# Augmenting the Zachman Enterprise Architecture Framework with a Systemic Conceptualization

Alain Wegmann[1], Anders Kotsalainen[2], Lionel Matthey[3], Gil Regev[4], Alain Giannattasio[5]

*[1, 3, 4] Ecole Polytechnique Fédérale de Lausanne (EPFL), School of Communication and Computer Science, CH-1015 Lausanne, Switzerland*
*[2] Royal Institute of Technology (KTH), Section for Industrial Information and Control Systems, Osquldas väg 12, SE-100 44 Stockholm, Sweden*
*[5] Cambridge Technology Partners, CH- 1214 Geneva, Switzerland*
*{[1]alain.wegmann@epfl.ch, [2]anderskotsalainen@gmail.com, [3]lionel.matthey@gmail.com, [4]gil.regev@epfl.ch, [5]a.giannattasio@ctp-consulting.com}*

## Abstract

*The Zachman Framework offers a classification of the models created in an enterprise architecture project. These models form a holistic representation of the organization. Despite the prominent position of the Framework, there is little information publicly available to help designers create exact models that fit each other. In this paper, we propose a conceptualization based on General Systems Thinking. Our conceptualization provides concrete guidelines for creating the models required by the Framework. The proposed conceptualization establishes a better understanding of the models and of their relationships. This facilitates the creation and interpretation of the models. It also improves the traceability between them. We illustrate our approach with the results of a case study.*

## 1 Introduction

One of the seminal works in holistic enterprise and application design is the *Information System Architecture* (ISA) – also called the Zachman Framework. Zachman developed his Framework in 1987 [28]. This Framework is at the inception of the enterprise architecture discipline. Zachman was motivated to create his Framework by the prediction that technology would enable massive distribution of computing power and that doing so without some order would lead to chaos. Zachman's proposal was to maintain order by imposing a set of logical constructs for "defining and controlling the interfaces and the integration of all the components of the system" [28].

With ISA, Zachman wants to specify the documentation necessary to represent an organization and its IT. The goal is to provide a holistic view of the organization while segmenting this view into independent models. To structure the Framework, he uses the metaphor of an architect building a house. At the core of the Information System Architecture, there is the ISA matrix. This matrix contains 36 models organized in six rows and six columns. The rows describe the interests of stakeholders (e.g. *planner, owner, designer, contractor* and *sub-contractor*). The columns of the matrix contain responses to the basic questions: *what, how, where, who, when* and *why*. Each of these thirty-six models is related to the others. However, each one is also self-contained; this counterbalances the holistic aspect of the Framework.

ISA is a classification scheme, a taxonomy for organization architecture models [18]. ISA provides a synoptic view of the models needed for enterprise architecture. ISA does not define in detail what the models should contain, it does not enforce the modeling language used for each model, and it does not propose a method for creating these models.

In this paper, we address these issues by taking an epistemological approach in which we relate the ISA matrix's rows and columns with a conceptualization of the universe of discourse shared by the organization's stakeholders. We base our conceptualization on the General Systems Theory (GST) [27]; consequently, we represent the organization as a hierarchy of systems that span from business down to IT. With this

Presented at the 12th IEEE International EDOC Conference (EDOC 2008), München, Germany, September 15-19, 2008

conceptualization, we can propose specific guidelines for creating the models of the first five rows and four columns of the ISA matrix. We can also make explicit relationships between the cells. We developed this conceptualization as a part of the SEAM enterprise architecture method [22].

We illustrate the conceptualization and its application on ISA with a case study that we conducted, in an international firm, during a six-month project. In this case study, we redesigned an existing access control system, using SEAM as conceptualization and Zachman as representation.

The paper organization is as follows. In Section 2, we explain ISA. In Section 3, we present the conceptualization we base our work on and give an overview of SEAM. In Section 4, we show how to apply a systemic conceptualization to Zachman Framework, and we illustrate our results with a case study. In Section 5, we review the related work. In Section 6, we conclude and present ideas for future work.

## 2 The Zachman Framework

Zachman initially describes the Information System Architecture (ISA) in [28]. Zachman and Sowa extend and formalize ISA in [19]. The term "architecture" in "Information System Architecture" shows the analogy between "the construction of a computer system and the construction of a house" [19]. The analogy between traditional building architecture and IT systems architecture is central to ISA.

Zachman describes the process for house building as follows. In response to a future owner's initial and vague request to have a house built, the architect draws a *planner view* that roughly represents the main items that the house will include. This view serves as an initial agreement between the owner and the architect regarding what the owner wants. Next, the architect draws an *owner's view* that represents what the architect proposes to build. She draws her plans in a way that is understandable to the owner. The architect then draws the *designer's view* that constitutes the architect's plans in a form understandable by the architect and generally not by the owner. The architect's plans serve as the basis for negotiation with the general contractor who will build the building. The general contractor draws his or her own plans, the *contractor's view*, for negotiating with sub-contractors. Each sub-contractor draws his or her own plans for their specific purpose. The sub-contractors have their own plans. They are part of the *subcontractor's view*. The last view, *enterprise view*, is the building itself.

These *perspectives* [28] are complemented with *types of description* [28], i.e. the kind of questions that can be asked about a given view. In [28], Zachman prescribes three types of description: *data*, *function* and *network*. These types of descriptions are answers to the questions, what, how and where. These two orthogonal dimensions form a six by three ISA matrix in which the rows represent the six perspectives and the columns represent the three types of description (gray area in Figure 1). In [19], Zachman and Sowa add three more types of descriptions: *people*, *time* and *motivation*. They correspond to the questions, who, when and why. The result is the complete six by six ISA matrix as illustrated in Figure 1. This paper proposes a conceptualization for the cells marked "SEAM" in Figure 1.

| | what | how | where | who | when | why |
|---|---|---|---|---|---|---|
| | **Data** (column1) | **Function** (column 2) | **Network** (column 3) | **People** (column 4) | **Time** (column 5) | **Motivation** (column 6) |
| **Planner** (scope - row 1) | SEAM | SEAM | SEAM | SEAM | | |
| **Owner** (business model - row 2) | SEAM | SEAM | SEAM | SEAM | | |
| **Designer** (system model - row 3) | SEAM | SEAM | SEAM | SEAM | | |
| **Builder** (technology model - row 4) | SEAM | SEAM | SEAM | SEAM | | |
| **Sub-contractor** (detailed imp. - row 5) | SEAM | SEAM | SEAM | SEAM | | |
| (functioning enterprise - row 6) | | | | | | |

Figure 1: ISA matrix [19], in gray original matrix as proposed in [28].

Transposing the architecture metaphor to business and IT leads to the definition of the following perspectives:

- **planner row (scope) –** the big picture of the organization and the scope of the project.
- **owner row (business model)** – all aspects pertaining to the actual daily running of the business and how the IT systems will support it.
- **designer row (system model)** – the designs of the IT systems that fulfill the business needs of the owner.
- **builder row (technology model)** – the construction of the IT systems specified by the designer.
- **sub-contractor row (detailed implementations)** – the construction of the IT systems' components.
- **functioning enterprise row** – the actual data, processes, departments, employees, IT systems, applications etc. that make up the organization.

The columns of the ISA matrix contain the types of descriptions. The list below describes the cells in each column:

- **cells in data column** – the data or information relevant to the perspective (e.g. from the organization information in the planner row down to the database table in the sub-contractor row).
- **cells in process column** – the processes relevant to the perspective (e.g. from the business processes down to the internal processing of the IT systems).
- **cells in network column** – the networks relevant to the perspective (e.g. from the list of organizations down to nodes on the communication network).
- **cells in people column** – the people relevant to the perspective (e.g. from the customers down to the employees).
- **cells in time column** – time information relevant to the perspective.
- **cells in motivation column** – the motivational aspects relevant to the perspective.

Although ISA does not include a process, it does specify seven rules that designers should conform to when filling the cells [19].

1. "The columns have no order." If the columns were to have an order, it would imply that one dimension is more important than the other, when inherently they are all equally important.
2. "Each column has a simple, basic model." In the case of the data dimension, this is an entity-relation model; in the case of the function dimension, this is a process. The designer can choose different modeling techniques.
3. "The basic model of each column must be unique." Without this uniqueness, the ISA would not be as rigorous. Uncertainty over what belongs to which cell would arise.
4. "Each row represents a distinct, unique perspective." A distinct perspective is not a change in the level of details; it is instead the nature of what is represented that changes.
5. “Each cell is unique." This rule is a consequence of the rules 2 and 3. It contributes to making the ISA a useful classification scheme.
6. “The composite or integration of all cell models in one row constitutes a complete model from the perspective of that row." All cells in a row must be logically consistent with all the other cells in the same row.
7. "The logic is recursive." The designer can apply the Framework within each row. This means that the designer can analyze each row from the planner’s, owner’s, builder’s, etc. sub-perspectives.

In addition to the ISA matrix, Zachman defines additional matrices that we call intra-row matrices. They document the relations between different cells in a row (e.g. data-to-function, function-to-network, and data-to-network – all within a same row) [28]. These matrices are important for the design process. For example, if the designers want to check which functions are dependent on a specific data, they create the data to function matrix. The dependencies between data and function are then visible. Zachman does not propose matrices to relate the cells between rows (e.g. owner row’s data cell–to- data cell in designer row – all within a same column). Not having such matrices is problematic to the designers, as they cannot check the relations between the different rows of the ISA matrix.

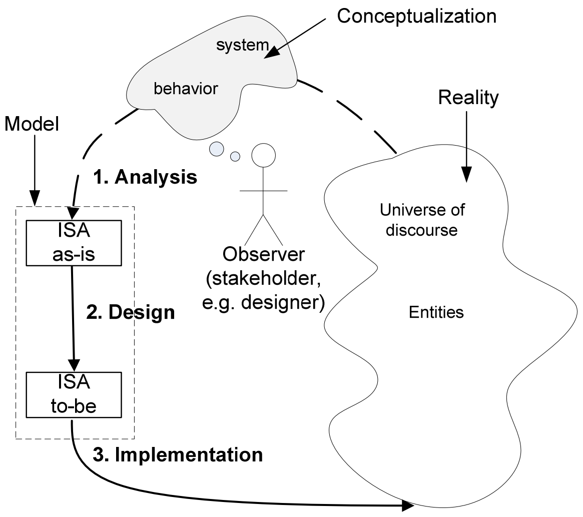


Figure 2: Description of the design process.

We can consider that an ISA-based development process has three broad phases (Figure 2). First, the designers analyze and represent the organization across the cells of the different rows and columns of the ISA matrix; they also define the intra-row matrices that relate the elements of the different cells within the rows of the ISA matrix. This model describes the *as-is* situation. Second, the designers analyze the existing situation and understand the problem to address. At this point, they design different possible solutions; each one described in a set of ISA and intra-row matrices. The designers compare these solutions and select one. The corresponding ISA and intra-row matrices describe the situation *to-be*, i.e. what the designers need to implement. Last, the designers implement the changes to the organization and to the IT system according to what the matrices describe.

## 3 Systemic Conceptualization

Designers face the problem of not knowing what to represent in each cell. It is important to understand what concretely represents each perspective. As we have seen above, the fourth rule on how to structure the contents of the ISA matrix states that crossing a row means changing perspectives, not just adding details. Concretely, what is the relation between the contents of the data cell in the designer row and the contents of the data cell in the builder row? In this section, we propose a way to conceptualize the reality that can guide the designers when filling the cells of the ISA matrix. We propose an approach based on General Systems Thinking (GST) [27]. GST provides a set of principles that modelers can apply to most domains of inquiry. GST draws attention to entities and their relationships, thus helping us to address the problems of understanding how to fill the cells and how to relate them. Our process is epistemological, i.e. we attempt to identify the relationships between the contents of the cells and the reality as observed by the stakeholders. We therefore propose a conceptualization of the universe of discourse. This conceptualization serves as a mediating structure between the perceived reality and the ISA matrix.

Our conceptualization is a part of the SEAM method. With SEAM, designers can analyze and design organizations and IT systems. GST [27] and RM-ODP [7] provides the foundations for SEAM. GST defines the principles necessary to understand how to model the reality. RM-ODP defines the modeling ontology [13, 14]. We have taught [25, 26] and have consulted [23] with SEAM since 2001. The theory underlying the conceptualization presented in this paper is in [17, 24].

In this section, we first describe how we conceptualize systems (Section 3.1), then processes and data (Section 3.2). We end with a discussion on the relations between the conceptualization and the matrices defined in ISA (Section 3.3).

### 3.1 Conceptualization of Systems

As a starting point of our conceptualization, we take the notion of system in its most general sense [28]. A system usually refers to any kind of entity in our environment, for example, an organization, an employee, an IT system, or an application. In GST, the common definition of a *system* is “a set of elements standing in interrelations” [21]. From an epistemological perspective, we have to understand where the set of elements and their interrelations come from [16]. This leads us to add the concept of *observer* as the person who observes entities in a universe of discourse (a part of some reality) and conceptualizes these entities as being systems or elements of a system. All stakeholders are observers. The designers, who need to carry out the project, are also observers. Figure 2 illustrates the relation, via the conceptualization, between the reality and the model.

During the project, the designers can choose to view the *systems as wholes* (black boxes) or *as composites* (white boxes). When they view a system as a whole, they ignore the system’s components. They focus then on the services offered by the system to its environment. When the designers view a system as a composite, the components and their relationships are visible. They can then understand the responsibility of each component and thus can specify them. Through this mechanism, the designers define a hierarchy of systems and components. We call this hierarchy the *organizational level hierarchy*. Each hierarchical level has its own language.

We illustrate these concepts with an example represented with the SEAM notation (Figure 3). The block arrows represent organizations, the stickmen represent people and the cubes represent the IT systems/applications. All are systems in the GST sense. In SEAM, we call *working object* the model element that represents what the designers conceptualize as a system. In this paper, the terms “system” and “working object” can be considered synonymous. Examples of working objects are `Employee_Mgt`, `HR_Dept`, `Employee`, and `HR_IT_System`. The designers can represent a working object as a whole or as a composite. For example, `Employee_Mgt [w]` is a working object as a whole. `Employee_Mgt [c]` is a working object as a composite. The dashed lines between the working objects in Figure 3 are whole/composite relationships. Figure 3 also illustrates that the designers might define non-trivial organizational hierarchies. This is a direct consequence of the influence of the observer, in the GST sense. For example, it is possible to have multiple parents for a same child. In Figure 3 (b), the `HR_Dept` and the `Security_Dept` are working objects as wholes. The designers consider them as two separate departments. In Figure 3 (c), the designers represent an ad-hoc department, called `HR_Dept_&_Security_Dept`

[c]. This department is the combination of HR_Dept [c] and Security_Dept [c]. We explain this unusual structure by two principles applied in the company: each department is responsible of the service it delivers; however, the implementation of these services is common to both departments. This illustrates the subjectivity of the enterprise model. The model can vary depending on what the designers wish to illustrate.

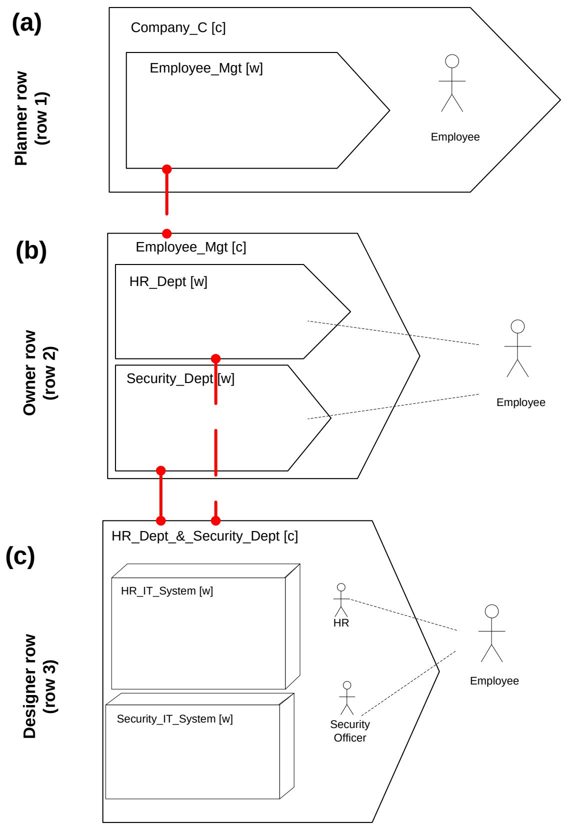


Figure 3: Organizational hierarchy (SEAM notation). The working object functionality is not represented.

In SEAM, we also represent interfaces between a system and its environment. For example, the Employee who is part of Company_C [c] in Figure 3 (a) is in the environment of Employee_Mgt in Figure 3 (b). The dashed lines make explicit that the HR_Dept and the Security_Dept are the systems in charge of the interface to the employee.

## 3.2 Conceptualization of Service Offered and of Service Implementation

Once the systems are defined, the designers can specify their behavior. Figure 4 illustrates how the designers represent the system functionality in SEAM. Our inspiration for behavior specification came from [4].

The functionality can be represented as the behavior of a system as a whole. For example, in Figure 4, HR_IT_System [w] executes HR_IT_Process_Data. We call this kind of behavior a localized action. A *localized action* modifies the state of the working object as a whole that hosts the action. A localized action represents a service offered by a working object.

The functionality can also be represented as the behavior within a working object as a composite. For example, HR_Dept_&_Security_Dept[c] performs Manage_Data. We call this second kind of behavior a distributed action. A *distributed action* is hosted in a working object as a composite and it modifies the state of one or more component working objects as wholes that participate in the action. A distributed action represents a service implemented by a working object.

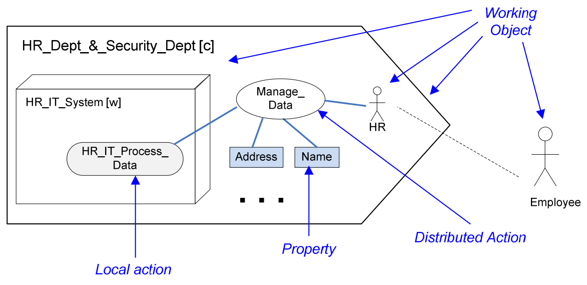


Figure 4: Working object functionality (SEAM notation). Only one organizational level is represented.

All these actions modify the state of properties of the working objects as wholes. For the sake of simplicity, we do not represent the properties of the working objects as wholes in this paper. When a distributed action is specified, shared property represents the information exchanged between working objects in the context of the distributed action. For example, in Figure 4, Name and Address are shared properties. This means that they are information exchanged between the working objects.

## 3.3 Mapping to the Zachman Framework

In most projects, the conceptualization of what people perceive is implicit. To make it explicit, the designers need to agree on the organizational level hierarchy that best represents the project. The designers typically organize a workshop for this purpose. In this workshop, they define the hierarchy of systems. We recommend starting the workshop by modeling the IT systems and then expanding from that point. Once the organizational levels are defined, it is possible to fill the network and the people columns. The artificial systems are listed in the network column. The humans are listed in the people column. Our conceptualization is also useful for defining what the modelers represent in the data and in the function columns of the ISA matrix. The data column describes the shared properties of the working objects s composite. The function column represents the distributed actions. Note that the ISA matrices do not represent the localized actions. We discuss in Section 4 how to represent the localized actions. Once the function, data, network and people columns are filled, we consider the ISA matrix complete.

After having filled the cells of the ISA matrix, the designers fill the matrices that define the relations between the cells. First, they specify the *intra-row matrices*. They relate within a same row: data-to-function, data-to-network/people, function-to-network/people. Note that we group network and people as we consider all these entities as systems. Thanks to our conceptualization, the designers can relate similar cells between rows. We call these matrices *inter-row matrices*. Zachman did not define these matrices. They relate, within a same column, the data, the function, the network and the people cells in adjacent rows. Our case study, in Section 4, illustrates how to fill and how to use all these matrices.

## 4 Application

In this section, we illustrate our conceptualization with the description of an enterprise model done in a concrete project (Section 4.1). We show how this conceptualization helps fill the ISA matrix (Section 4.2), the intra-row, and inter-row matrices (Section 4.3). We end this section by giving an idea about how to design an IT system using these matrices (Section 4.4).

### 4.1 Conceptualizing the Enterprise

The case study was conducted in a multinational firm we call `Company_C`. The company requires that the personnel, on-site, wear badges. Devices located at the doors scan the badges and control the access to the premises. To manage the badges and the employee related information, the company has two IT systems, one that manages the employee general information (e.g. address) and one that manages the access rights.

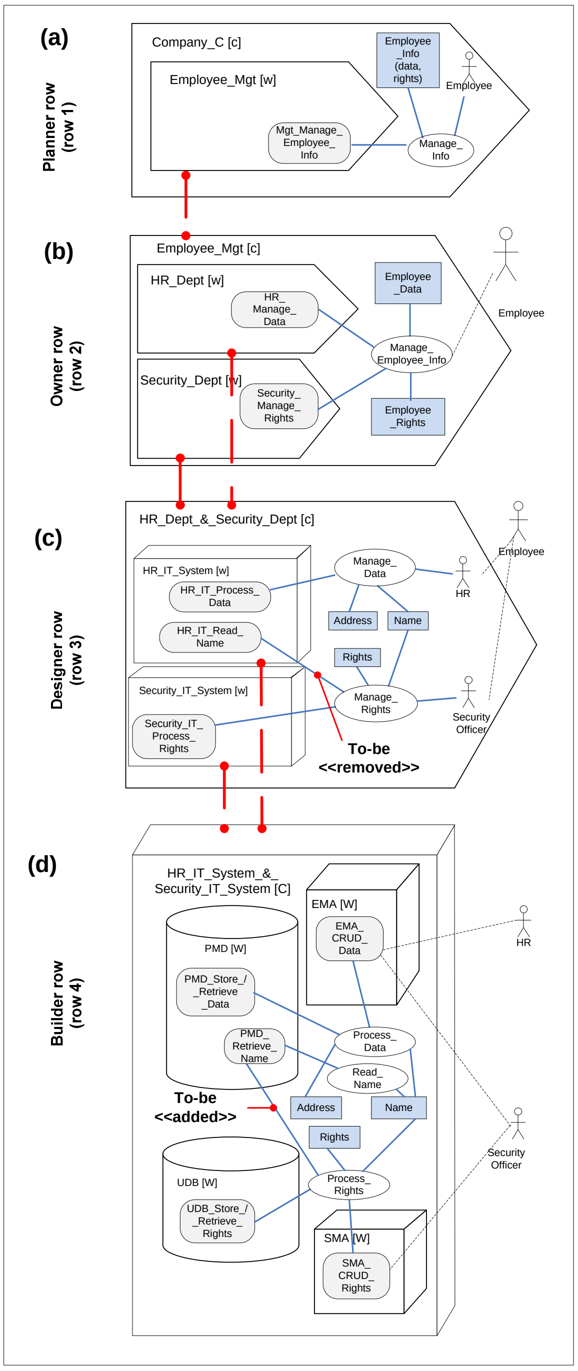


Figure 5: Description of the case (SEAM notation).

The `HR_IT_System` manages the information about the employees. It is composed of the People Master Data (`PMD`) database that stores this data and of the Employee Management Application(`EMA`) that manages these data. The `Security_IT_System` manages the employee's access rights. It is composed of the User Database (UDB) that stores which rights a specific employee has and of the Security_ Management_Application (SMA) that manages this information.

When an employee needs a badge, he or she asks the Security_Dept. The Security_Officer must retrieve the name of the employee from the HR_IT_System and feeds this data to the IT_Security_System that issues the badge. The designers need to remove the need to access the HR_IT_System. They use the Zachman Framework to analyze the existing situation and specify the solution to be developed.

From the business standpoint, the company considers that there is an Employee_Mgt organization. This organization provides all the services to the employee. The HR_Dept and the Security_Dept are part of this organization.

Before filling the ISA matrix, the designers need first to understand which systems they need to represent in the Framework. Then, they will be able to fill the matrix. To do so, they conceptualize the organization as a hierarchy of systems. Figure 5 illustrates this conceptualization.

The designers begin by modeling the designer row (Figure 5 (c)). This row includes the IT systems to modify. The goal is to understand the functionalities that these IT systems provide. In our case, HR_IT_System participates in the Manage_Data business process. The Security_IT_System participates in the Manage_Rights business process. The designers can then analyze the construction of the IT systems. To do so, they model the builder row (Figure 5 (d)). The applications PMD, UDB, SMA and EMA collaborate to provide the functionality defined in the previous row. In this paper, we do not present the sub-contractor row. It would show the architecture of the application and of the database. Zachman & Sowa call this the component model [19].

The designers need also to analyze the business aspects. They do so by modeling the owner row. In this row, they represent the services provided by the HR_Dept and by the Security_Dept (Figure 5 (b)). If wanted, they can also model the planner row. In that row, they only represent that the organization Employee_Mgt needs to manage the Employee (Figure 5 (a)).

The designers make the traceability between the rows explicit by using similar names: for example, Process_Data in HR_IT_System_&_ Security_IT_System (Figure 5 (d)) corresponds to HR_IT_Process_Data in HR_IT_System (Figure 5 (c)).

Figure 5 makes also explicit the project's goal. It shows, in Figure 5 (c), that the relation between Manage_Rights and HR_IT_System should be removed (i.e. the Security_Officer shall no longer use the HR_IT_System to issue a badge). This is possible if the designers involve the PMD application in Process_Rights. This requirement is represented in Figure 5 (d).

## 4.2 Filling the ISA Matrix

Once the conceptualization is completed, it is straightforward to fill the ISA matrix. Figure 6 illustrates the ISA matrix for our case study. With our conceptualization, we address only the first five rows and four columns. Note, we present only four rows in the case study. The designers use the same technique to fill the cells of the fifth row.

| **ISA** | Data | Function | Network | People |
|---|---|---|---|---|
| Planner Row (row 1) | - Employee _Info (data, rights) | - Manage_ Info | - Employee _Mgt [w] | - Employee |
| Owner Row (row 2) | - Employee _Data<br>- Employee _Rights | - Manage_ _Employee_ _Info | - HR _Dept [w]<br>- Security _Dept [w] | |
| Designer Row (row 3) | - Address<br>- Name<br>- Rights | - Manage _Data<br>- Manage _Rights | - HR_IT _System [W]<br>- Security_IT _System [W] | - HR<br>- Security _Officer |
| Builder Row (row 4) | - Address<br>- Name<br>- Rights | - Process _Data<br>- Read _Name<br>- Process _Rights | - EMA [W]<br>- PMD [W]<br>- SMA [W]<br>- UDB [W] | |

Figure 6: ISA matrix.

The network and people columns of the ISA matrix are filled based on the conceptualization as described in

Figure 5. The network column enumerates the business organizations and IT systems visible in Figure 5. The people column enumerates the employee. The data column represents the shared properties and the function column represents the distributed actions. The elements listed in the matrix in Figure 6 correspond to the graphical elements in Figure 5. For example, the element of the function cell of the planner row in Figure 6, i.e. `Manage_Info`, corresponds to the distributed actions, with the same name in Figure 5 (a), All cells of the ISA matrix (Figure 6) can be related, in a similar manner, to the graphical representation of the conceptualization (Figure 5).

## 4.3 Filling the Intra-Row and Inter-Row Matrices

The conceptualization gives the relations between the elements in the ISA cells. The designers represent these relations in additional matrices, called intra-row matrices and inter-row matrices.

(a)

| Row 3 Function<br>Row 3 Data | Manage _Data | Manage _Rights |
|---|---|---|
| Name | X | X |
| Address | X | |
| Rights | | X |

(b)

| Row 3 Network / People<br>Row 3 Data | HR_ IT_System | Security_ IT_System | HR | Security_ Officer |
|---|---|---|---|---|
| Name | X | X | X | X |
| Address | X | | X | |
| Rights | | X | | X |

(c)

| Row 3 Network / People<br>Row 3 Function | HR_ IT_System | Security_ IT_System | HR | Security _Officer |
|---|---|---|---|---|
| Manage_Data | X | | X | |
| Manage_Rights | ~~X~~ To be | X | | X |

Figure 7: Intra-row matrices of designer row.

We illustrate the three intra-row matrices: First, the designers represent the relations between the function and the data cells (Figure 7 (a)). These relations are visible in the Figure 5 (c); they correspond to the relations between the distributed actions and the shared properties. An "X" between a function and a data indicates that the function accesses or manipulates the data. To be more specific, it is possible to write one or more letters of the CRUD acronym. CRUD means Create, Read, Update and Delete: the different kinds of processing done on the data.

Then, the designers represent the relations between the data cells and the network/people cells (Figure 7 (b)). These relations make explicit which working objects know which data.

Last, the designers represent the relations between the function cells and the network/people cells (Figure 7 (c)). These relations are visible in Figure 5 (c). They make explicit which working objects participate in which distributed action.

In summary, the intra-row matrices, with the proposed conceptualization, make explicit the relations between the distributed actions (function cells), the properties (data cells) and the working objects (network/people cells). Note that we merge the contents of the network and of the people systems as we consider all the entities referenced by these cells as systems.

(a)

| Row 2 Data<br>Row 3 Data | Employee _Data (HR_Dept) | Employee _Rights (Security _Dept) |
|---|---|---|
| Name | X | X |
| Address | X | |
| Rights | | X |

(b)

| Row 2 Function<br>Row 3 Function | Manage_ Employee_ Info (HR_Dept) | Manage_ Employee_ Info (Security _Dept) |
|---|---|---|
| Manage_Data | X | |
| Manage_Rights | | X |

(c)

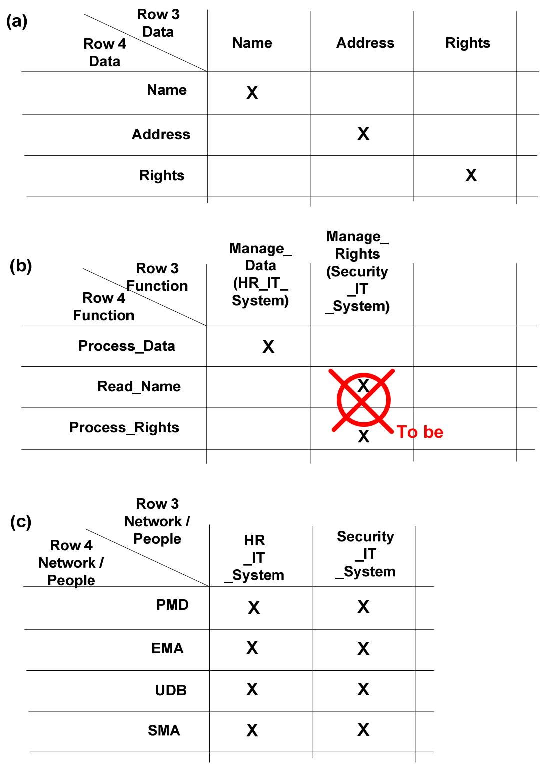
| Row 2 Network/ People<br>Row 3 Network/ People | HR_Dept | Secuity _Dept |
|---|---|---|
| HR_IT_System | X | X |
| Security_IT_System | X | X |
| HR | X | X |
| Security_Officer | X | X |

Figure 8: Inter-row matrices between owner and designer rows.

Thanks to our conceptualization, it is also possible to define the relations between rows. These are the inter-row matrices. Figure 8 shows examples of inter-row matrices. With them, the designers can relate cells of the same kind between consecutive rows: e.g., owner row's data cell-to-data cell in designer row. The relations between adjacent cells require indirections.

We illustrate this with the function column. In fact, it is the localized actions of a system (seen as a whole in a given row) that correspond to the distributed actions of the same system (seen as a composite in the following row). Unfortunately, the localized actions are not visible in the ISA matrix. We need to overcome this limitation. We do this by replacing the localized action by the corresponding distributed action and the system name that participates in the distributed action. For example, it is the localized action `HR_Manage_Data` in `HR Dept` [w] (Figure 5 (b)) that corresponds to the distributed action `Manage_Data` in `HR_Dept_&_Security_Dept` [c] (Figure 5 (c)). As the localized action `HR_Manage_Data` is not represented in the ISA matrix, we refer to the localized action by stating that the system `HR Dept` [w] participates to the distributed action `Manage_Employee_Info` (Figure 5 (b)). This explains why the function-to-function inter-row matrix (Figure 8 (b)) relates `Manage_Employee_Info` (`HR Dept` [w] ) and `Manage_Data`.

The originality of the inter-row matrices is their capability to relate adjacent cells between consecutive rows. This makes explicit the traceability between the rows. This feature does not exist in the approach proposed by Zachman.



(a)

| Row 3 Data / Row 4 Data | Name | Address | Rights |
|---|---|---|---|
| Name | X | | |
| Address | | X | |
| Rights | | | X |

(b)

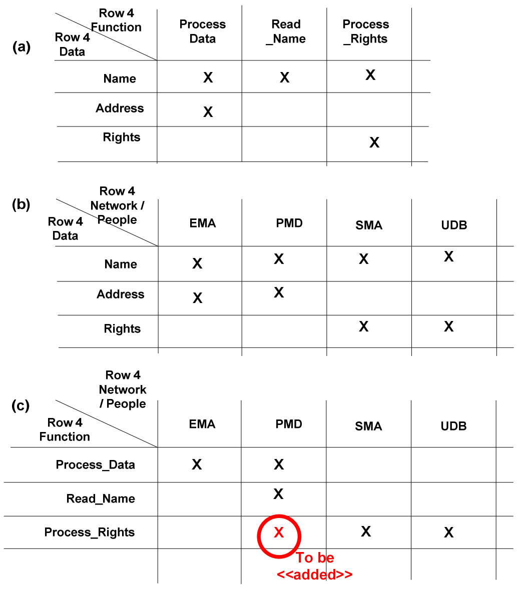
| Row 3 Function / Row 4 Function | Manage_ Data (HR_IT_ System) | Manage_ Rights (Security _IT _System) | |
|---|---|---|---|
| Process_Data | X | | |
| Read_Name | | ~~X~~ | |
| Process_Rights | | X To be | |

(c)

| Row 3 Network / People / Row 4 Network / People | HR _IT _System | Security _IT _System |
|---|---|---|
| PMD | X | X |
| EMA | X | X |
| UDB | X | X |
| SMA | X | X |

Figure 9: Inter-row matrices between designer and builder rows.

The designers can apply the techniques presented in this paper in all rows. Figure 7 represents the intra-row matrices of the designer row. Figure 10 represents the intra-row matrices of the builder row. The designers can describe the owner row and the planner row with the same technique. Figure 8 represents the inter-row matrices between the owner and the designer rows. Figure 9 represents those between the designer and the builder rows. The designers can relate the cells of the planner and the owner rows with the same technique.



(a)

| Row 4 Function / Row 4 Data | Process Data | Read _Name | Process _Rights |
|---|---|---|---|
| Name | X | X | X |
| Address | X | | |
| Rights | | | X |

(b)

| Row 4 Network / People / Row 4 Data | EMA | PMD | SMA | UDB |
|---|---|---|---|---|
| Name | X | X | X | X |
| Address | X | X | | |
| Rights | | | X | X |

(c)

| Row 4 Network / People / Row 4 Function | EMA | PMD | SMA | UDB |
|---|---|---|---|---|
| Process_Data | X | X | | |
| Read_Name | | X | | |
| Process_Rights | | X To be <<added>> | X | X |

Figure 10: Intra-row matrices of builder row.

## 4.4 Designing the New System

Once the designers have filled the matrices, the designing of the new organization and of the new IT systems can begin. We will investigate the design techniques in our future work. However, we can already provide an idea on how the designers can proceed. In the design phase, the designers manipulate the matrices (as described in BSP [29]: method that inspired Zachman). In our example, the goal is to eliminate the need to access two IT systems for the `Security_Officer` when she issues a badge. The designers can realize this by automating the access to the `HR_IT_System` when issuing the badge. The proposed redesign is visible in the to-be annotations in Figure 7, 9 and 10.

First, the designers make explicit that issuing a badge (action `Manage_Rights`) should not require the participation of the HR_IT_System. This is visible in Figure 7. The relation between the HR_IT_System and the distributed action Manage_Rights is eliminated.

Then, using the inter-row matrix designer row's function–to-function in builder row (Figure 9 (b)), the designers identify that the actions Read_Name and Process_Rights are related to Manage_Rights. The designers make explicit that the Read_Name distributed action (builder row) is not anymore necessary to support the Manage_Rights process (designer row). Figure 9 makes this explicit with the crossed X between the Read_Name and Manage_Rights functions.

Last, the designers specify the need to automatically access to the PMD application when the Process_Rights distributed action executes. Figure 10 (c) makes this explicit with the additional X at the intersection between the Process_Rights function and the PMD application.

## 5 Related Work

Authors (such as [20]) propose processes for using the Zachman Framework. But, they do not propose a conceptualization. Only very few publications provide a conceptualization. [6] shows that different conceptualizations of a situation can co-exist: different units within a company produce different ISAs given that they have a different perception of the reality. They propose to add a Z-axis to ISA and to create an ISA matrix for each point of view. However, the authors do not provide concrete guidelines on how to accomplish the merging of the different matrices. [5] provides practical instructions for creating the models for some of the ISA cells. These instructions consist of mainly following the meta-models proposed for each column in [28], such as entity-relationship diagrams for the data column and process diagrams for the function column. [15] proposes a method for working with ISA matrices. This method specifies the order to fill the cells based on the dependencies between them. The method also introduces the concept of anchor cells; cells that are critical to complete before going on to the other cells in the row. Although this method gives an order in which to treat the cell, it does not specify the content of the cells.

Other authors have developed new frameworks, inspired or not by Zachman: for example TOGAF [12], E2AF [8] and Urba-CIGREF [11]. These frameworks do not propose a conceptualization as the one described in this paper.

We base our conceptualization on SEAM. Other methods exist to graphically model systems. Examples of these methods are Archimate [9], Demo [2], OPM [3], and SSM [1]. The originality of SEAM is the combination of GST and of RM-ODP. This combination provides both philosophical and formal foundations to the approach. None of the above methods have both aspects.

## 6 Conclusions and Future Work

Enterprise architecture, with its holistic view, can be one of the main enablers of the alignment between business and IT. The most prominent and holistic of EA framework, the Zachman Framework, provides a canvas for the models that the designers create when working on an enterprise model. Unfortunately, Zachman does not specify how to create these models. In this publication, we use an epistemological process to augment ISA with a conceptualization of the universe of discourse. We have shown how this conceptualization contributes to ISA by making explicit what should be in the ISA matrix, the intra-row and inter-row matrices. We have illustrated this contribution with a case study that served as a running example.

ISA and SEAM might appear to address the same problem. Our experience shows that they are complementary. SEAM is useful for understanding what to model and how to graphically represent a hierarchy of systems that span from business down to IT. SEAM, however, cannot represent large systems, as the diagrams would become too complex. Here ISA brings value, as it is possible to represent large systems with lists. Therefore, we can consider the Zachman Framework as a taxonomy useful to access a multitude of smaller SEAM models. This is consistent with the definition of ISA given by Sessions [18]. We will explore the SEAM / Zachman integration as our future work.

## 7 References

[1] Checkland, P. and Scholes, J., Soft System Methodology in Action, Wiley, 1990.
[2] Dietz, J., "Basic notions regarding business processes and supporting information systems", Proceedings of CAiSE 2004 workshops, Riga, Latvia, 2004.
[3] Dori D, Object Process Methodology, Springer Verlag, 2002.
[4] D'Souza D. Wills. A. C., Objects, Components and Frameworks with UML: the Catalysis Approach, Addison Wesley, 1999.

[5] Finkelstein, C., Enterprise Architecture for Integration: Rapid Delivery Methods and Technologies, Artech House, 2006.
[6] Garner, B.J., and Raban, R., “Context management in modeling information systems (IS)”, in Information & Software Technology 41(14): 957-961, 1999.
[7] ISO/IEC 10746-1, 2, 3, 4 | ITU-T Recommendation, X.901, X.902, X.903, X.904, “Reference Model of Open Distributed Processing”, 1995-1996.
[8] Institute for Enterprise Architecture Developments, Extended Enterprise Architecture Framework, http://www.enterprise-architecture.info, accessed April 2008.
[9] Lankhorst M. et al., Enterprise Architecture at Work, Springer Verlag, 2005.
[10] Lê, L.S. and Wegmann A., “SeamCAD: Object-Oriented Modeling Tool for Hierarchical Systems in Enterprise Architecture”, Proceedings of the 39h IEEE Hawaii International Conference on System Sciences (HICSS), Hawai, 2006.
[11] Longépé, C., Le projet d’urbanisation des S.I., Dunod, 2006.
[12] The Open Group Architecture Framework (TOGAF), http://www.opengroup.org/togaf, accessed June 2008.
[13] Naumenko, A., Triune Continuum Paradigm: a paradigm for General System Modeling and its applications for UML and RM-ODP, Ph.D thesis number 2581, Swiss Federal Institute of Technology - Lausanne. EPFL, June 2002.
[14] Naumenko, A., and Wegmann. A. "Formalization of the RM-ODP foundations based on the Triune Continuum Paradigm", in Computer Standards & Interfaces, 29(1):39-53, Elsevier, 2007.
[15] Pereira, C. M., Sousa P., “A Method to Define an Enterprise Architecture using the Zachman Framework”, Proceedings of the Symposium on Applied Computing, 2004.
[16] Regev, G., and Wegmann, A., “Where do Goals Come From: the Underlying Principles of Goal-Oriented Requirements Engineering”, Proceedings of the 13th IEEE International Requirements Engineering Conference (RE’05), Paris, France, 2005.
[17] Rychkova, I., Regev, G., Wegmann, A., “High Level Design and Analysis of Business Processes. Advantages of declarative specifications.”, Proceedings of RCIS 2008, Marrakech, Morocco, 2008.
[18] Sessions, R., A Comparison of the Top Four Enterprise Architecture Methodologies, URL: http://www.objectwatch.com/whitepapers/4EAComparison.pdf, accessed November 2007.
[19] Sowa, J. F., “Zachman, J.A., Extending and formalizing the framework for information systems architecture”, in IBM Systems Journal, 31(3):590-616, 1992.
[20] Spewak, H. S., Enterprise Architecture Planning Developing a Blueprint for Data, Applications and Technology, QED Publishing Group, 1992.
[21] von Bertalanffy, L., General System Theory, George Braziller, 1968.
[22] Wegmann, A., “On the Systemic Enterprise Architecture Methodology (SEAM)”, Proceedings of the International Conference on Enterprise Information Systems (ICEIS), Angers, France, 2003.
[23] Wegmann, A., Regev, G. and Loison, B.. “Business and IT Alignment with SEAM”. Proceedings of the 1st International Workshop on Requirements Engineering for Business Need, and IT Alignment, Paris, 2005.
[24] Wegmann, A., Lê, L. S., Regev, G., and Wood, B., "Enterprise Modeling Using the Foundation Concepts of the RM- ODP ISO/ITU Standard," in Information Systems and E-Business Management, 5(4):397-413, 2007.
[25] Wegmann A.; Regev G.; de la Cruz J. D., Lê L.S., and Rychkova I, “Teaching Enterprise and Service-Oriented Architecture in Practice”, Proceedings of the Workshop on Trends in Enterprise Architecture Research (TEAR 2007), 2007.
[26] Wegmann, A., Julia, P., Regev, R., Perroud, O., and Rychkova I., “Early Requirements and Business-IT Alignment with SEAM for Business”, Proceedings of the 15th IEEE International Requirements Engineering Conference (RE’07) Dehli, India, October 2007.
[27] Weinberg, G. M., An Introduction to General Systems Thinking, Wiley & Sons. New York, 1975.
[28] Zachman, J. A., “Framework for Information Systems Architecture”, in IBM Systems Journal, 26(3):276-292, 1987.
[29] Zachman, J.A., "The Zachman Framework and Observations on Methodologies," Business Rules Journal, 5(11), 2004, URL: http://www.BRCommunity.com/a2004/b206.html, accessed December 2007.