS analysis phase. In this stage of the course, students learn how to analyze and model (using UAD) an organization's existing state and its business processes. In particular, students are exposed to the constructs described in Table 1, and learn how to use them while modeling business processes. After that, the course concentrates on IS design using UML notation [‎30].

To ensure compliance with ethical norms, all tasks performed during the study were required tasks that were similar to those assigned as part of the course in previous years. In addition, participants were notified in advance (through the syllabus) what tasks they were expected to complete as part of the course requirements. The tasks were approved by the school's teaching committee, and three of the authors assessed participants' level of effort in completing the tasks to ensure it coincided with the expected effort for the course. Participants were not promised any bonuses for task submission. Finally, participants' work products were not identifiable to the authors at the time of grading, and two of the authors assessed the work products separately. All ethical aspects of the study were approved by the college research institute.

## Tasks

The study was based on two business process modeling tasks assigned at different stages of the course: one as part of a midterm exam, after participants had studied modeling in theory; and one at the end of the course, during the final exam (see Table 3). Each student completed two models as part of the first task. The models used in the analysis were those completed as part of the first and second tasks.

Table 3: Summary of the tasks

|  |  |  |
| --- | --- | --- |
| **Task no.** | **Task description** | **Task schedule** |
| **1** | Modeling two scenarios using AD | Mid-course |
| **2** | Modeling one scenario using AD | End of the course |

### **The First Task**

The first task was given to the participants as a midterm, worth 5% of their final grade. The midterm took place after participants had completed their theoretical study of business process analysis and modeling via UAD, and before they submitted the related home assignment. The task took the form of narrative descriptions of two scenarios in a hospital emergency room (ER). The first scenario referred to office admission for the ER (*office scenario*), and was described in one paragraph of eight lines. The second referred to nursing admission for the ER (*nursing scenario*), and was described in one paragraph of eighteen lines. The second scenario was slightly more difficult to model than the first, as it was longer, more detailed, and contained several distractions (i.e., information that appeared in the scenario but was not relevant to the process). Participants were given one hour to model the two scenarios using UAD.

### **The Second Task**

The second task took place at the end of the course as part of the students' final exam, after students had had time to learn and practice modeling and design principles. The task comprised one question on the exam, out of five questions in total, and was worth 15 points out of 100. As the exam was 70% of the final grade, the second task worth 10.5% of the final grade. As in the first task, participants were given a scenario and were required to model it via UAD. As noted above, 163 students participated in this task. Different versions of the task were used in semester A and semester B. In the former, the task scenario consisted of 23 lines over three paragraphs, and in the latter, it consisted of 26 lines in two paragraphs. Both scenarios were also drawn from the realm of healthcare and involved hospitalization of a patient (semester A) and providing approval to a patient for exceptional medical treatment (Semester B).

## Task Analysis and Scoring

The study tasks were analyzed and scored by the three lecturers, who as noted above included two of the current authors. To reach an agreed measure of participants' performance and ensure inter-rater reliability [‎2], we used the following procedure: Before analyzing participants' assignments, each of three lecturers first proposed a complete, correct, and irredundant solution for each task (i.e., a solution that fully satisfied the Correctness, Irredundancy, and Completeness criteria). The solutions were then discussed until agreement on acceptable solutions was reached, including mandatory and optional elements (see Figure 1 and Figure 2). Participants' submissions were analyzed against these expert solutions. (See the Introduction for definitions of the three chosen criteria and the general measurement methodology.)

In analyzing the students' models, we employed a qualitative content-analysis methodology, derived from the principles of the UAD modeling language. First, we compared each model to the appropriate expert solution and identified and highlighted error types relevant to the research objectives. Then, we classified the error types into categories and subcategories, which we later mapped onto the three chosen quality criteria. For each task and each criterion, some possible categories and subcategories were clear from the start. For example, by definition the Completeness criterion can be broken down into Completeness of actions, Completeness of flows, Completeness of nodes, etc. More specific effort types emerged through the work of identifying errors.

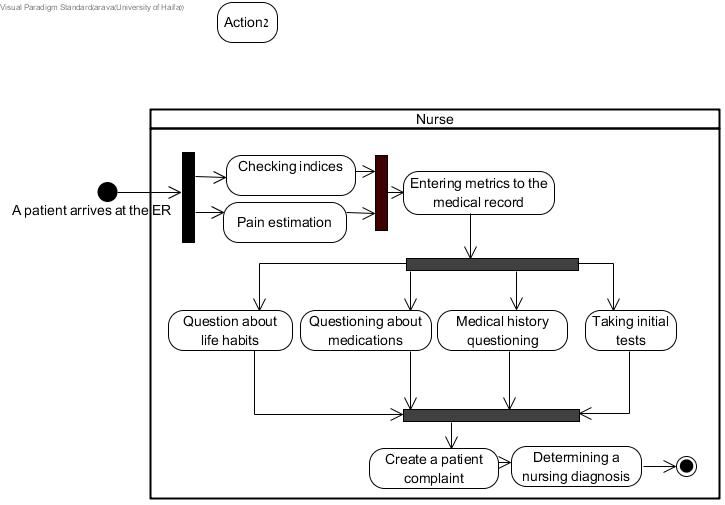
Finally, we scored each model. We awarded a "good" point for each construct that was presented appropriately, and a "bad" point for each inappropriate presentation of a construct (similar to Shoval and Shiran [‎33]). The constructs we analyzed and scored are the constructs listed in Table 1 (i.e., actions, roles, start/end points, flows, and all types of nodes). Recall, these are the constructs that the students learned how to use while modeling business processes via UAD.

For example, for the Completeness criterion, 1 "good" point was awarded for each required construct that was represented. Hence, the higher the score, the more required constructs were included, and the more complete the model. For the Irredundancy criterion, we added 1 "bad" point for each redundant construct represented. Here, a lower score implies fewer redundancies, and so the lower the score, the better (more irredundant) the model. Finally, for the Correctness criterion, we added one "bad" point for each error. Again, the lower the score, the more correct the model. After completing the scoring for each individual model, we averaged the scores from all participants within each category and criterion for each scenario. This allowed us to identify common and persistent errors.

In the following sections, we outline our recommended expert solutions for each task. The identification of error subcategories and the scoring will be discussed in more detail in section 6.

### Expectations for the **First and Second Tasks**

As described above, the first task involved two separate hospital ER scenarios (the office and nursing scenarios), while the second task involved patient hospitalization and approving exceptional treatment to a patient (see sections ‎5.2.1 and ‎5.2.2). We used the qualitative content-analysis methodology [‎10] and scoring system described above. These will be elaborated in section 6. Figure 1 and Figure 2 illustrate our expert solutions using UAD for the office and nursing scenarios (Task 1), while Table 4 summarizes the number of model constructs in the expert solutions for these scenarios. Figure 4 and Figure 5 (located in the Appendix) illustrate our expert solutions using UAD for the scenarios of semester A and B (Task 2), while Table 5 summarizes the number of model constructs in the expert solutions for these scenarios.

**Figure 1: Recommended solution for the office scenario of Task 1**

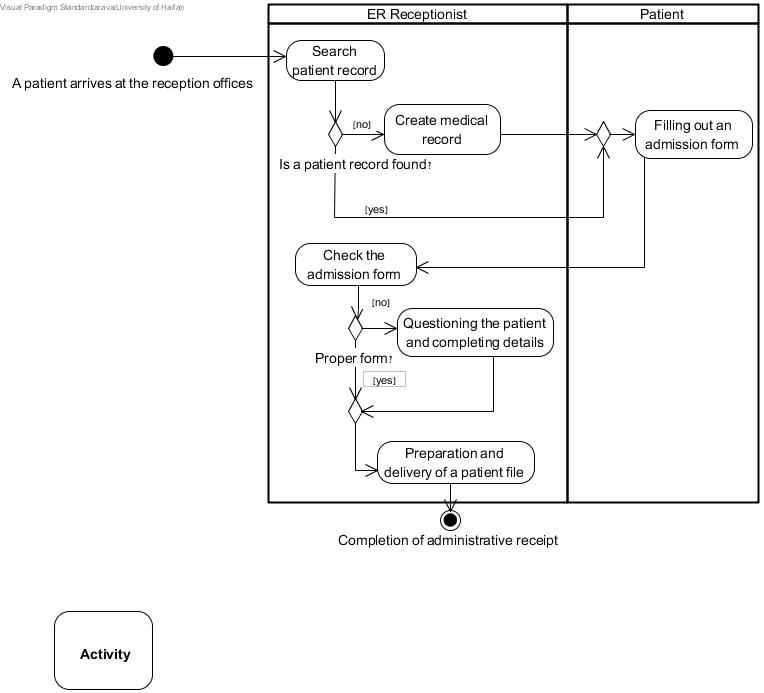


Figure 2: Recommended solution for the nursing scenario of Task 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Total no. of **constructs** | No. of **Start points/ Endpoints** | No. of  **roles** | No. of  **nodes** | No. of **flows** | No. of **actions** | **Scenario** |
| 27 | 2 | 2 | 4 | 13 | 6 | **Office** |
| 34 | 2 | 1 | 4 | 18 | 9 | **Nursing** |

Table 4: No. of model constructs appearing in the recommended solutions for Task 1

Table 5 presents the model constructs used in each expert solution for the two scenarios used in Task 2 (for students who completed the task in semester A and semester B, respectively).

Table 5: No. of model constructs appearing in the recommended solutions for Task 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Total no. of **constructs** | No. of  **Start points/ Endpoints** | No. of  **roles** | No. of  **nodes** | No. of **flows** | No. of **actions** | **Semester** |
| 58 | 3 | 3 | 8 | 30 | 14 | **A** |
| 59 | 4 | 2 | 10 | 30 | 13 | **B** |

# Results

In this section, we elaborate on the results of the study in terms of both classifications of errors (derived from our qualitative content-analysis methodology, which is based on the principles of the UAD modeling language) and students' performance, which enables identification of common and persistent errors.

We elaborate first on the results of the qualitative content analysis, in which we identified errors in the students' work and mapped them into categories and subcategories (section 6.1). This analysis produced a 4-layer hierarchical structure of error types (see Figure 3). We then look specifically at participants' individual-level performance, and particularly at the averages for the different error types for each category and criterion (section 6.2). This analysis enables us to identify the most common error types, and also to distinguish between error types that undergo improvement over time (i.e., errors that diminish in number as the students moved from theory to practice), and errors that persist over time.

## Error Classification and Scoring

The following subsections elaborate on each of the criteria, their respective categories and subcategories, and the scoring for each error type.

### Completeness

Within this criterion, we formulated the following **5** categories:

1. Completeness of actions: Whether a specific action was identified and specified.
2. Completeness of flows: Whether a specific flow was identified and specified.
3. Completeness of nodes: Whether a specific node of any type (split, merge, fork, join) was identified and specified.
4. Completeness of roles: Whether a specific role was identified and specified.
5. Completeness of start points/endpoints: Whether a specific point was identified and specified.

For every scenario and each of the above categories, the scoring was performed as follows: for each required construct, participants received one point for its identification and specification, otherwise 0. Then, we summed the number of misidentifications (which received 0 points) of all participants for all required constructs. This number was divided by the number of required constructs multiplied by the number of participants. This produced an error average score per participant for each category in every scenario. The lower the score, the more complete the model.

For instance, the office scenario includes six action constructs. Thus, each of our 181 participants was required to identify each of these six actions, resulting in a total of 6\*181 (=1,086) possible action identifications. With respect to the category "completeness of actions," we counted 150 misidentifications. Dividing 150 by 1,086 produces an average of 0.14 errors per participant for the "completeness of actions" category in the office scenario.

### Irredundancy

Within this criterion, we formulated the following **4** categories:

1. Redundancy of actions: The model contains any action not required by the scenario. For example: "Arriving via ambulance."
2. Redundancy of flows: The model contains any flow not required by the scenario.
3. Redundancy of nodes: The model contains a node of any type (split, merge, fork, join) not required by the scenario. For example: "Is there an escort?"
4. Redundancy of roles: The model contains any role not required by the scenario (for example, a "doctor" role in the office scenario).

Models were scored as follows: for every scenario and each of the above categories, 1 "bad" point was given for each redundantconstruct (i.e., a score of 0 is a "good" score). Then, for each category, we summed the "bad" points and divided this by the number of participants, to produce an average error score per participant for that category. Thus, the lower the score, the more irredundant the model.

For instance, in the office scenario, regarding the "redundancy of actions category," we identified 377 errors. Diving this number by 181 participants produced an average score of 2.08 redundant-action errors per participant.

Note that within this criterion, we did not analyze the construct of start point/endpoint, since in the vast majority of the models, we did not identify redundancy of this construct.

### Correctness

Most of the error types we found were mapped to this criterion. Our qualitative content analysis, derived from the principles of the UAD modeling language, led us to formulate two main categories: *semantic errors* and *syntactic errors*.

Semantic errors are errors resulting from incorrect interpretation of model constructs or their permissible combinations – for example, modeling a correct action under an incorrect role. We identified **16** semantic error types and mapped them into **5** subcategories (excess of generalization, lack of generalization, incorrect use of one construct instead of another, incorrect timing or positioning of constructs, and miscellaneous semantic errors). Part 1 of Table 6 presents all possible semantic error types along with their assigned subcategories.

Syntactic errors are errors resulting from the incorrect use of model constructs or their permissible combinations – for example, two flows emerging from the same action without use of a split node. We identified **13** syntax error types and mapped them into **4** subcategories (action errors, flow errors, node errors, and role errors). Part 2 of Table 6 presents all possible syntactic error types along with their assigned subcategories.

Table 6: Summary of Correctness error types mapped into categories and subcategories

|  |  |  |
| --- | --- | --- |
| **Error Type** | **Subcategory** | **Category** |
| Reducing two actions to one action | Excess of generalization | *Semantic errors* |
| Not generalizing an action name | Lack of generalization |
| Not generalizing a role name |
| Splitting an action into multiple actions |
| Modeling a precondition as an action | Incorrect use of one construct instead of another |
| Modeling an action as a flow |
| Modeling an action as a precondition |
| Modeling a decision node (split node) as a join node |
| Modeling a join node as a decision node |
| Modeling an endpoint as an action |
| Incorrect flow, action, or precondition | Incorrect timing or positioning of constructs |
| Correct action under an incorrect role |
| Representing actions as parallel instead of serial |
| Representing actions as serial instead of parallel |
| Incorrect order of actions |
| Any semantic error not classified as one of the above | Miscellaneous semantic errors |
| Action without incoming/outgoing flow | Action errors | *Syntactic errors* |
| Unnamed action |
| Two flows into one action | Flow errors |
| Two flows out of one action |
| A flow without an arrowhead |
| A decision (split) node with one outgoing flow | Node errors |
| A merge node with one incoming flow |
| A decision node with nameless outgoing flows |
| A join node connected to a decision node |
| A decision node connected to a join node |
| A fork node without a respective join node |
| Unnamed role | Role errors |
| Role without actions |

We next elaborate on the five semantic subcategories:

1. Excess of generalization: Any case where two separate required actions are represented by a single modeled action. Reflected in action names that relate to two required actions. For example: "Opening and updating a file" should be two actions, "Opening a file" and "Updating the file."
2. Lack of generalization: Any case where an action or role name is not generalized, or is split into multiple constituent components. An example of the former: using a role name from the scenario (e.g., "Rebecca") instead of the generalized name "Patient." An example of the latter: modeling the action "Creating a customer file" as the constituent actions "Printing a label," "Attaching the label to a file," and "Delivering the file to a patient."
3. Incorrect use of one construct instead of another: Any case where one construct (e.g., a precondition) is modeled instead of another (e.g., an action).
4. Incorrect timing or positioning of constructs: Any case where a flow, an action, or a precondition is incorrectly positioned in the model, whether in terms of their relationships or their sequence. Examples of the former: an outgoing flow from action one is modeled as an incoming flow for action three instead of action two; a required action is assigned to the wrong role. Examples of the latter: actions are modeled in the wrong order, or are modeled in parallel instead of serially or vice versa.
5. Miscellaneous semantic errors: Any error that cannot be mapped into one of the other four subcategories. For example: modeling an action twice, once as an action and once by sub-actions. Such miscellaneous errors are rare.

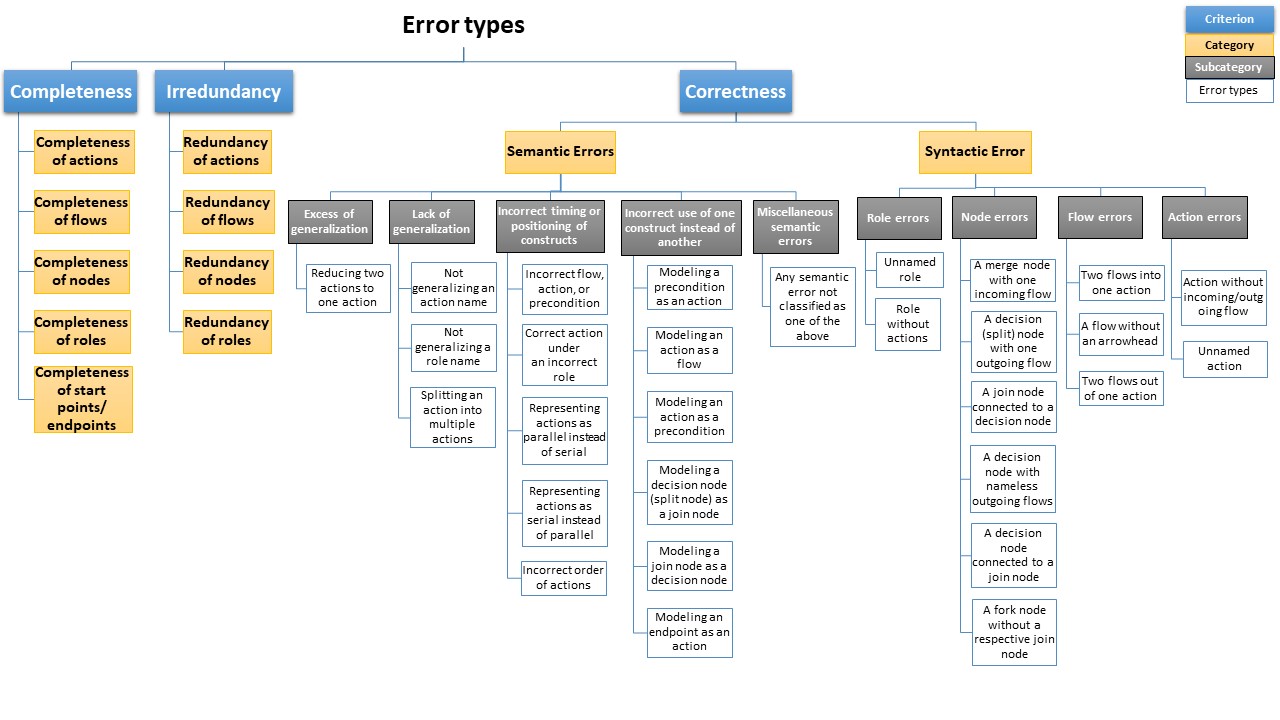
We do not elaborate on the syntactic errors as these are self-explanatory from Table 8, and because semantic errors are substantially more common than syntactic errors (this will become clear in section 6.2.3).

Similarly to the Irredundancy criterion, scoring was performed as follows: for every scenario and each of the above categories, 1 "bad" point was given for each error (i.e., a score of 0 is a "good" score). Then, for each category, we summed the "bad" points and divided the outcome by the number of participants, to produce an average error score per participant for that category. Thus, the lower the score, the more correct the model.

For instance, in the office scenario, regarding the "syntactic errors" category, we identified 40 errors. Dividing this number by 181 participants produced an average score of 0.22 syntactic errors per participant.

### Summary of Results for Error Classification and Scoring

Putting the above findings together, the analysis of the first and second tasks resulted in 38 error types within a **4-layer hierarchical structure** composed of **52** elements (criteria, categories, and subcategories). The full structure is visually depicted in Figure 3. A detailed table summarizing all the above results, including the error classifications and their respective scoring methods, can be found in the appendix (Table 11).

Figure 3: 4-layer hierarchy of error types

## Individual-level Performance

We now look at participants' performance for the tasks in terms of the three criteria.

### Completeness

Table 7 describes the average score and median number of errors in each of the Completeness categories for each of the four scenarios (recall that each participant modeled two scenarios in Task 1 and one of two scenarios in Task 2). For instance, looking at the category "completeness of actions" in the office scenario, each of our 181 participants was required to identify six actions. 79 of 181 participants (44%) succeed to identified all six required actions (= 0 errors), 69 (38%) erred to identify one out of six (= 69 errors), 27 (15%) erred to identify two out of six (=54 errors), 1 (1%) erred to identify three out of six (=3 errors), 3 (2%) erred to identify four out of six (=12 errors), and 2 participants (1%) failed to identify any of the six actions (12 errors). This resulted in total of 150 misidentifications, out of 6\*181 actions, producing an average of 0.14 action errors per participant.

Table 7: Participants' performance in the Completeness categories in Tasks 1 and 2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Categories  for Completeness** | **First Task** | | | | **Second Task** | | | |
| Office Scenario  (N=181) | | Nursing Scenario  (N=181) | | Semester A  (N= 72) | | Semester B  (N=91) | |
|  | Mean | Med. | Mean | Med. | Mean | Med. | Mean | Med. |
| Comp. of actions | 0.14 | 1 of 6 | 0.18 | 1 of 9 | 0.13 | 1 of 14 | 0.14 | 1 of 13 |
| Comp. of flows | 0.01 | 0 of 13 | 0.01 | 0 of 18 | 0.01 | 0 of 30 | 0.01 | 0 of 30 |
| Comp. of nodes | 0.65 | 3 of 4 | 0.56 | 2 of 4 | 0.32 | 3 of 8 | 0.32 | 3 of 10 |
| Comp. of roles | 0.11 | 0 of 2 | 0.07 | 0 of 1 | NA[[1]](#footnote-1) | NA | 0.10 | 0 of 2 |
| Comp. of start points/endpoints | 0.04 | 0 of 2 | 0.12 | 0 of 2 | 0.03 | 0 of 3 | 0.07 | 0 of 4 |
| Num of Constructs | 27 | | 34 | | 55 | | 59 | |
| **Completeness Error Average** | **0.14** | | **0.13** | | **0.09** | | **0.10** | |

In addition, for each scenario, we calculated a **Completeness error average** per participant in the following way: for each category, we multiplied the category error average by the number of its constructs included in that scenario. Then, we summed all the results and multiplied it by the total number of constructs belonging to the scenario. For instance, for the office scenario, which includes 27 constructs, the calculation was as follows: (0.14\*6 + 0.01\*13 + 0.65\*4 + 0.11\*2 + 0.04\*2)/27, producing a Completeness error average of 0.14 per participant for the office scenario.

Our participantsachieved consistent high levels of performance with respect to four categories under the Completeness criterion: actions, flows, roles, and start points/endpoints. They performed less well for Completeness of nodes in the first task. Yet an improvement in Completeness of nodes is seen from Task 1 to Task 2. We assume that identifying nodes requires an understanding of process flow, which involves more abstract thinking than is necessary to identify actions. Also, the Completeness average error scores for the first task's scenarios are similar (0.14 and 0.13) despite differences in the difficulty level between them (e.g., in the length of the scenario, its complexity, the number of model constructs required, etc.). Thus, we may assume that difficulty level did not affect participant achievement in the context of Completeness.

We further address the Completeness error average in section ‎6.2.4.

### Irredundancy

Table 8 presents the average score and median number of errors in each of the Irredundancy categories for each of the four scenarios. For instance, considering actions redundancy in the office scenario, 24 of 181 participants (13%) included no redundant actions, 46 (25%) included one redundant action (=46 errors), 42 (23%) included two (=84 errors), 41 (23%) included three (=123 errors), 18 (10%) included four (=72 errors), 8 (4%) included five (=40 errors), and 2 (1%) included six redundant actions (=12 errors). This sums to 377 errors for all 181 participants, producing an average of 2.08 redundant-action errors per participant. Recall, the lower the score, the better.

Table 8: Participants' performance in the Irredundancy categories in Tasks 1 and 2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Categories for Irredundancy** | **First Task** | | | | **Second Task** | | | |
| Office Scenario  (N=181) | | Nursing Scenario  (N=181) | | Semester A  (N=72) | | Semester B  (N=91) | |
| Mean | Med. | Mean | Med. | Mean | Med. | Mean | Med. |
| Redun. of actions | 2.08 | 2 | 1.81 | 1 | 1.51 | 1 | 2.59 | 2 |
| Redun. of flows | 0.01 | 0 | 0.03 | 0 | 0.0 | 0 | 0.01 | 0 |
| Redun. of nodes | 0.10 | 0 | 0.44 | 0 | 2.38 | 3 | 2.20 | 2 |
| Redun. of roles | 0.27 | 0 | 0.94 | 1 | NA | NA | 0.05 | 0 |
| **Irredundancy Error Average** | **2.46** | | **3.22** | | **3.89** | | **4.86** | |
| Num of Constructs | 27 | | 34 | | 55 | | 59 | |
| **Normalized  Error Average** | **0.09** | | **0.09** | | **0.07** | | **0.08** | |

In addition, for each scenario, we calculated an **Irredundancy error average** per participant as the sum of the category error averages for that scenario. For instance, for the office scenario, the calculation was 2.08 + 0.01 + 0.10 + 0.27, producing an Irredundancyerror average of 2.46 per participant. Then, in order to compare between the different scenarios, we **normalized** the Irredundancyerror average by dividing it by the total number of constructs belonging to the relevant scenario.

We can see from the table that the number of redundant actions is very high across all the scenarios. Thus, students' learning during the course did not reduce levels of this error. On the other hand, the number of redundant flows is low across all the scenarios. The number of redundant roles is higher in the nursing scenario, though this scenario required modeling only one role (compared to the office scenario, which required two roles). We assume that identifying only a single role is difficult as it requires an understanding of who is active in the process (i.e., performing actions) and who is passive (i.e., no need to be modeled). As students were given the roles to model in the second task, we can draw no conclusions about persistence of this error.

The most interesting insight regards redundant nodes. As noted above, the nursing scenario was designed to be more difficult than the office scenario; and the difficulty level in the second task was even higher. We assume that as the level of difficulty increases, participants tend to add more redundant nodes.

We address the normalized Irredundancyerror average in section ‎6.2.4.

### Correctness

This is the most challenging criterion and yielded most of the error types we found in this research. Table 9 presents the average score and median number of errors in selected Correctness categories, followed by total semantic and syntactic errors, for each of the four scenarios. Semantic errors are substantially more common than syntactic errors, suggesting that the rules underlying these categories are more difficult for novices to grasp. Hence, in Table 6 we delineate the semantic categories to a greater level of detail (the level of primary subcategories – the grey boxes in Figure 3).

Altogether we found 16 semantic error types (secondary subcategories) in five primary subcategories. As an example, the semantic error "splitting an action into multiple actions" (mapped to the "lack of generalization" category) yielded the following results in the office scenario: 67 of 181 participants (37%) did not incorrectly split any actions, 105 (58%) incorrectly split one action (=105 errors), 8 (4%) split two actions (=16 errors), and 1 participant (less than 1%) split three actions (3 errors). This sums to 124 errors for 181 participants, producing an average of 0.69 split errors per participant. Recall that in this case, again, the lower the score, the better.

Table 9: Participants' performance in the Correctness categories in Tasks 1 and 2

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Subcategories for Correctness** | **First Task** | | | | | **Second Task** | | | |
| Office scenario  (N=181) | | Nursing Scenario  (N=181) | | | Semester A  (N=72) | | Semester B  (N= 91) | |
| Mean | Med. | Mean | Med. | | Mean | Med. | Mean | Med. |
| Excess of generalization | 0.09 | 0 | 0.51 | | 0 | 0.61 | 0 | 0.20 | 0 |
| Lack of generalization | 1.87 | 2 | 1.38 | | 1 | 0.44 | 0 | 0.48 | 0 |
| Incorrect time/position | 0.74 | 1 | 1.19 | | 1 | 0.60 | 0 | 1.42 | 1 |
| Incorrect construct use | 0.22 | 0 | 0.43 | | 0 | 0.08 | 0 | 0.16 | 0 |
| Misc. semantic error | 0 | 0 | 0.13 | | 0 | 0.11 | 0 | 0.02 | 0 |
| **Semantic Error Aver.** | 2.92 | 3 | 3.64 | | 3 | 1.84 | 1 | 2.28 | 2 |
| **Syntactic Errors Aver.** | 0.22 | 0 | 0.41 | | 0 | 0.97 | 1 | 0.74 | 1 |
| **Correctness Error Aver.** | **3.14** |  | **4.04** | |  | **2.82** |  | **3.02** |  |
| Num of Constructs | 27 |  | 34 | |  | 55 |  | 59 |  |
| **Normalized Error Aver.** | **0.12** |  | **0.12** | |  | **0.05** |  | **0.05** |  |

In addition, for each scenario, we calculated a **Semantic error average** per participant as the sum of the semantic category error averages for that scenario. For instance, for the office scenario, the calculation was as follows: 0.09 + 1.87 + 0.74 + 0.22 + 0, producing a semantic error average of 2.92 per participant. Then, we calculated the C**orrectness error average** as sum of the semantic and syntactic error averages. For instance, for the office scenario, 2.92 + 0.22 produced a Correctness error average of 3.14 per participant. As with the Irredundancy criterion, in order to compare between the different scenarios, we **normalized** the Correctnesserror average by dividing it by the total number of constructs belonging to this scenario.

Overall, we found 13 different syntactic error types (secondary subcategories), which we mapped into four primary subcategories according to the four main structures: actions, flows, nodes, and roles. The most common error was entering two flows into one action, in the syntactic flow errors category. The number of syntactic errors increases with the complexity of the scenario, a finding which we attribute to the greater length of more-difficult scenarios (previous work has confirmed the intuitive observation that the longer the scenario, the greater the chance of making a syntax error [‎20],[‎25]).

With respect to persistence, the number of semantic errors in the first task was very high (a mean of 2.92 errors per participant for the office scenario and 3.64 for the nursing scenario). We assume that the latter figure is higher because the nursing scenario is more complex. However, the number of semantic errors decreased in the second task to an average of 1.84 and 2.28. We infer from this that as participants practiced and accrued experience over the semester, their semantic skills improved. The most significant improvements were in the categories "lack of generalization" and "incorrect construct use."

We address the normalized Correctness error average in the next section.

### Summary of Results for Individual-level Performance

Table 10 summarizes the number of errors found in each category within each scenario. The most common errors appeared in three categories: the semantic category, with 1528 errors; the Completeness of nodes category, with 1354 errors; and the redundancy of actions category, with 1050 errors. The total number of errors we found exceeded 5910.

Table 10: The number of errors found in each category within each scenario

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Criterion** | **Category** | **First Task** | | **Second Task** | | **Total no. of Errors** |
| Office scenario  (N=181) | Nursing Scenario  (N=181) | Semester A  (N=72) | Semester B  (N= 91) |
| **Completeness** | Comp. of actions | 150 | 294 | 132 | 160 | 736 |
| Comp. of flows | 34 | 44 | 28 | 24 | 130 |
| Comp. of nodes | 472 | 408 | 184 | 290 | **1354** |
| Comp. of roles | 39 | 13 |  | 19 | 71 |
| Comp. of start points/endpoints | 15 | 42 | 7 | 25 | 89 |
| **Irredundancy** | Redun. of actions | 377 | 328 | 109 | 236 | **1050** |
| Redun. of flows | 2 | 5 | 0 | 1 | 8 |
| Redun. of nodes | 18 | 79 | 171 | 200 | 468 |
| Redun. of roles | 49 | 171 |  | 5 | 225 |
| **Correctness** | Semantic Errors | 529 | 658 | 133 | 208 | **1528** |
| Syntactic Errors | 40 | 74 | 70 | 67 | 251 |

Of the 11 main categories (the yellow boxes in Figure 3), semantic errors – and particularly their first five primary subcategories (the first five grey boxes in Figure 3) – appear to be most common, suggesting that the rules underlying semantic logic in process modeling are difficult for learners to make sense of, in particular to novices. As we mentioned, novices lack abstracting skills, which mostly related to semantic errors, for example, to distinguish between necessary and unnecessary concepts, or to correctly map essential concepts into appropriate and required model constructs [‎25]. However, as noted, common errors also appeared in other categories as well. With respect to the persistence of errors, we needed to identify the error types in which there was no significant improvement over time. For this purpose, each participant was given two general scores for his performance in each of the three criteria, one for the office scenario (Task 1) and the other for the exam scenario (Task 2), resulting in a total of six scores per participant. We ran paired T-tests on these scores.

For the Completeness criterion, we found that the P value was 0.0001, indicating a significant improvement in performance from Task 1 to Task 2. A similar result was found for the Correctness criterion (0.0001), pointing to a significant improvement in performance as well. However, for the Irredundancy criterion the P value was 0.0261, indicating an improvement in performance, but not a significant one (if we set the threshold P value at 0.01).

Though most semantic error types decreased over time (as the Paired T-tests indicated), suggesting that learning and practice are effective in this area, still, some subcategories seemed to persist over time (e.g., incorrect timing and/or positioning of a construct). In addition, the large number of semantic errors throughout the semester implies that there remains room to further enhance the quality of models in the semantic context.

The category most resistant to learning appears to relate to redundancy of nodes. This is consistent with the findings in a study by [22], which examined the quality improvement of students' models in light of the same quality criteria as the present study. Similar to our results, in [22] there was no significant improvement in the Irredundancy criterion.

The implications of all these results will be discussed in detail in the next section.

# Discussion

As noted by Becker et al. [‎4], process modeling had its genesis in software engineering, but it is becoming increasingly important for many purposes besides the development of software. However, these contexts pose a challenge, due to the need to convert a business process reality into a rigid collection of formal symbols that will precisely reflect it [‎15]. As such, systems analysts (in particular, novices) who are required to cope with this task often experience difficulties that lead to modeling errors.

Our research contributes to the literature on two levels. At the first level, we identify error types made by novices when modeling processes and map these error types into a detailed classification. At the second level, we identify which error types are most common and which are most persistent (i.e., we differentiate between errors that improve over time and those that do not), and we suggest reasons for these findings.

With respect to the first level, Sadowska [‎31] was one of the first to propose an approach for assessing the quality of business process models, using BPMN. However, he created his metamodel based on relevant quality criteria found in the literature while conducting an experimental study. By contrast, we analyzed errors made by 181 participants, most of whom modeled three processes, for a total of **525** models. Our suggested categories were defined in a bottom-up approach, following a qualitative content-analysis research methodology, derived from the UAD language principles. The vast number of errors we identified (~5900) allowed us uniquely to categorize them into a 4-level, 52-element hierarchy of 38 error categories, subcategories, and irreducible types (see Figure 3). Our classification begins with three generic criteria that are commonly used when examining models in terms of quality, then branches into categories, subcategories, and types of errors in a manner oriented to business-process modeling. Yet some of the classification elements may be associated with other models as well (e.g., excess of generalization). On the one hand, the classification is rich and therefore makes it possible to pinpoint and refine errors. On the other hand, its hierarchical structure also allows for referring generically to groups of errors.

With respect to the second level, our findings add to a line of research motivated by the aim of improving both the practice and teaching of producing conceptual IS models [‎6] [‎24]. Many of these studies focus on the difference in skills between novices and more experienced modelers. Batra and Davis [‎3] explored the similarities and differences between novices and experts in the realm of database design. They found that the novices made measurably more errors, which they attributed to the fact that experts were better able "to categorize problem descriptions into standard abstractions," whereas novices struggled "to integrate the various parts of the problem description and map them into appropriate knowledge structures."

We suggest that a similar weakness in abstracting problem components is apparent among the novices in the present study. The ability to abstract is mainly expressed as three interrelated skills: an ability (1) to "separate the wheat from the chaff" – i.e., to distinguish between essential concepts that must be modeled (e.g., nurse) and unnecessary concepts that should not be modeled (e.g., arriving at the ER via ambulance); (2) to map essential concepts into appropriate constructs (e.g., identifying roles); and (3) to generalize (e.g., grouping objects into classes). We argue that Completeness and Irredundancy relate mainly to the first skill (separating the wheat from the chaff), while Correctness relates mainly to the last two skills (mapping and generalizing).

We found that abstraction is a skill which to some extent can be learned. Many of the measures studied here improved by the second task as students accrued experience, suggesting that they became more adept at abstraction with practice. Yet, along with the observed improvement, we identified difficulty in coping with model redundancy, which increased as the model became more complex. Redundancy in a model is expressed, as we noted, mainly via the first component skill of abstraction, i.e. as an inability to "separate the wheat from the chaff." As such, we can assume that this skill is the most challenging one for students to internalize.

The study has several limitations. The main limitation is that the second task was not the same for all the participants – a result of the fact that the second task was assigned as part of the final exam for a course offered in different semesters, and to ensure reliability of the exam results the scenarios in the two exams could not be the same. Second, due to the difficulty of the scenario in the semester A exam, participants were given the roles as part of the question. Hence, we do not have data regarding role completeness and role redundancy for the second task. Third, different groups of students taking the course were taught by two different lecturers. However, both lecturers used the same syllabus, the same presentations, and the same instruction format.

The present study will also provide a foundation for future work. In particular, we executed a preliminary task, which was completed during the first lesson of the course, before students even learned what process modeling is. The students were given ten minutes and asked to draw instructions for making an omelet. Students were told merely that they should use only pictures, not words; beyond that, no instructions, guidance, or explanations were given. The purpose of this preliminary task was to gather raw data about participants' process thinking abilities, even before the word "process" was said aloud. The results from this task can serve as a benchmark for participants' process thinking and will be used in our future research. On top of this, we also collected a range of information about participants in the present study, including but not limited to age, gender, cumulative grade point average, and course grade. Finally, each participant completed a self-efficacy questionnaire at the end of both tasks. In a follow-up study, we plan to analyze that data and explore correlations between different participant attributes. For instance, is there a correlation between participants' cumulative average and their modeling abilities? Does success in the preliminary task predict success in the subsequent tasks? Can we define process thinking and measure it?

In future research, we aim to define a learning methodology that will help novice analysts identify persistent errors and avoid them when designing a process model. We will test this methodology in new experiments and compare the results with the current study results.

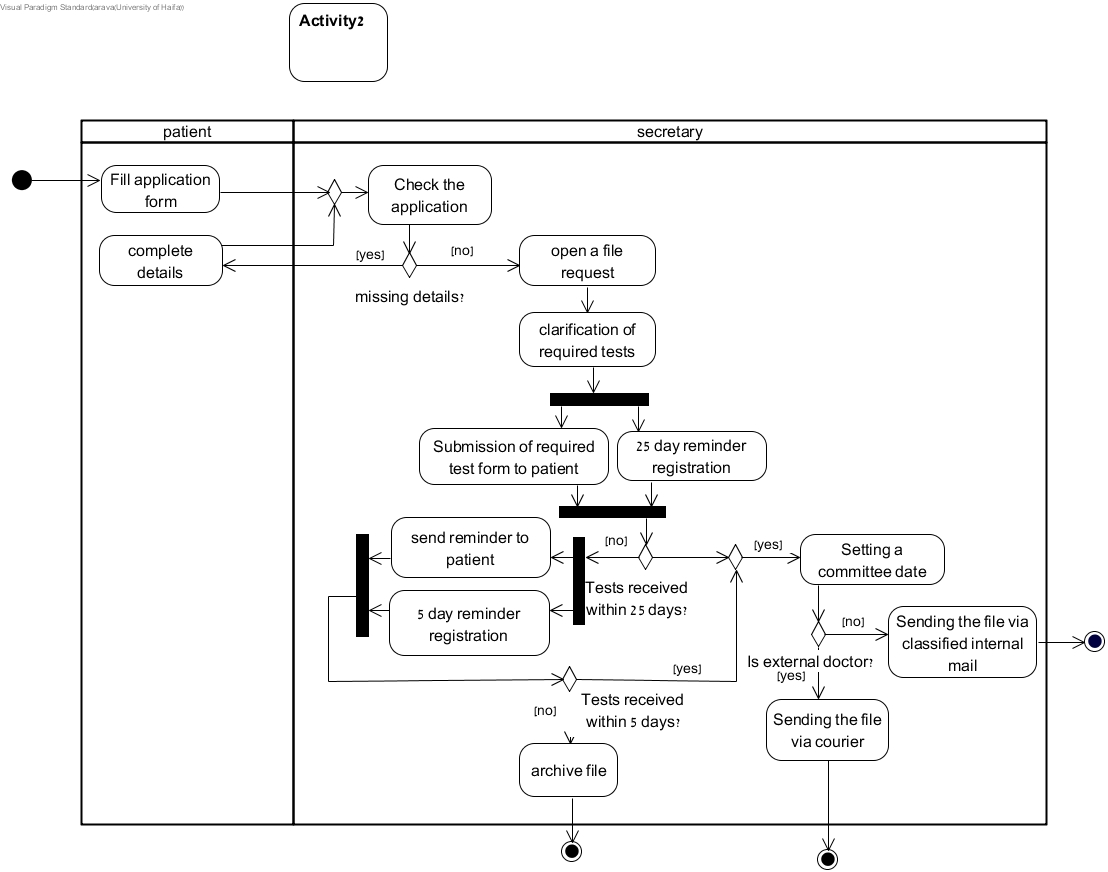
# References

1. Anda B, Dreiem H, Sjøberg DI, Jørgensen M (2001) Estimating software development effort based on use cases-experiences from industry. In: *Proceedings of the International Conference on the Unified Modeling Language*, Springer Berlin Heidelberg, pp 487-502
2. Armstrong, D., Gosling, A., Weinman, J., & Marteau, T. (1997). The place of inter-rater reliability in qualitative research: An empirical study. *Sociology*, *31*(3), 597-606.
3. Batra, D., & Davis, J. G. (1992). Conceptual data modelling in database design: similarities and differences between expert and novice designers. *International Journal of Man-Machine Studies*, 37(1), 83-101.
4. Becker, J., Rosemann, M., & Von Uthmann, C. (2000). Guidelines of business process modeling. In *Business process management* (pp. 30-49). Springer, Berlin, Heidelberg.
5. Beimel, D., & Kedmi-Shahar, E. (2019). Improving the identification of functional system requirements when novice analysts create use case diagrams: the benefits of applying conceptual mental models. *Requirements Engineering*, 24(4), 483-502.
6. Bider, I., Henkel, M., Kowalski, S., & Perjons, E. (2015, May). Teaching enterprise modeling based on multi-media simulation: a pragmatic approach. In *International Conference on E-Technologies* (pp. 239-254). Springer, Cham.
7. Santos, E., Campbell, J., Patel, D., Hindle, A., & Amaral, J. (2018). Syntax and sensibility: Using language models to detect and correct syntax errors. *In 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)* (pp. 311-322). IEEE.
8. BPMI. 2003. Business Process Modeling Language. Business Process Management Institute
9. Dahan M, Shoval P, Sturm A (2014) Comparing the impact of the OO-DFD and the Use Case methods for modeling functional requirements on comprehension and quality of models: a controlled experiment. *Requirements Engineering* 19(1):27-43
10. Elo, S., & Kyngäs, H. (2008). The qualitative content analysis process. *Journal of Advanced Nursing*, *62*(1), 107-115.
11. Figl, K., Recker, J., & Mendling, J. (2013). A study on the effects of routing symbol design on process model comprehension. *Decision Support Systems*, 54(2), 1104-1118.
12. Geambaşu, C. V. (2012). BPMN vs UML activity diagram for business process modeling. *Accounting and Management Information Systems*, 11(4), 637-651.
13. Genero M, Fernández-Saez AM, Nelson HJ, Poels G, Piattini M (2011) Research review: a systematic literature review on the quality of UML models. *Journal of Database Management* (JDM) 22(3):46-70
14. Hadar I, Hazzan O (2004) On the contribution of UML diagrams to software system comprehension. *Journal of Object Technology* 3(1): 143–156
15. Harrison, M. I., Koppel, R., & Bar-Lev, S. (2007). Unintended consequences of information technologies in health care—an interactive sociotechnical analysis. *Journal of the American Medical Informatics Association*, 14(5), 542-549.
16. Helfert, M. (2009). Challenges of business processes management in healthcare. *Business Process Management Journal.*
17. Jun, G. T., Ward, J., Morris, Z., & Clarkson, J. (2009). Health care process modelling: which method when? *International Journal for Quality in Health Care*, 21(3), 214-224.
18. Katz, A., & Shmallo, R. (2015). Improving relational data modeling through learning from errors. In *Proceedings of the IADIS Multi Conference of Computer Science and Information Systems MCCSIS, Theory and Practice in Modern Computing TPMC* (pp. 198-202).
19. Lindland, O. I., Sindre, G., & Solvberg, A. (1994). Understanding quality in conceptual modeling. *IEEE Software*, *11*(2), 42-49.
20. Mendling, J., Verbeek, H. M. W., van Dongen, B. F., van der Aalst, W. M., & Neumann, G. (2008). Detection and prediction of errors in EPCs of the SAP reference model. *Data & Knowledge Engineering*, 64(1), 312-329.
21. Mili, H., Tremblay, G., Jaoude, G. B., Lefebvre, É., Elabed, L., & Boussaidi, G. E. (2010). Business process modeling languages: Sorting through the alphabet soup. *ACM Computing Surveys* (CSUR), 43(1), 1-56.
22. Moody DL, Shanks GG (2003) Improving the quality of data models: empirical validation of a quality management framework. *Information Systems*, 28(6):619-650
23. Peleg M, Dori D (2000) The model multiplicity problem: experimenting with real time specification methods. *IEEE Transactions on Software Engineering* 26(8):742–759
24. Recker, J., & Rosemann, M. (2009). Teaching business process modelling: experiences and recommendations. *Communications of the Association for Information Systems*, 25(1), 32.
25. Recker, J., Safrudin, N., & Rosemann, M. (2012). How novices design business processes. *Information Systems*, 37(6), 557-573.
26. Reijers, H. A., Freytag, T., Mendling, J., & Eckleder, A. (2011). Syntax highlighting in business process models. *Decision Support Systems*, *51*(3), 339-349.
27. Reinhartz-Berger I, Dori D (2005) OPM vs. UML-Experimenting with comprehension and construction of web application models. *Empirical Software Engineering* 10:57–79
28. Roy, S., Sajeev, A. S. M., Bihary, S., & Ranjan, A. (2013). An empirical study of error patterns in industrial business process models. IEEE Transactions on Services Computing, 7(2), 140-153
29. Rozman, T., Polancic, G., & Horvat, R. V. (2008). Analysis of most common process modeling mistakes in BPMN process models. Eur SPI'2007.
30. Rumbaugh, J., Jacobson, I., & Booch, G. (1999). The unified modeling language. Reference manual.
31. Sadowska, M. (2015). An approach to assessing the quality of business process models expressed in BPMN. *e-Informatica Software Engineering Journal*, 9(1).
32. Shoval P, Frumermann I (1994) OO and EER Conceptual Schemas: A comparison of User Comprehension. *Journal of Database Management* 5(4):28-38
33. Shoval P, Shiran S (1997) Entity-relationship and object-oriented data modeling-an experimental compari-son of design quality. *Data & Knowledge Engineering* 21(3):297-315
34. Söderström, E., Andersson, B., Johannesson, P., Perjons, E., & Wangler, B. (2002, May). Towards a framework for comparing process modelling languages. In International Conference on Advanced Information Systems Engineering (pp. 600-611). Springer, Berlin, Heidelberg.
35. Srinivasan, A., & Te' eni, D. (1995). Modeling as constrained problem solving: an empirical study of the data modeling process. *Management Science*, 41(3), 419-434.
36. Van der Aalst, W. M. (1999). Formalization and verification of event-driven process chains. *Information and Software Technology*, 41(10), 639-650.

# Appendix

Table 11: Summary of Task 1 and 2 criteria and categories along with their scoring method

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Category** | **Scoring method** |
| Completeness | Completeness of actions | **+1** point for each detected action |
| Completeness of flows | **–1** point for each missing flow |
| Completeness of nodes | **+1** point for each detected node |
| Completeness of roles | **+1** point for each detected role |
| Completeness of starting/endpoints | **+1** point for each detected start point/endpoint |
| Completeness: Measured by summing all points. The **higher,** the better. | | |
| Irredundancy | Redundancy of actions | **1** point for each redundant action |
| Redundancy of flows | **1** point for each redundant flow |
| Redundancy of nodes | **1** point for each redundant node |
| Redundancy of roles | **1** point for each redundant role |
| Irredundancy: Measured by summing all points. The **lower,** the better. | | |
| Correctness  (Semantic) | Excess of generalization | **1** point for each excess generalization |
| Lack of generalization | **1** point for each lack of generalization |
| Incorrect timing and/or positioning of a construct | **1** point for each wrong timing/position |
| Incorrect use of one construct instead of another | **1** point for each incorrect use |
| Miscellaneous semantic error | **1** point for each miscellaneous semantic error |
| Correctness  (Syntactic) | Syntactic action errors | **1** point for each syntactic action error |
| Syntactic flow errors | **1** point for each syntactic flow error |
| Syntactic node errors | **1** point for each syntactic node error |
| Syntactic role errors | **1** point for each syntactic role error |
| Correctness: Measured by summing all points. The **lower,** the better. | | |

Figure 4: Recommended solution for semester A scenario of Task 2

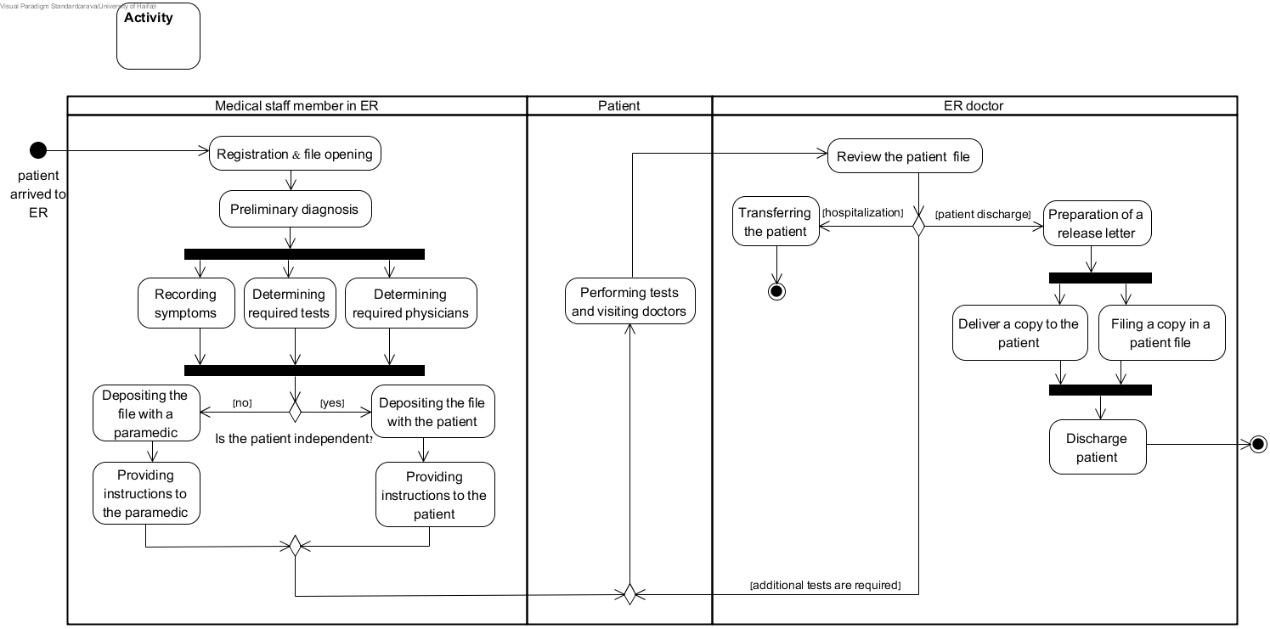


Figure 5: Recommended solution for semester B scenario of Task 2

1. In the Semester A exam we provided the roles, and so students were not required to model these. [↑](#footnote-ref-1)