Defining and Providing Pragmatic Interoperability: The MIDAS Middleware Case

Elivaldo Lozer Fracalossi Ribeiro^{1,2}, Daniela Barreiro Claro^{1} (Advisor), Rita Suzana Pitangueira Maciel^{3} (Co-advisor)

^{1}FORMAS Research Group, PGCOMP, Federal University of Bahia (UFBA), Salvador, Brazil

^{2}Federal University of Southern Bahia (UFSB), Porto Seguro, Brazil

^{3}MATisSE Research Group, PGCOMP, Federal University of Bahia (UFBA), Salvador, Brazil

elivaldolozer@ufsb.edu.br, dclaro@ufba.br, rita.suzana@ufba.br

Abstract. *Modern information systems are becoming increasingly complex due to the need to combine heterogeneous software. A common understanding of interoperability issues is not a trivial task since complex systems may contain many independent software components. This work presents a Conceptual frAmework for Pragmatic InTeroperAbiLity (CAPITAL) to enhance a pragmatic interoperability unified definition. We evaluate our framework through a modeling and coding guide, a controlled experiment, and applying CAPITAL in the Cloud Computing domain. Results suggest that CAPITAL positively influences the understanding, modeling, and codification of pragmatic interoperability solutions, facilitating pragmatic interoperability standardization opportunities.*

1. Introduction

Moderns solutions mean a variety of technologies, such as programming languages, operating systems, databases, Application Programming Interface (API), protocols, data formats, among others [Zhang et al. 2013]. The heterogeneity of technologies hinders two or more systems from working together to solve a specific problem [Armbrust and et al. 2010]. In this situation, users are dependent on the provider since they cannot easily change nor move their applications. This problem caused by dependence on the provider is known as lock-in [Opara-Martins et al. 2016]. Vendor lock-in problem occurs when customers are dependent (i.e., locked-in) on a specific provider, for instance, due to the different technologies adopted by different providers. This heterogeneity hampers communication, and the absence of the communication hampers interoperability among systems [Opara-Martins et al. 2016].

Interoperability is the ability of multiple systems to work together to provide transparent communication regardless of the providers' differences [Zhang et al. 2013]. Despite the absence of consensus [Ribeiro et al. 2019a], interoperability may be classified into syntactic, semantic, and pragmatic levels [Asuncion et al. 2010]. In summary, (i) syntactic level ensures the exchange of information among systems based on message standard, (ii) semantic interoperability is concerned with the communication meaning, and (iii) pragmatic interoperability provides that systems share the same communication intention [Asuncion et al. 2010, Maciel et al. 2017]. Another related term, full interoperability is achieved when a system reaches all desired interoperability levels [Maciel et al. 2017]. For instance, some systems may require only syntactic interoperability, while other systems may require syntactic, semantic, and pragmatic levels.

Some research works present solutions to provide the different levels of interoperability in several domains [Lee et al. 2007, Neiva et al. 2016, Tamani and Evripidou 2007, Webster 2014]. While syntactic and semantic levels have consensus definitions by the community, there is a lack of canonical understanding for pragmatic interoperability. Asuncion and Sinderen claim that the definition of pragmatic interoperability is unstable, and so the absence of a collective understanding can cause incompatible solutions and ambiguous interpretations of messages meaning [Asuncion et al. 2010].

In this context, this Ph.D. thesis *investigates and provides a model for pragmatic interoperability among systems*. To that end, we present a Conceptual frAmework for Pragmatic InTeroperAbiLity (CAPITAL). CAPITAL comprises a canonical model, a textual definition, and a Z notation. We present a CAPITAL initial version in [Ribeiro et al. 2019a]. In the CAPITAL final version, we updated the canonical model and textual definition, and we added the Z notation. We argue that this work is relevant to Information Systems since interoperability is a challenge in Information Systems in Brazil between 2016 and 2026 [Maciel et al. 2017].

We employed a top-down strategy in the specification and validation process of our CAPITAL framework comprising of four activities: *definition*, *investigation*, *specification*, and *evaluation* (Figure 1). We started by unifying the definition. We performed a literature review searching for definitions on pragmatic interoperability, and then we listed the eight most cited definitions. Afterward, we analyze each definition and identify four recurring terms to present a unified definition. We specify our conceptual framework and, finally, we evaluate CAPITAL in three ways: (i) exploratory study towards a modeling and coding guide, (ii) controlled experiment, and (iii) an exploratory study in the Cloud Computing domain.

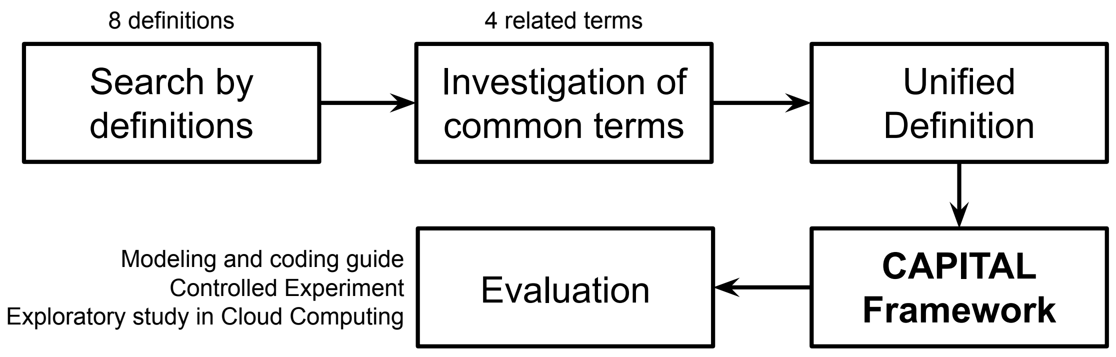

Figure 1. Activities for CAPITAL framework.

Additionally, we apply the Kolb cycle [Kolb 1984] as guideline. Kolb's methodology describes that the construction or learning process is based on the continuous and cyclical execution of four stages: *act*, *reflect*, *conceptualize*, and *apply*. We adapted and performed this methodology in a continuous and cyclical execution of the first three steps (*act*, *reflect*, and *conceptualize*). This methodology allowed us to adjust the CAPITAL framework at any stage.

The remainder of this paper is organized as follows: Section 2 presents some related work; Section 3 describes our CAPITAL framework; Section 4 presents the CAPITAL evaluation (modeling and coding guide, controlled experiment, and exploratory study considering the Cloud Computing domain); and Section 5 concludes our paper with some envision work.

2. State of the Art

This section presents our most relevant related work. Subsection 2.1 discusses some pragmatic interoperability definitions and then presents our unified definition. Subsection 2.2 presents some solutions that address pragmatic interoperability.

2.1. Interoperability Definitions

Pragmatic interoperability definitions towards a unified definition required a specific literature review. We present and discuss these results in [Ribeiro et al. 2019a]. The eight pragmatic interoperability definitions from the literature are summarized in Table 1. The set of definitions evidences that there is no consensus of pragmatic interoperability definition.

Table 1. Definitions of Pragmatic Interoperability in chronological order

Definition [Ref.]
Pragmatic interoperability ensures that the messages exchanged have the desired effect. Usually, this occurs by sending and receiving multiple messages in a specific order, defined by a protocol [Pokraev et al. 2005].
Pragmatic interoperability is achieved when one system knows the methods and procedures of other systems [Lee et al. 2007].
The pragmatic web is a set of pragmatic contexts about semantic resources [Tamani and Evripidou 2007].
Pragmatic interoperability is the compatibility between the intended effect and the actual effect of message exchange [Asuncion and van Sinderen 2010].
Pragmatic interoperability is the compatibility between intended use and actual use of message within a relevant shared context [Asuncion et al. 2011].
At the system level, pragmatic interoperability is a shared understanding between intended use and messages actual use within a context. At the business level, pragmatic interoperability considers the compatibility of business intentions, business rules, and organizational systems policies [Asuncion et al. 2011].
Pragmatic interoperability is achieved when processes from different contexts are compounded to support a common intention. The integration emphasis is context-awareness [Liu et al. 2014].
Pragmatic interoperability uses syntax and semantics as a tool to achieve goals [Webster 2014].

Despite the lack of consensus, we identify in Table 1 the existence of repeated terms. For that reason, we list the related terms to make up a unified definition. The literature presents several definitions for related terms, such as:

- **use**: (i) how to realize the sender's intention; (ii) how the receiver interprets the intended information; and (iii) how systems use data;
- **intention**: (i) message objective; (ii) actions desired by the message; and (iii) possible desired state that a sender can achieve through collaboration;
- **context**: (i) circumstance in which the message is shared; (ii) contextual dimensions: why, how, when, who, where, and what; and (iii) set of contexts in semantic resources; and
- **effect**: (i) relationship between information, action, and context; (ii) it requires the receiver to understand the message intent deeply; and (iii) it can be accomplished by sending and receiving messages in a specific order.

Based on these terms, we present our pragmatic interoperability unified definition as follows:

*At the system level, pragmatic interoperability is achieved when there is a shared understanding of the **intention** and **context** necessary for communication, aiming to provide the correct **use** of the message and produce results within the expected **effects**.*

2.2. Interoperability Solutions

State of the art provides evidence that there is no consensus on the definition. This lack of consensus makes it difficult for pragmatic interoperability since the solutions apply different strategies.

Authors in [Neiva et al. 2016] introduce PRIME (Pragmatic Interoperability to Meaningful Collaboration), an architecture to support pragmatic interoperability in the collaborative development of scientific workflows. PRIME provides a mechanism for scientists to find services that meet their expectations. The architecture receives some search parameters, and it returns a list of services that satisfy the request. Although this search results in services ordered by relevance, contextual elements are not applied to compare two or more services.

Authors in [Liu et al. 2014] present a framework that aims to interoperate data from a radiology department pragmatically. Similar to other work, authors offer a solution for a specific domain, and they do not consider contextual elements when modeling the pragmatic model.

Authors in [Lee et al. 2007] propose a context-aware geospatial data and service integration framework based on the combination of syntactic, semantic, and pragmatic models. Although the authors use notions of context, the pragmatic model considers only contextual aspects related to place and time. The absence of a representation model can limit the representation of other scenarios. Additionally, even if not explicit, authors consider that the pragmatic model has a single intention.

The solutions presented are domain specific, i.e., each approach fulfills problems in a specific area: scientific workflows, radiology field, and geospatial data. Similar to the definitions, there is no consensus on the elements needed to provide pragmatic interoperability among systems, e.g., contextual and intentional data.

3. CAPITAL: a Conceptual frAmework for Pragmatic InTeroperAbiLity

According to Figure 2, our CAPITAL framework is composed of three artifacts: a canonical model, a textual definition, and schemes in Z notation.

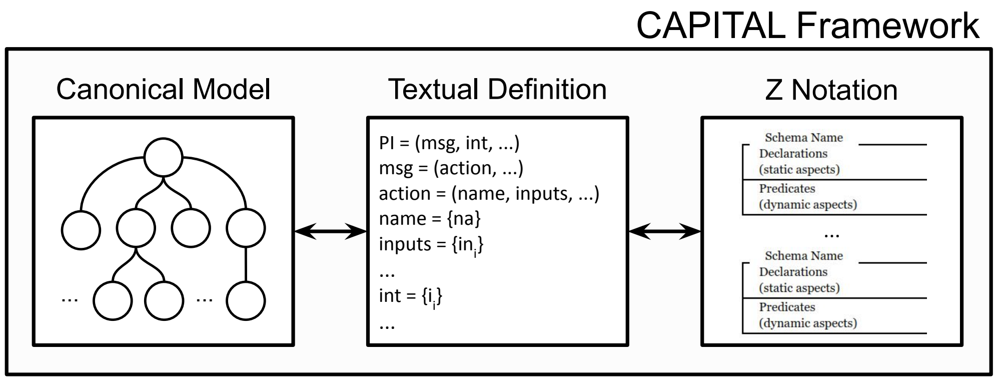

Figure 2. CAPITAL's Artifacts: Canonical model, textual definition, and schemes in Z notation.

Our canonical model is composed of trees and key-values [Schreiner et al. 2015] and defines terminologies related to pragmatic interoperability towards a unified definition. Figure 3 depicts the canonical model of our CAPITAL framework. Although there are other approaches to model context, we apply the 5W1H1[1] format [Isoda et al. 2005]. This format is frequently employed in context-sensitive systems.

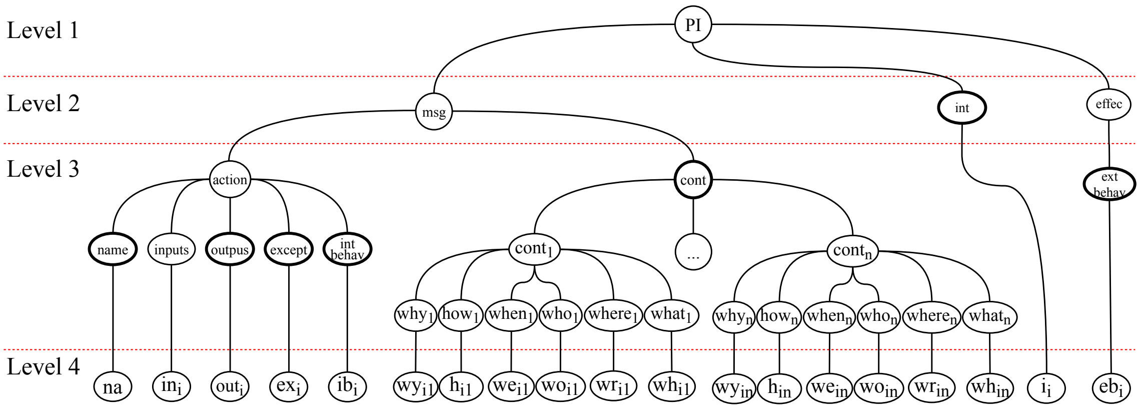

Figure 3. Canonical model of our CAPITAL framework. Mandatory elements are bold.

Level 1 indicates the modeled term: pragmatic interoperability (PI). PI is composed of three sets of information at Level 2: *msg* (representing the message), *int* (representing the **intention**), and *effec* (representing the **effect**). Level 3 details the elements of the previous level: (i) *msg* contains *action* (representing the **use**) and *cont* (representing the **context**), and (ii) *effec* contains the behavior performed by the receiver (*ext behav*). Action is composed of the service *name*, *inputs*, *outputs*, exceptions (*except*), and behavior (*int behav*). The *cont* element is composed of a set of contexts.

According to the key-value structure, value(s) in Level 4 is related to the key in Level 3. Therefore, *cont* contains a set of sub-levels where each sub-level represents a specific context (*cont*$_n$). Each specific context stores the information in the 5W1H format. We consider a set of contexts since each context represents a possible circumstance

[1]Why, When, Who, Where, What, and How

or situation. We intend to provide a domain-independent framework. Therefore, other elements can be added to the model.

Additionally, we also provide a textual definition of the canonical model. The text notation provides a textual description of the elements involved in the pragmatic interoperability concept. While the canonical model presents a structural relationship among elements, the textual description enables a non-formal understanding of the framework. For instance, the following tuples describe the model's *inputs*, exceptions (*except*), and contexts (*cont*), respectively:

- $inputs = \{in_i\}$, $i = 1..j$, where in_i is the i-th input argument;
- $except = \{ex_i\}$, $i = 1..j$, where ex_i is the i-th exception; and
- $cont = (cont_1, \ldots, cont_n)$, where $cont_n$ is the n-th context.

Finally, Z notation [Spivey 1989] is a formal language based on mathematical concepts for specifying computing systems. We apply Z notation in the CAPITAL framework to facilitate the implementations of pragmatic interoperability. The next section presents the CAPITAL framework evaluation.

4. CAPITAL Framework Evaluation

This section presents evaluation results and introduces some discussions.

4.1. Modeling and Coding Guide Exploratory Study

The purpose of this exploratory study is to evaluate the effectiveness, completeness, and mandatory elements of the CAPITAL framework on scenarios with pragmatic interoperability. In the first study, we modeled the CAPITAL framework in four distinct scenarios that aim to provide a modeling and coding guide. We consider the following scenarios: (i) service from laboratory tests [Asuncion et al. 2011], (ii) service that checks DNA sequence ancestry [Neiva et al. 2016], (iii) service of the car's Bluetooth, and (iv) service related to the public security domain.

Contextual elements vary depending on the scenario. Scenario 1 contains one contextual variable (*importance*) that can assume three values: *urgency*, *emergency*, and *normal*. Scenario 2 contains two contextual variables (*sequence* and *method*) and each variable can assume two values: *sequence* can be *DNA* or *RNA*; and *method* can be *local* or *global*. Scenario 3 contains two contextual variables (*distance* and *relationship*) and each variable can assume three values: *distance* can be *far*, *moderate*, or *near*; and *relationship* can be *unknown*, *known*, or *familiar*. Scenario 4 contains one contextual variable (*crimes*) that can assume four values: *child pornography*, *graffiti*, *stolen car*, and *person with firearm*.

The representation for each context considers distinct 5W1H elements. For instance, the first context in scenario 1 (importance: emergency) uses *why*, *how*, *when*, *who*, and *what*, while first context in scenario 2 (sequence: DNA) uses only *who*, *what*, and *who*.

The scenarios aim to verify whether the framework answers the following research questions:

RQ1 *Does the CAPITAL framework model scenarios with pragmatic interoperability?* This question focuses on the *effectiveness* of our framework as we intend to verify if CAPITAL may represent pragmatic interoperability among different scenarios.

RQ1.1 *Does the framework consider all the elements necessary to represent the pragmatic interoperability of the scenarios?* This question focuses on the *completeness* of our framework: CAPITAL considers all the elements necessary to provide pragmatic interoperability?

RQ1.2 *What elements are needed to provide pragmatic interoperability?* This question focuses on the *mandatory elements* of our framework: CAPITAL considers all mandatory elements when it models pragmatic interoperability among systems?

RQ1.3 *The strategy chosen to model context was enough?* This question investigates whether the 5W1H format is capable of representing varied contexts.

Scenario 4 is briefly presented below. The other scenarios are presented in detail by [Ribeiro 2020].

In this scenario, we consider smart security cameras that recognize and report the occurrence of new crimes. After receiving a new crime, the system forwards the incident to the appropriate agency. Depending on the crime, the system sends the notification to the Federal Police of Brazil (PFB), Military Police of Bahia State (PMBA), or City Guard of Salvador city (GMSSA). Each agency acts according to its internal rules, i.e., outside the system scope.

Problems on pragmatic interoperability occur when, for instance, the security cameras report “person with a firearm” and “the person” is a police officer. We consider that there is no crime when a police officer carries a firearm. In this case, the system must understand the situation context to trigger the appropriate agency or not. *Uniform* and *location* are examples of elements that can assist in decision making.

Based on our CAPITAL framework, we represent the **action** as follows:

- **Name**: report a crime
- **Internal behavior**: system (i) receives the possible crime, (ii) determines if there is a crime, and (iii) triggers the appropriate agency
- **Input**: the crime (literal), date (date), time (time), and location (literal)
- **Output**: crime (boolean) and appropriate agency (literal)
- **Exceptions**: non-existent crime, inactive service, and timeout

We model the **context** based on four crimes and some contextual situations. We assume the system recognizes the following crimes: *child pornography*, illegal *graffiti*, *stolen car*, and *person with firearm*. These crimes are modeled according to contexts $Cont_1$, $Cont_2$, $Cont_3$, and $Cont_4$, respectively. Figure 4 presents the canonical model for scenario 4. Contexts are presented separately in 4(a), 4(b), 4(c), and 4(d) parts.

Each context represents a crime. We assume the Federal Police of Brazil (PFB) is responsible for the *child pornography* crime, the City Guard of Salvador city (GMSSA) is responsible for the *graffiti* crime, and the Military Police of Bahia State (PMBA) is responsible for *stolen car* and *person with firearm* crimes.

The model considers some contextual elements of each crime. For instance, *graffiti* (**why** in $Cont_2$) crime can occur in a *square*, *monument*, or *building* (**where**), except when *practiced by artists* (**who**). In this crime, the City Guard of Salvador city (**when**) must be called through the API *gmssa.com* (**how**). Similarly, *person with firearm* (**why** in $Cont_4$) crime can occur in anywhere, *except in police agencies* (**where**). We assume that

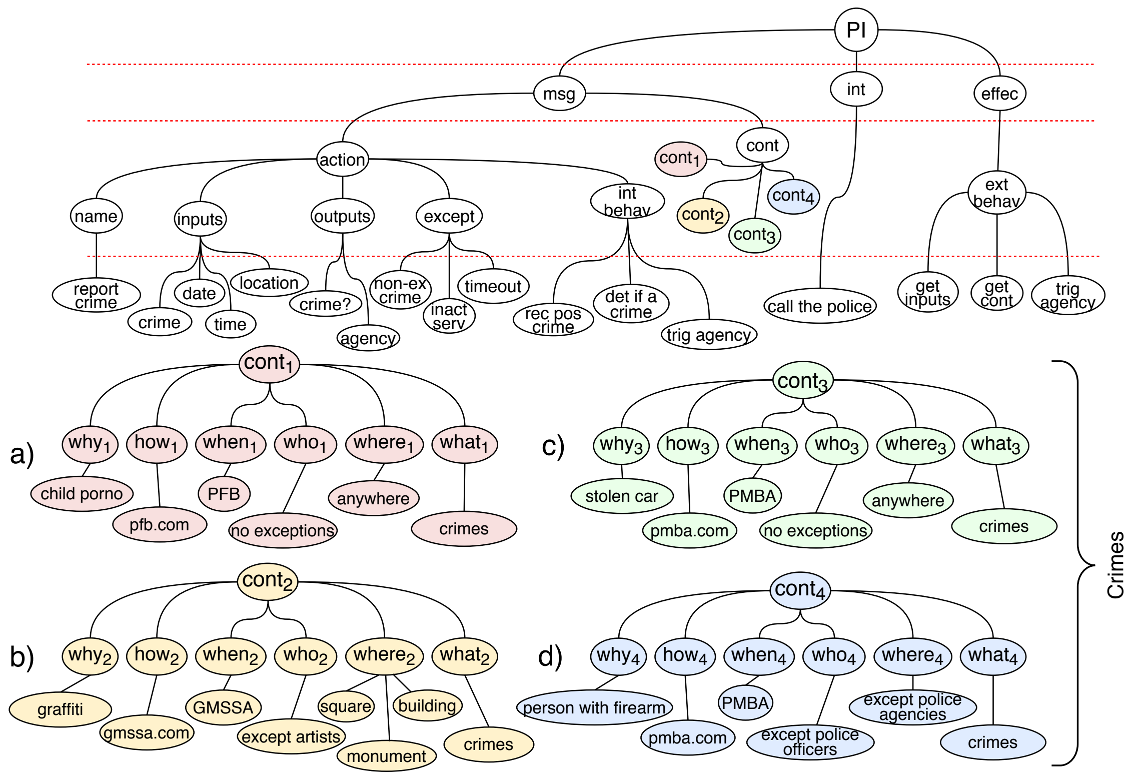

Figure 4. The canonical model for the public security domain scenario. Figure 4(a), Figure 4(b), Figure 4(c), and Figure 4(d) represent the crimes: child pornography (in red), graffiti (in yellow), stolen car (in green), and person with firearm (in blue), respectively.

this crime is not committed by police officers (**who**). In this crime, the Military Police of Bahia State (**when**) must be called through the API *pmba.com* (**how**).

The following definitions list abstract data types in Z notation in this scenario:

isCrime ::= ‘`yes`’ | ‘`no`’
CrimesTypes ::= ‘`child porno`’ | ‘`graffiti`’ | ‘`stolen car`’ |
 ‘`person with firearm`’
RetLogic ::= ‘`yes`’ | ‘`no`’
why ::= ‘`CrimesTypes`’
how ::= ‘`pfb.com`’ | ‘`gmssa.com`’ | ‘`pmba.com`’
when ::= ‘`PFB`’ | ‘`GMSSA`’ | ‘`PMBA`’
who ::= ‘`no exceptions`’ | ‘`not artists`’ | ‘`not police officers`’
where ::= ‘`anywhere`’ | ‘`square`’ | ‘`monument`’ | ‘`building`’ |
 ‘`not police agencies`’
what ::= ‘`crimes`’

Smart security cameras sender one notification with *possible crime*, *date*, *time*, and *location*. The last three elements can be sent automatically depending on the device used in the notification. Other elements may be considered.

Notification

$possCrime : CrimesType$
$date : Date$
$time : Time$
$location : \mathrm{seq}\, Char$

In this scenario, we have one contextual variable (*crimes*) that can assume four values: *child pornography*, *graffiti*, *stolen car*, and *person with firearm*.

CrimesContext

$\Xi Child\_porn$
$\Xi Graffiti$
$\Xi Stolen\_car$
$\Xi Person\_w\_fireman$

Each context contains the all contextual elements: *why*, *how*, *when*, *who*, *where*, and *what*. The value of each contextual element depends on the context. The following specification shows the context for *child pornography*. The other crimes are modeled in a similar way.

Child_porn

$whyChPorn : why$
$howChPorn : how$
$whenChPorn : when$
$whoChPorn : who$
$whereChPorn : where$
$whatChPorn : what$

$whyChPorn =$ ‘`child porno`’
$howChPorn =$ ‘`pfb.com`’
$whenChPorn =$ ‘`PFB`’
$whoChPorn =$ ‘`no exceptions`’
$whereChPorn =$ ‘`anywhere`’
$whatChPorn =$ ‘`crimes`’

Given a set of contexts and one notification, we define public security service with pragmatic interoperability (*PublicSecurityWithPI*) as follows. The central idea is to capture the possible crime (*Notification.possCrime*) and check for divergence between the contextual elements of the notification (*data*, *time*, and *location*) and the context (*who* and *where*). If there are no divergences, the system triggers the most appropriate agency.

```
PublicSecurityWithPI
$\Xi Notification$
$\Xi CrimesContext$
$isCrime? : isCrime$
$apprAgency? : when$
$crime : CrimesType$
$local : where$
----------
$crime = Notification.possCrime$
$local = Notification.location$
$isCrime? =$ ‘yes’                                [Based on location only]
if $crime = CrimesContext.Child\_porn.whyChPorn$ then
    if $local = CrimesContext.Child\_porn.whereChPorn$ then
        $apprAgency? =$ ‘PFB’
elseif $crime = CrimesContext.Graffiti.whyGraf$ then
    if $local = CrimesContext.Graffiti.whereGraf\_a \lor$
      $local = CrimesContext.Graffiti.whereGraf\_b \lor$
      $local = CrimesContext.Graffiti.whereGraf\_c$ then
        $apprAgency? =$ ‘GMSSA’
elseif $crime = CrimesContext.Stolen\_car.whyStCar$ then
    if $local = CrimesContext.Stolen\_car.whereStCar$ then
        $apprAgency? =$ ‘PMBA’
elseif $crime = CrimesContext.Person\_w\_fireman.whyPFire$ then
    if $local = CrimesContext.Person\_w\_fireman.wherePFire$ then
        $apprAgency? =$ ‘PMBA’
else $isCrime? =$ ‘no’
```

Finally, we specify the exceptions as follows.

```
Exceptions
$\Xi Notification$
$error! : RetLogic$
----------
if $Notification.possCrime \notin why$ then $error! =$ ‘yes’
else $error! =$ ‘no’
```

The Z notation formalizes the canonical model of Figure 4. At the current stage, we evaluate crime based only on the reported location.

We evaluated the effectiveness, completeness, and mandatory elements of the CAPITAL framework based on the four scenarios. Our framework recommends the mandatory elements to provide pragmatic interoperability based on mandatory elements in each scenario. These results suggest that (i) our framework might be generalized to other scenarios and (ii) framework identifies the mandatory elements to provide pragmatic interoperability based on different scenarios. Our intuition is that a common and shared understanding of pragmatic interoperability guides towards full interoperability, even with different strategies and systems.

4.2. Controlled Experiment

The second study is a controlled experiment that investigates whether our framework eases to understand the concept and interpret scenarios with pragmatic interoperability. This experiment is structured according to guidelines defined by [Wohlin et al. 2012].

In Goal-Question-Metric (GQM) template [Basili and Rombach 1988], the objective of this experiment is *analyze <the CAPITAL framework> for the purpose of <evaluation> with respect to <understandability, completeness, consistency, conciseness, and performance> from the point of view of <developers, professors, and students> in the context of <scenarios with pragmatic interoperability>*. Based on the objective, we defined the following research question: *Does the use of the CAPITAL framework influence the understanding and modeling of pragmatic interoperability scenarios?*

Since there is no similar framework in the literature, we investigate whether there is a difference between modeling scenarios (i) with the use of CAPITAL framework and (ii) with the use pragmatic interoperability definition founded in the literature.

The design of our experiment comprises *one factor* with *two treatments*. The factor considered was *the modeling technique*, and the treatments were *CAPITAL framework* and *ad-hoc* (i.e., with and without CAPITAL framework). Therefore, we carried out this experiment based on two groups: *CAPITAL group* and *Control group*. The *CAPITAL group* modeled pragmatic scenarios with the CAPITAL framework, and *Control group* modeled pragmatic scenarios only with pragmatic interoperability definitions from literature (i.e., without the CAPITAL framework, ad-hoc).

According to our design, we created two similar groups based on the participants' experience and status: *CAPITAL group* and *Control group*. Each participant belongs only to one group, and we recruited 46 participants in this study.

The dependent variables are the attributes *understandability*, *completeness*, *consistency*, *conciseness*, and *performance*. Independent variable is the *modeling technique*. We vary the independent variable in two values: *CAPITAL framework* and *ad-hoc*.

Based on the objective, research question, and attributes, we define five sets of null and alternative hypotheses. Each hypothesis corresponds to an attribute. For instance, concerning *understandability*, we define the following hypothesis:

- $H1_0$: the use of CAPITAL does not influence the understanding of scenario with pragmatic interoperability
- $H1_a$: the use of CAPITAL influences the understanding of scenario with pragmatic interoperability

Our framework simplifies the pragmatic interoperability definitions from the literature based on our definition unified. Results suggest that the CAPITAL framework positively influenced the *understandability* (p-value = 0.000187), *completeness* (p-value = 0.000032), and *consistency* (p-value = 0.000281) of scenarios. By contrast, the CAPITAL framework did not influence *conciseness* (p-value = 0.174552) and *performance* (p-value = 0.185026). We evaluate the probability for each attribute based on the t-test (two-tailed) with a 95% confidence level.

4.3. MIDAS exploratory study

The volume of digital data generated by collaboration among systems grows exponentially [Reinsel et al. 2018]. Consequently, this data needs to be stored and available to both consumers and organizations anytime and anywhere. Cloud Computing has emerged to fulfill some of these requirements. Cloud Computing is a paradigm that enables access to a ubiquitous and on-demand network of logical and physical resources (e.g., applications, platforms, and hardware) as services. Software as a Service (SaaS), Platform as a service (PaaS), Data as a Service (DaaS), and Database as a Service (DBaaS) are instances of cloud services [Armbrust and et al. 2010, Mell and Grance 2011].

In the third study, we develop a prototype of the pragmatic middleware for SaaS and DaaS: MIDAS 3.0. Similar to the previous versions [Marinho et al. 2016, Vieira et al. 2017, Ribeiro et al. 2018, Ribeiro et al. 2019b, Mane et al. 2020], MIDAS 3.0 intermediates communication between SaaS applications and heterogeneous data sources (e.g., DaaS and DBaaS) independently of API. We incorporated our framework into MIDAS to discuss and present a middleware version for pragmatic interoperability. The following section describes and discusses the MIDAS 3.0 architecture.

4.3.1. MIDAS 3.0

MIDAS 3.0 is our first attempt to address pragmatic interoperability in cloud environments. MIDAS 3.0 recognizes (i) SQL and document-based NoSQL queries, (ii) data stored in a single or multiple DaaS/DBaaS providers, and (iii) queries with and without data join. Based on the CAPITAL framework, MIDAS 3.0 stores contextual information for each data source in the DIS. This information enables pragmatic MIDAS to understand the SaaS intent and access the appropriate source.

We reuse functionalities of all modules and components of MIDAS 2.0 [Mane et al. 2020]. This suggests that appropriate syntax and semantic are necessary before implementing the pragmatic concept. We upgrade *Query Builder*, *Crawler*, and *DIS* components, and we add a new component: *Pragmatism Mapping*. Although SaaS is not within MIDAS scope, it requires adjustments since it must send contextual elements to MIDAS. The encoding of this step depends on the device used in the query. Figure 5 depicts MIDAS 3.0 architecture. The layered presentation illustrates the modules and components provided and reused by each level of interoperability.

The *Pragmatism Mapping* component (i) receives the SaaS request, (ii) separates the query and format from pragmatic information, such as user intent and contextual elements, (iii) forwards the query and format to *Query Decomposer*, and (iv) forwards the pragmatic information to the *Query Builder*. In MIDAS 2.0, SaaS sends only the query and the desired return format. Nonetheless, MIDAS 3.0 requires pragmatic information (e.g., intention and contextual elements) to facilitate the query in the source desired by the user and to provide pragmatic interoperability.

The following section presents the MIDAS 3.0 evaluation. We implement the MIDAS 3.0 based on service from laboratory tests [Asuncion et al. 2011]. Our pragmatic MIDAS considers elements of the CAPITAL framework aiming to provide pragmatic interoperability among cloud services.

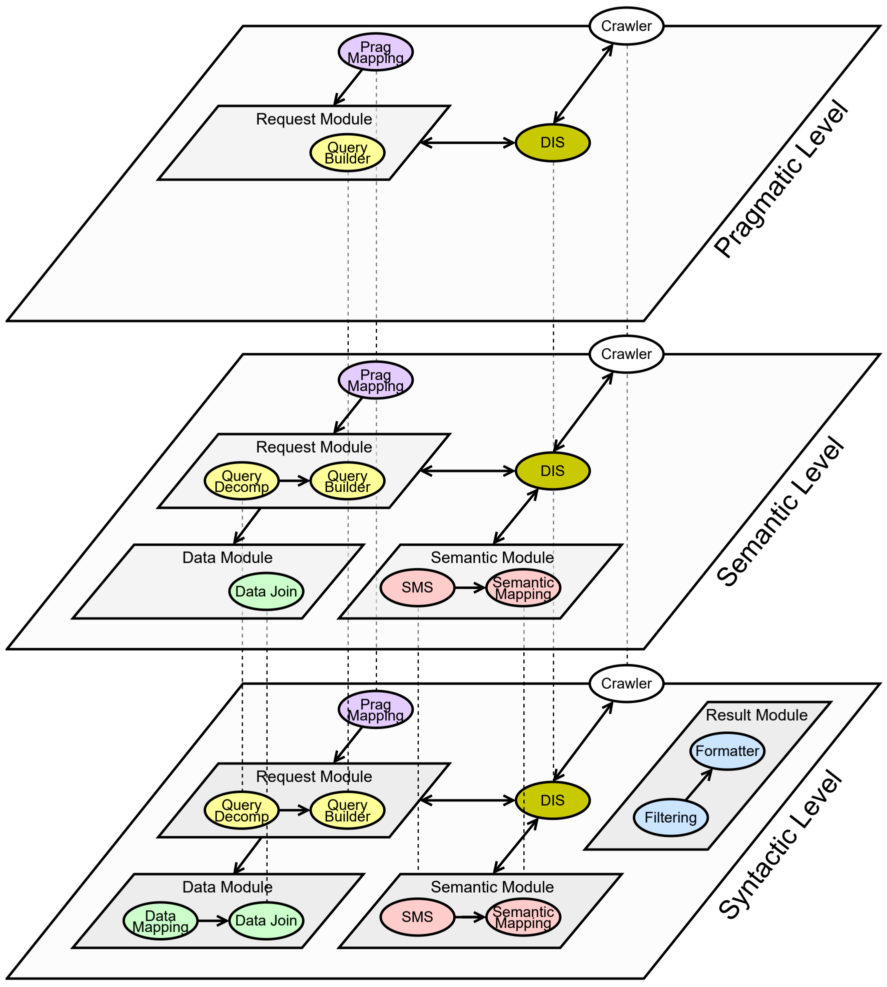

Figure 5. MIDAS 3.0 architecture.

4.3.2. MIDAS 3.0 evaluation

This evaluation assumes that MIDAS recognizes two DaaS and SaaS requests data stored in a single DaaS. Although the scenario is concerned with retrieving a patient's laboratory tests, we assume a query that requests a patient's blood type. We also assume that a query must be performed by an *ambulance*, *hospital*, or *patient*. Each requester has some specific characteristics, such as time waiting for the return. Consequently, *DIS* contains two DaaS and three contexts, one context for each requester. Each context is modeled based on the 5W1H template. The key for each context is the *who* element. Since automatic detection of contextual elements is outside this work's scope, we set contexts and intentions, and we capture the requester. The intention depends on the requester, for instance: the *patient*'s intention is to receive the result *within 24 hours*.

We performed three experiments to evaluate our MIDAS 3.0. For this, we develop an application to simulate a SaaS service. Our prototype is available at `http://pragmidas.herokuapp.com/test`

The first experiment evaluates the overhead of our *Pragmatism Mapping* module in MIDAS middleware. For this, we submitted: (i) 100 queries without our *Pragmatism*

Mapping module, and (ii) 100 queries with our module. In both tasks, we perform queries for a single DaaS, varying the number of records. Queries without our pragmatic module were on average faster than queries with our module: (i) 12.2% for queries with 100 returned data, (ii) 15.2% for queries with 1000 returned data, and (iii) 40.5% for queries with 10000 returned data. The results show an overhead caused by the *Pragmatism Mapping* module of up to 40.5%. The *Data Join* component is one of MIDAS middleware's most critical components because it aggregates results from distinct and heterogeneous providers [Ribeiro et al. 2018].

In the second experiment, we evaluate MIDAS middleware's correctness when receiving a query. For this, we submit 100 queries to MIDAS, randomly varying the requester. After that, we check the correctness of the responses based on the context of each requester. We do not limit the number of records returned. As expected, MIDAS 3.0 correctly selected 100% of contexts based on the requester and sent the results within the estimated deadline based on the requester's *when* element.

Finally, the third experiment evaluates the effort to implement MIDAS 3.0 with dynamic pragmatic information, such as intention and context. In this experiment, we provide a function point estimate that aims to estimate the effort to receive pragmatic information from online sources. We conducted this experiment with the collaboration of two MIDAS developers. As a result, MIDAS has an estimate of 140.97 function points. Currently, MIDAS 3.0 has 9,155 non-comment lines of code. We estimate MIDAS 3.0 with dynamic pragmatic information with 18,599 $(9,155 + 9,444)$ lines of code, where $9,444 = 67 * 140.97$ and 67 is the relationship between line of code in PHP[2] per function point. Although this work is an advance in state of the art, we understand that this analysis illustrates the complexity of providing pragmatic interoperability among clouds.

5. Conclusion

Interoperability is the ability of heterogeneous systems to exchange and to use mutually exchanged information. Interoperability is generally described into syntactic, semantic, and pragmatic levels. Pragmatic interoperability enables systems to affect one another's state and behavior so that the produced result matches the expected result. Our main efforts focused on investigating how to provide pragmatic interoperability among heterogeneous systems. Our model for pragmatic interoperability introduces the CAPITAL framework. For this, we investigated and provided (i) the data needed to provide pragmatic interoperability among systems, (ii) a consensual definition for pragmatic interoperability, and (iii) a conceptual framework to represent pragmatic interoperability.

The main contributions of this thesis are listed as follows: (i) a unified definition for pragmatic interoperability based on various definitions; (ii) mandatory elements to provide pragmatic interoperability; (iii) a conceptual framework capable of representing pragmatic interoperability among systems; and (iv) MIDAS architecture focused on pragmatic interoperability between SaaS and DaaS levels: MIDAS 3.0.

Our future research directions include: (i) converting canonical models into other formats; (ii) improving the capture of intention and contexts; (iii) implementing the CAPITAL framework in a real public security environment; and (iv) detailing more the crimes that may emphasize the need for other contextual elements.

[2]https://www.cs.helsinki.fi/u/taina/ohtu/fp.html

Acknowledgment

Elivaldo Lozer Fracalossi Ribeiro would like to thank FAPESB (Foundation for Research Support of the State of Bahia) for financial support (BOL0631/2016).

References

Armbrust, M. and et al. (2010). A View of Cloud Computing. *Commun. ACM*, 53(4):50–58.

Asuncion, C. H., Boldyreff, C., Islam, S., Leonard, M., and Thalheim, B. (2011). Pragmatic interoperability in the enterprise - A research agenda. In *23rd Conf. on Advanced Information Systems Engineering (CAiSE)*, London, UK. Springer.

Asuncion, C. H., Iacob, M., and van Sinderen, M. J. (2010). Towards a Flexible Service Integration through Separation of Business Rules. In *14th Int. Enterprise Distributed Object Computing Conf. (EDOC)*, pages 184–193, Vitoria, Brazil. IEEE.

Asuncion, C. H. and van Sinderen, M. J. (2010). Pragmatic Interoperability: A Systematic Review of Published Definitions. In *5th Int. Conf. Enterprise Architecture, Integration and Interoperability (EAI2N)*, pages 164–175, Brisbane, Australia. Springer.

Basili, V. and Rombach, H. (1988). The TAME Project: Towards Improvement-oriented Software Environments. *IEEE Transactions on Software Engineering*, 14(6):758–773.

Isoda, Y., Kurakake, S., and Imai, K. (2005). Context-Aware Computing System for Heterogeneous Applications. In *1st Int. Workshop on Personalized Context Modeling and Management for UbiComp Applications (ubiPCMM)*, pages 17–25, Tokyo, Japan. Springer.

Kolb, D. A. (1984). *Experiential Learning: Experience as the Source of Learning and Development*. Prentice Hall, 1 edition.

Lee, J., Lee, Y., Shah, S., and Geller, J. (2007). HIS-KCWater: Context-aware Geospatial Data and Service Integration. In *21st ACM Symposium on Applied Computing (SAC)*, pages 24–29, Seoul, Korea. ACM.

Liu, S., Li, W., and Liu, K. (2014). Pragmatic Oriented Data Interoperability for Smart Healthcare Information Systems. In *14th Int. Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pages 811–818, Chicago, USA. IEEE.

Maciel, R. S. P., David, J. M. N., Claro, D. B., and Braga, R. (2017). *Full Interoperability: Challenges and Opportunities for Future Information Systems*, chapter 9, pages 107–118. SBC.

Mane, B., Rocha, W. S., Ribeiro, E. L. F., Jesus, L. E. N., Motta, I. C., Lima, E., and Claro, D. B. (2020). Enhancing Semantic Interoperability on MIDAS with Similar DaaS Parameters. In *16th Brazilian Symposium on Information Systems (SBSI)*, pages 1–8, São Bernardo do Campo, Brazil. SBC.

Marinho, T., Cidreira, V., Claro, D. B., and Mane, B. (2016). MIDAS: A Middleware to Provide Interoperability Between SaaS and DaaS. In *12th Brazilian Symposium on Information Systems (SBSI)*, pages 401–408, Florianópolis, Brazil. SBC.

Mell, P. M. and Grance, T. (2011). The NIST Definition of Cloud Computing (SP 800-145). Technical report, NIST, Gaithersburg, USA.

Neiva, F. W., David, J. M. N., Braga, R., and Campos, F. (2016). Towards pragmatic interoperability to support collaboration: A systematic review and mapping of the literature. *Information and Software Technology*, 72(1):137–150.

Opara-Martins, J., Sahandi, R., and Tian, F. (2016). Critical analysis of vendor lock-in and its impact on cloud computing migration: a business perspective. *Journal of Cloud Computing*, 5(1):4.

Pokraev, S., Reichert, M. U., Steen, M., and Wieringa, R. J. (2005). Semantic and Pragmatic Interoperability: A Model for Understanding. In *17th Conf. on Advanced Information Systems Engineering (CAiSE)*, pages 377–382, Porto, Portugal. FEUP.

Reinsel, D., Gantz, J., and Rydning, J. (2018). The Digitization of the World from Edge to Core. Technical report, IDC, Framingham, USA.

Ribeiro, E. L. F. (2020). *Defining and Providing Pragmatic Interoperability - The MIDAS Middleware Case*. PhD thesis, Universidade Federal da Bahia, Salvador, Brazil.

Ribeiro, E. L. F., Monteiro, E. L., Claro, D. B., and Maciel, R. S. P. (2019a). A Conceptual Framework for Pragmatic Interoperability. In *15th Brazilian Symposium on Information Systems (SBSI)*, pages 1–8, Aracaju, Brazil. SBC.

Ribeiro, E. L. F., Vieira, M. A., Claro, D. B., and Silva, N. (2018). Transparent Interoperability Middleware between Data and Service Cloud Layers. In *8th Int. Conf. on Cloud Computing and Services Science (CLOSER)*, pages 148–157, Funchal, Portugal. SCITEPRESS.

Ribeiro, E. L. F., Vieira, M. A., Claro, D. B., and Silva, N. (2019b). *Interoperability between SaaS and Data Layers: Enhancing the MIDAS Middleware*, chapter 6, pages 102–125. Springer International Publishing.

Schreiner, G. A., Duarte, D., and dos Santos Mello, R. (2015). SQLtoKeyNoSQL: a layer for relational to key-based NoSQL database mapping. In *17th Int. Conf. on Information Integration and Web-based Applications & Services (iiWAS2019)*, Brussels, Belgium. ACM.

Spivey, J. M. (1989). *The Z Notation: A Reference Manual*. Prentice-Hall, Inc., 1 edition.

Tamani, E. and Evripidou, P. (2007). A Pragmatic Methodology to Web Service Discovery. In *4th Int. Conf. on Web Services (ICSW)*, pages 1168–1171, Salt Lake City, USA. IEEE.

Vieira, M. A., Ribeiro, E. L. F., Rocha, W. S., Mane, B., Claro, D. B., Oliveira, J. S., and Lima, E. (2017). Enhancing MIDAS Towards a Transparent Interoperability Between SaaS and DaaS. In *13th Brazilian Symposium on Information Systems (SBSI)*, pages 356–363, Lavras, Brazil. SBC.

Webster, C. (2014). From Syntactic & Semantic To Pragmatic Interoperability In Healthcare.

Wohlin, C., Runeson, P., Hst, M., Ohlsson, M. C., Regnell, B., and Wessln, A. (2012). *Experimentation in Software Engineering*. Springer, 1 edition.

Zhang, Z., Wu, C., and Cheung, D. W. (2013). A survey on cloud interoperability: taxonomies, standards, and practice. *ACM SIGMETRICS Performance Evaluation Review*, 40(4):13–22.