Cite this material as:

Chapter 3

Virtual Institutions

Every day in the real world we participate in a number of institutions. Once we enter a workplace, shop or university we realize the change of the context and start obeying the rules of the environment we have entered. Our behavior is highly influenced by these rules, which range from not strictly enforceable and rather implicit social conventions (like etiquette) to more explicit and usually strictly controlled rules or instructional norms (like walking through a metal detector in an airport or queuing before paying for the products in a shop).

Virtual Worlds represent a replication of the real world with a difference that most of them allow some degree of anonymity, have no physical adherence to the culture of a particular region and provide richer facilities to observe and control the behavior of participants. To avoid anarchy, that usually comes together with anonymity, Virtual Worlds require clear ways to define and enforce the rules of the interactions. Moreover, in virtual societies as established in Virtual Worlds it is highly desirable to be able to easily explain the institutional rules to the carriers of different cultural backgrounds and representatives of different language groups. The most efficient way to achieve these goals is to have the institutional rules expressed in a formal way (using mathematical equations) and create natural language translations for each of the desired languages.

The concept of Virtual Institutions proposed in this chapter helps in bringing structuring of interactions into virtual spaces by formalizing the interaction rules. Although in the current form it is only concerned with establishing explicit rules it can also be applied for establishing implicit social conventions. The proposed approach is not only useful for helping humans to establish social order in Virtual Worlds but also for helping autonomous agents to reduce the uncertainty about the world, understand and learn the rules of the interactions and interpret the actions of the others. Further we present the concept in more details, illustrate it with an example and outline the implementation requirements.
3.1 The Concept

The concept of Virtual Institutions is defined as follows.

**Definition:** Virtual Institutions are 3D Virtual Worlds with normative regulation of interactions.

More precisely, we propose to separate the development of Virtual Worlds based on the concept of Virtual Institutions into two independent phases: specification of the interaction rules and design of the 3D Interaction environment. For producing more efficient designs such separation is widely used in architecture [130], whose metaphor inspires Virtual Worlds. Apart from design efficiency, in our case this separation has the following advantages:

- it helps in achieving the clear distribution of the development tasks between system analysts and designers;

- explicitly focusing the attention of system analysts on the interactions forces them to inspect the system in details before creating the visualization, which is useful for detecting the critical points and errors at an early stage;

- it makes the specification of the interaction rules independent of particular Virtual Worlds technology used for the visualization of the system, permitting a quick and easy portability to new visualization platforms.

To be able to support the conceptual separation between the design of a Virtual World and normative control of the interactions within this space we subdivide the Virtual Institutions concept into two conceptual layers: Visual Interaction Layer and Normative Control Layer.

The Visual Interaction Layer maps to the domain of 3D Virtual Worlds. It is concerned with audio and visual aspects of the multimedia, as well as with visualization of the interactions of participants in the 3D Virtual World.

The Normative Control Layer maps to the domain of Electronic Institutions and is concerned with the institutional control of interactions that happen in the Visual Interaction Layer.

For the purpose of normative control of the interactions we suggest employing the Electronic Institutions methodology, which is supplied with facilities for rules specification and tools for helping to ensure the validity of the specified rules and their correct execution. However, we would like to point out that in Virtual Institutions these rules are used in a slightly different manner. In contrast to the majority of normative Multiagent
Systems (and Electronic Institutions in particular) the normative part of a Virtual Institution does not represent all the activities that are allowed to be performed in a Virtual World. On the contrary, the normative part can be seen as defining what is prohibited to do and if not said then everything is allowed.

Not every Virtual World requires normative control of interactions as well as not every real world institution needs 3D Visualization. Only systems that have a high degree of interactions, and only if these interactions need to be structured in order to avoid violations) may need institutional modeling. And, only the institutions where 3D visualization of active components is possible and beneficial should be visualized in Virtual Worlds.

Systems that could benefit from both interaction control and 3D visualization, provided by the concept of Virtual Institutions, should be built following the Virtual Institutions metaphor presented in the next section.

### 3.2 The Metaphor

The concept of Virtual Institutions requires the development of a new metaphorical basis.

Choosing an appropriate metaphor for a new concept is very important and is a critical factor in users’ overall acceptance of this concept [176]. Metaphors provide useful abstractions for technological concepts and help in explaining the users something unfamiliar (and usually complicated) in terms of something they are well familiar with [126].

The original metaphor behind the concept of Electronic Institutions is a “theater”. As in a theater, Electronic Institutions have a set of scenes, and only the “players” with specific roles can appear in particular scenes. The conversations inside each of the scenes follow a strict protocol [167]. Although metaphorically they originate from this limited domain, Electronic Institutions have much higher expressive power and are applicable to much wider range of problems than the original theater metaphor suggests. Moreover, combining Electronic Institutions with 3D Virtual Worlds makes the theater metaphor rather inappropriate.

Virtual Worlds do not only suggest a metaphorical explanation of the concept, but also provide a concept visualization in terms of the proposed metaphor. Visualizing activities like E-Commerce in the theater metaphor could cause more confusion to the users than bring actual benefits. Therefore, we see a strong need to use another more general metaphor for Virtual Institutions.

While choosing a new metaphor for software it is important to select amongst those whose meaning is known to very general audiences and doesn’t change in different cultures. Building on the metaphors users deal with very often, and especially on the metaphors reflecting physical structures, increases the chance of the metaphor being as-
Virtual Worlds are inspired by the metaphor of architecture. They often employ physical structures like buildings, rooms, walls, etc. to represent different kinds of activities and to separate them from one another [176]. The metaphor of architecture is universal as humans are mostly familiar with the concept of a building. It is also culturally independent. Therefore, the metaphor of architecture seems like a good choice for Virtual Institutions and can be applied to the same (or an even wider) range of problems that we are concerned with in this thesis. However, the fact that we are dealing not with arbitrary Virtual Worlds, but with Virtual Worlds that support normative regulation of interactions, requires us to use a constrained version of the metaphor used by the Virtual Worlds.

We propose seeing the concept of Virtual Institutions as a virtual space called 3D Interaction Space. This space can correspond to an arbitrary 3D Virtual World populated by avatars and various objects. Inside the 3D Interaction Space a set of buildings are located, where each of the buildings represents an Electronic Institution. The appearance of the 3D Interaction Space outside the buildings can be arbitrary and the behavior of avatars is not controlled by the institutional norms. In contrast, there are restrictions on appearance and interactions whilst inside the buildings.

As it is hard to provide a physical metaphor for the 3D Interaction space we find it necessary to introduce some sort of substitute. Thus, we present the metaphor of a garden. A garden should be seen as a place surrounding the institutional buildings. This metaphor is well known to humans as gardens often surround residential buildings in the real world. In this way the 3D Interaction Space can be described as the combination of the garden and institutional buildings.

Employing the building metaphor for the visualization of an institution is motivated by the fact that many institutions familiar to the participants in the real world (like universities, courts, banks etc.) also have a brick and mortar representation. As in the real world the walls of a virtual building create visible boundaries for norm enforcement.

Each institutional building in our metaphor is associated with its unique set of interaction rules, which are controlled by the specification of the corresponding Electronic Institution. The participants are visualized as avatars and each of them is assigned with at least one role. Only participants with specific roles can enter the institutional buildings and once there should act according to the specification of the corresponding institution. The concept of a role is widely used in Virtual Worlds. In many game based Virtual Worlds a role reflects the fact of being a part of a selected group and determines different abilities of the participants associated with it. In non-game based Virtual Worlds the role is normally used to distinguish between fee paying subscribers and participants with trial membership.

Further elaborating the metaphor, we see each of the institutional buildings being
divided into a set of rooms, which are separated from each other by walls and doors. The doors are opened or closed for a participant depending on the role and the institutional state. Again, this choice of metaphors is fully consistent with real world institutions. Walls and doors are often employed to restrict access to some activities.

Figure 3.1 outlines the brief idea behind the Virtual Institutions metaphor presented so far.

With the help of the 3D Virtual Worlds technology the metaphor of Virtual Institutions can be visualized. All of the concepts contributing to the metaphor (Garden, Buildings, etc.) will have a virtual representation inside the resulting 3D Virtual World.

The findings in the area of Virtual Worlds design research suggest that during the construction of a 3D representation of a Virtual World, it is important to keep the benefits of traditional 2D interface design in mind [27]. Participating in a 3D environment, where users can manipulate 3D objects, does not necessarily mean exclusion of 2D interface elements. In fact, the interaction with 2D interface elements offers a number of advantages over a 3D representation for particular tasks [142]. Most efficient selection techniques, for instance, are widely realized in 2D, whereas, the selection process in a 3D user interface must consider the user’s viewpoint and distance to the object. Combining the advantages of 2D and 3D design is a very powerful and intuitive approach for the construction of virtual environments.

Based on these findings we suggest enhancing the Virtual Institutions metaphor with 2 additional components (which in the resulting Virtual World will be visualized as additional 2-dimensional elements). The first component is a map of the 3D Interactions
space, which inside a building is transformed into a building map. The map metaphor is widely used in all kinds of Virtual Worlds to improve the navigation of participants. Virtual Worlds can be harder to navigate than the real worlds, and employment of maps proved to be very useful in Virtual Worlds [54]. As Virtual Institutions are visualized as Virtual Worlds, we find it necessary to introduce, maps into Virtual Institutions as well.

Another additional component that contributes to the overall metaphor and which is also visualized in a 2-dimensional way is what we call “backpack with obligations”. Electronic Institutions provide technological facilities for participants to collect obligations while acting inside the institution. Not fulfilling the obligation might lead to access restriction to some activities by the participant. To inform the participants about the reasons for imposed restrictions and to provide them with easy facilities to visualize their commitments (so that enforcement of restrictions can be promptly avoided) we introduce the backpack metaphor. This metaphor is taken from military-oriented games, in which it is usually the case, that a player is given a mission, which in turn consists of a set of submissions. Any time during the game, the player can click on his/her backpack to see the mission related details. In a similar way, the backpack can be used to show the acquired obligations to the participants.

Next we describe the components of each of the Virtual Institutions conceptual layers in terms of the new metaphor.

### 3.3 Visual Interaction Layer

Visualization of the Virtual World and its participants as well as providing the participants with interaction facilities is associated with the following set of concepts.

#### 3.3.1 3D Interaction Space

The 3D Interaction Space is the entrance point into the visualized 3D Virtual World. It is represented as a graphical 3-dimensional area and associated with an Euclidean 3-dimensional coordinate system, where (0, 0, 0) is the reference point for locating the objects. Most of the laws of physics usually apply within this space, however, their effect may be slightly different to the corresponding effect in the real world and some of the rules may be absent or purposely allowed to be violated.

All the interactions of the participants are limited to interacting within this space. There is no possibility to move beyond it and the only way to leave is by disconnecting from the Virtual World. Once someone enters it, he/she will become embodied as an avatar and will be physically located inside.
To enhance the believability of the visualization the space is usually populated with a number of various 3D Objects. The most typical case is that a 3D Interaction Space is decorated with grass, trees, bushes, cars, etc.

The special type of objects located inside the 3D Interaction Space are buildings. Each of the buildings is metaphorically seen as an institution. It is important for every building to collide with the participants trying to enter it and there should be no way to switch the collision off. Most of the rendering engines utilized by 3D Virtual Worlds usually support forcing the collision, and for those which do not it is required to introduce the necessary changes to force the collision before using them for Virtual Institutions.

Anywhere outside the institutional buildings the interactions of participating avatars are not regulated and every event that happens inside this space is immediately visualized without any prior validation. This part of the 3D Interaction Space is called the Garden.

### 3.3.2 Institutional Buildings

Institutions are excluded from the uncontrolled interactions. Each institution is seen as a separate normative structure, and for its visualization we use the “building” metaphor. The 3D model of a building for every institution is present in the system located within the 3D Interaction Space and for regulating the interactions each building is associated with an Electronic Institution specification. The Electronic Institution is seen as an infrastructure that establishes a set of norms on the behavior of participants, who can be either humans or autonomous agents. The institutional buildings do not necessarily follow an Euclidean model. The rooms inside the building may not be involved in any kind of spatial relationship and the total size of the building as perceived from the outside, may be significantly different to the total size of the internal space.

The enforcement of the rules is achieved via strict institutional control of the actions by participants inside the institutional buildings. Every event that a participant requests by pressing keys on the keyboard or by operating the mouse are first sent to the institutional infrastructure for validation. If the institution permits the event execution then the corresponding action is visualized, otherwise the event is ignored. It is also possible for the institution to provide context-based explanations for the reasons why a particular event cannot be processed. This can be done by the institution requesting an action to be visualized for a desired participant. This action may be just a sound which is played when a participant is trying to open a door but has no permission to enter it, or it may be a more sophisticated action, such as an employee of the institution appearing in front of the participant and explaining the reasons for the imposed restrictions.

Each of the institutional buildings has a single entrance door, through which the participants can enter it. The entrance to each institution is restricted and only participants...
with specific roles will be accepted to the institutional building. Whether or not a particular avatar is allowed to enter a given institution is determined by matching the role initially assigned to the corresponding participant against the set of possible roles accepted by the Electronic Institution, responsible for regulating the interactions inside the given building. The initial roles are assigned before entry to the 3D Interaction Space and can be further changed during their interactions inside the institutional buildings. The avatars with roles that are not accepted in a given institution will experience the locked entrance door. For the avatars with a role accepted by the institution the entrance door will be unlocked or open.

One of the most typical cases is that after entering the 3D Interaction space all of the participants are initially given the “Guest” role and most of the institutional buildings accept the participants with this role, letting them enter the registration room. In this room, the participants may decide to explore the building further, but to do this, they are required to enter their registration details (login/password) and only then, will be assigned a new role to be able to enter other rooms.

### 3.3.3 Avatars

The participants of the 3D Interaction Space are visualized as avatars. We distinguish between the following two types of avatars: avatars for visitors and avatars for the institutional employees. The visitors are all agents or human playing external roles. Internal agents and governors are institutional employees.

Visitors’ avatars are provided with an initial set of default appearances, but these can be changed later. The institutional employees are assumed to have a similar appearance which is in line with the dress code of the institution that they are employed with. The governors are usually not displayed, but they may appear as avatars when a participant tries to violate the institutional rules. Governors may also be visualized in case a participant experiences navigation problems and requires help (the trajectory of the avatar’s movement may clearly indicate whether the participant is lost in the Virtual World).

According to [46], following an embodied agent may dramatically improve the navigation. For this reason if the participant appears to experience navigational problems (to determine this, the trajectory of each avatar is constantly observed) the embodied governor may appear embodied as an avatar to guide the participant to a desired destination. The default appearance of the governor is a police officer dressed in uniform.

We assume that some of the participants of Virtual Institutions are autonomous software agents (not humans). Such agents are visualized as avatars and may replace humans playing any institutional role and try to believably act on their behalf. The fact that they

---

1 An electronic representation of one’s self in a form of a graphical character.
have similar embodiment as human participants and, therefore, may be mistakenly be-
lied to be humans raises an important question: whether to explicitly notify the humans
that they are talking to an autonomous agent or hide this fact? On the one hand, hiding
this fact has a number of practical benefits connected with improving the acceptance and
the development of trust in the actions of such avatars [184]. On the other hand, such
approach poses a number of ethical questions. Can software agents influence human rela-
tions [140]? Can humans develop a false sense of trust towards agents and can the agents
abuse this trust [19]? Can humans develop personal relationships with agents mistakenly
accepting them as humans and what are the consequences of this [56]?

We do not have a preferred position in the debate regarding the need to notify hu-
mans that they communicate with an agent and believe that the questions raised above
require a detailed further investigation. Therefore, we propose that the system architect
should be solely responsible for making the final decision on this topic. As an overall
guideline, however, we propose to pay attention to the ethical issues connected with the
presence of agent-driven avatars and introduce the following mechanism for indication
that the avatar is controlled by an autonomous agent. Each time an autonomous agent
gains control over an avatar, this avatar is forced to change its appearance to the standard
“bot” appearance, which will become an indication for other participants that this avatar
is not controlled by a human anymore. The only exception to this is the appearance of the
institutional employees, which should not be changed. The institutional employees are
institutional representatives and the institution takes full responsibility for their actions,
so they are not likely to express deviant behavior and will not provide misleading infor-
mation to other participants. Some of the institutional employees may act, for example,
as sales assistants and behind each of the assistants humans and autonomous agents may
constantly interchange each other following the implicit training approach presented in
Chapter 5. Changing the appearance of the sales assistant avatar on a regular basis may
become highly frustrating for the participants interacting with this avatar and we suggest
to keep their appearance unchanged.

Another important issue concerning avatars is that in some of the rooms it is allowed
by the institution to split the participant into several alteroids\(^2\) (avatars). The concept
of alteroids is important as in the virtual space we are not physically restricted to being
just one actor. This fact received a lot of attention in computer games and other kinds of
software. In order to support this feature in Virtual Institutions, each time a new alteroid
is created a human participant should decide which avatar to choose to control further
and the remaining avatars will be controlled by autonomous agents. This functionality
allows a human to employ autonomous agents for performing some routine tasks on a

---
\(^2\)We borrow this metaphor from computer games, where a player is often in control of several avatars simultaneously.
human’s behalf, while the human may be involved in other (more complex) activities.

Finally, it is necessary to impose restrictions on the avatar movement inside the 3D Interaction Space. These restrictions are associated with the role played by the corresponding avatar. For being assigned with a role before entering the 3D Interaction Space the participants are first required to type in their identification details. This normally includes entering the unique nickname and a password that the system can map to an acceptable role. After the identification process, the corresponding avatar appears in a specified location within the 3D Interaction Space, outside any of the institutional buildings. While outside, the avatars are free to execute any possible actions and their communication is not moderated by any of the institutions. Entering the institutional buildings imposes all the limitations associated with the chosen role.

3.3.4 Rooms

Every institutional building consists of a set of rooms. The rooms are supposed to be represented as rectangular boxes closed by walls from every side. Each room has at least one door, through which it can be entered. The collision inside each of the rooms is forced and cannot be switched off. Once a participant enters a room, the only possibility for the corresponding avatar to move outside the room boundaries is to walk through one of the doors embedded into the walls of the room. For a participant to be able to instantly move from one room to another, it is necessary for those two rooms to be connected via a door. This connection is not required to be spatial. A door may simply act as a teleport, instantly moving an avatar to the initial position in the room that is being entered through the door.

Unlike in most of the typical Virtual Worlds, in Virtual Institutions it is not always the case that a participant can enter the room through a door and then will be able to exit through the same door. On the contrary, when the room is entered, the entrance door may become automatically locked for the user. This depends on the specification of the underlying Electronic Institution, whether it allows the backward movement between the rooms or not. In order to provide a consistent experience for the participants we suggest that the system designers should include the backward movement where possible.

3.3.5 Doors

The Doors are used to connect different rooms in the institutional building. Each door is associated with a number of execution states (a state per avatar). Each state logically explains whether a door is open or closed for a particular avatar. Due to this fact, at the same point of time, the same door may be open for one participant, but closed for another
one. In order to avoid breaking the immersion of participants by the fact that some of them will be observing the others walking through closed doors, we use the following approach. Every time an avatar changes the state of a door by opening it – every other avatar which is currently observing the door will also see that the door is open but will not necessarily be able to enter it. Technically this is achieved by making the door invisible when it is open and only switching off the collision for the avatars that are allowed to proceed through it.

3.3.6 Map

In order to simplify the navigation of the participants, every institution is supplied with a map of the building. The map usually appears in the upper-right corner of the screen as a semitransparent schematic plan. Each of the available rooms are shown on the map and the human-like figures show every participant the positions of all the associated alteroids. While moving throughout the institution the positions are updated accordingly.

The human individual can use the map to find the desired alteroid and may click on the corresponding figure to take control of it. Once an alteroid is selected the human fully controls it and the alteroid that the human has previously controlled is assigned to an autonomous agent attached to the selected alteroid.

3.3.7 Backpack with obligations

While acting in an institution a participant may acquire some commitments. An example of such a commitment may be that a participant who has just won the auction will not be able to directly leave the institution, but is committed to visit the payment room before leaving. These commitments are expressed in the specification of the underlying Electronic Institution and their fulfillment is fully controlled by the system. In order to have a simple way to present those commitments to a human we use the metaphor of a backpack utilized in many computer games. The backpack icon is usually displayed in the lower right part of the screen and a participant may decide to hide it, or reveal it. Clicking on the backpack will result in the participant being presented with the textual list of commitments acquired so far.

3.3.8 Events/Actions/Messages

Although we anticipate that the participants may use all sorts of different devices for navigating virtual worlds, in a standard case a participant in a 3D Interaction Space is able to control the avatar and change the state of the Virtual World by pressing keyboard buttons, moving a mouse or clicking mouse buttons. These physical actions executed by a human
in the real world generate events inside the Virtual World, which are then visualized as actions executed within the 3D Interaction Space. The events that a participant is trying to execute inside an institutional building are not directly visualized. Before visualization every event is transformed into a message understandable by the institution and sent to the institutional infrastructure for validation. Only if the message is consistent with the current state of the institution and it is not against the institutional rules to visualize the corresponding action, the action is then performed.

3.4 Normative Control Layer

The description of the Interaction Layer provided the explanation of the Virtual Institutions metaphor in terms of the Virtual Worlds domain. In order to explain how the institutional control of the processes inside the Virtual World is achieved we next present the mapping of the concepts presented above to the domain of Electronic Institutions. This mapping serves 2 conceptual purposes:

- explaining the Electronic Institutions metaphor in terms of Virtual Worlds using the concepts familiar to most of the humans. We see this part as being useful for providing richer explanation of the Electronic Institutions to people without Multiagent Systems background.

- explaining the Virtual Worlds in terms of institutional specification of the underlying processes. This part is useful for the designers who will be able to see the direct connection between the designs and the activities that these designs are thought to support.

Further in this section we provide an explanation how each of the concepts described in the previous section is expressed in terms of Electronic Institutions.

Table 3.1 presents a short overview of the mapping between the concepts of Electronic Institutions and the corresponding concepts of the Virtual World.

3.4.1 Federations

The part of the 3D Interaction Space called Garden, in its current form, has no conceptual relationship to the Electronic Institutions metaphor, although the developers of the Electronic Institutions are working on introducing the notion of federations as the unity of institutions that serve a similar purpose and, therefore, the 3D Interactions Space can be seen as the Federation. A good real world example for a federation would be a university. Most of the universities usually have a campus (federation), which contains a
Table 3.1: Mapping between 3D Virtual Worlds and Electronic Institutions

<table>
<thead>
<tr>
<th>Specification Element</th>
<th>Example</th>
<th>Virtual World Element</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene</td>
<td>[Image]</td>
<td>Room</td>
<td>[Image]</td>
</tr>
<tr>
<td>Transition</td>
<td>![Image]</td>
<td>Room/No representation</td>
<td>![Image]</td>
</tr>
<tr>
<td>Connection</td>
<td>![Image]</td>
<td>Door</td>
<td>![Image]</td>
</tr>
<tr>
<td>Number of participants</td>
<td>![Image]</td>
<td>Size of the room</td>
<td>![Image]</td>
</tr>
<tr>
<td>Agent</td>
<td>![Image]</td>
<td>Avatar</td>
<td>![Image]</td>
</tr>
<tr>
<td>Message</td>
<td>![Image]</td>
<td>Action</td>
<td>![Image]</td>
</tr>
<tr>
<td>Root Scene</td>
<td>![Image]</td>
<td>Room/No representation</td>
<td>![Image]</td>
</tr>
<tr>
<td>Exit Scene</td>
<td>![Image]</td>
<td>Garden</td>
<td>![Image]</td>
</tr>
<tr>
<td>Obligations</td>
<td>![Image]</td>
<td>Backpack</td>
<td>![Image]</td>
</tr>
<tr>
<td>Performative Structure</td>
<td>![Image]</td>
<td>Map</td>
<td>![Image]</td>
</tr>
<tr>
<td>Data Types in Ontology</td>
<td>![Image]</td>
<td>3D Objects or No Representation</td>
<td>![Image]</td>
</tr>
</tbody>
</table>
number of faculties (institutions) united under the common purpose of education and un-
der the same umbrella institution. In the case of the university the institutional rules are
very much the same across different faculties and the rules and policies of the university
are mentioned much more frequently than the rules of the particular faculty.

3.4.2 Institution

In Virtual Institutions every building is seen as an institution. This metaphor is borrowed
from the real world, where most of the institutions are brick and mortar. Entering or
exiting such an institution changes the behavior of participants and their conversation
style. Most of the processes inside each of the brick and mortar institutions are controlled
by the institutional infrastructure.

3.4.3 Scenes and Transitions

A scene in Electronic Institutions is seen as the protocol that describes a basic activity.
Giving it a brick and mortar representation to this concept we suggest seeing each of
such basic activities as a separate physical room inside the institutional building. As in a
theater there is usually a different set of decorations for different scenes – the same way
we see it being represented in Virtual Worlds. The only difference is that at the current
stage each of the rooms in Virtual Institutions are physically separated by a set of walls
from all other rooms.

Transitions in Electronic Institutions serve the purpose of a middle point between
two different rooms and sometimes are used for synchronization of agents. In a case
when a transition is used for synchronization purposes we suggest to visualize it as a
room of a special kind. Probably the most appropriate appearance for such a room would
be a waiting room similar to the waiting rooms in the airports. That’s the place where
participants will have to wait for someone else to join them if the institution demands so.

In the most general case, when a transition is used as a simple middle point between
two rooms we propose not to visualize it at all, to avoid forcing the participants to move
through the spaces that do not correspond to any behavioral process.

In the Electronic Institutions specification for each of the scenes the number of par-
ticipants for each of the accepted roles is always specified. We use this information to
determine the size of the room. The more avatars can enter the scene the bigger it should
be. So, the size of the room is proportional to the sum of the maximum number of
participants in all the accepted roles.

Electronic Institutions support having multiple executions of a scene. This may, for
example, be useful for the case when the maximum number of participants in a scene has
been reached and a new participant wants to enter the scene. A classic solution to this problem is to create another instance of the scene and place the new participant inside of this new scene instance. For dealing with the issue of multiple scene executions in Virtual Worlds we use the metaphor of a floor. Every time it is required to create another instance of an already existing scene – another instance of the room that corresponds to this scene is created, placed on top of the previous instance and the building is extended to have an additional floor. So, the height of the buildings that contain scenes with multiple execution will be dynamically changing. There are cases when a participant is allowed to choose which scene execution to enter. In order to support entering a different execution of a scene in the 3D Virtual World, transitions function as elevators, which allow for choosing the room to be entered in terms of pressing the corresponding button.

### 3.4.4 Performative Structure

The original idea behind the Performative Structure is to define the main activities and specify the role flow of participants between those activities (scenes). The closest concept present in the Virtual Worlds domain is the map of the building. Although it is not usually the case that the map contains information on who is allowed to enter which room, this information becomes unnecessary after personalizing the interface. As it naturally happens in the Virtual Worlds, the map for each of the participants is personalized. The participants are usually only interested in the places that they can access and have much less interest in the rest. This allows us to personalize the map in a way that only the scenes that a participant is able to access are visualized there.

Every performative structure contains special types of scenes called “root” and “exit”. These scenes are not associated with any processes inside the institution and cannot have a protocol specified for them. The root scene only serves the purpose of being the entrance point into the institution. In some cases when a root scene is connected to more than one transition it is useful to visualize it as a small room with a set of doors. Otherwise, when there is only one transition connected to it we suggest to avoid creating unnecessary rooms and do not visualize it. The entrance door of the institution in this case will lead straight into the room connected to the root scene via a first transition.

The exit scene defines the exit point. It is never visualized. Reaching the exit scene means leaving the institutional building. So, once a participant walks through a door that corresponds to the connection leading to the exit scene, the corresponding avatar will be located outside the institutional building and the exit door will be closed (as there can be no return connection from the exit scene).
3.4.5 Connections

The connections in the Performative Structure graph are used to determine the flow of participants between a scene and a transition. In the Virtual World we see them visualized as doors connecting different rooms. Many connections should not be visualized at all. For example, when in the specification there are several incoming connections that define entering a scene by the agent with different roles there is no need to create a separate door inside the corresponding rooms for each of them. In such a case only one door will be present and all of the agents will use the same door for entering the room.

In the case when a transition is not visualized all the connections leading to it should not be visualized too. In such situation the rooms that are connected with each other via this transition will have a direct door connection.

3.4.6 Obligations

As it was already mentioned before, the obligations the participants acquire in an institution are represented as the backpack, which on opening displays the textual description of the obligations.

3.4.7 Data Types in Ontology

To be able for the agents operating in an Electronic Institution to “understand” each other it is necessary for them to establish common language. Therefore, the ontology defines a set of data types that the agents should operate with (by sending messages to each other). The majority of the Virtual Worlds inhabitants are humans, who usually are pretty good in establishing a common language without any ontological help. But it doesn’t mean that the ontology is absolutely useless for visualization. The data types in the ontology are used to describe objects that the participants operate while interacting inside the institution. Although some data types may refer to intangible concepts, the majority of them should be represented as 3-dimensional objects.

3.4.8 Illocutions and Messages

The notion of illocutions originates from the speech act theory [180]. It is suggested that every time a human speaks there is always some intended action behind each uttered illocution. For example, in an extreme case such as the a president of the United States declaring war on Iraq - the result is not just a set of words spoken out by someone, but a set of actions that change the state of the world around us. That’s why in Electronic Institutions every message sent between two agents contains an illocution. Consider the
example of a classroom with an open door. Here two possible illocutions wrapped around
the same message can be understood in a very different way:

- **Illocution 1:** Request(Student, closed(door))
- **Illocution 2:** Inform(T eacher, closed(door))

Illocution 1 has the clear expectation on the side of the agent (a student) who receives
it to perform the action of closing the door, while Illocution is probably not associated
with any action. Notice that it is possible to have a simple dialogue using those two
illocutions, where a teacher executes Illocution 1 and the student replies with Illocution
2. Being concerned with visualization of the illocutions we have a set of actions for
every message, where the illocution used in the dialogue defines which action of this set
to visualize.

### 3.4.9 Synchronization Issues

The Electronic Institution specification permits the modeling of situations where several
agents have to proceed to the next scene together. To do that, agents must synchronize.
While synchronization is easily supported by the Electronic Institution infrastructure, it
requires detailed thinking when applied to Virtual Worlds.

We want to provide an immersive environment where humans are “driving” their
avatars throughout different rooms. Moving to another room in a Virtual World is ex-
pressed by opening a door, walking through the transition and opening the entrance door
of the next room. When more than one participant is required to leave the room (open the
exit door) we propose the following approach: each participant approaches the door and
tries to open it (showing the intention to leave). The last agent to synchronize approaches
the door and is able to open it. When the door is opened all the waiting agents will be
moved through it automatically (synchronized).

### 3.5 Concept Illustration: Trading Institution

In order to illustrate the concept of Virtual Institutions we propose the reader to consider
the following scenario.

**Scenario:** Imagine a businessman who is very interested in contemporary art. He is a
regular customer of a Virtual Institution and uses its fish market auction for buying and
selling fish. One of the rooms in this institution serves as the gallery for graffiti posters.
During the vernissage the artist is present in the room and is looking forward to conversations with visitors. The businessman enters the poster exhibition and spends his time browsing through the art works, while his another alteroid, driven by an autonomous agent, participates in the fish market auction and buys fish on his behalf.

In Virtual Institutions the above scenario can be mapped onto the norms of the trading institution presented in section 2.5.4. The specification of the Trading Institution forms a basis for the Normative Control Layer. The Visual Interaction Layer of this institution corresponds to the 3D Virtual World outlined in Figure 3.2.

The Virtual World contains a landscaped garden, within which the institutional building is located. The building consists of five rooms, three of which correspond to Registration, Meeting and Trading scenes in the Electronic Institution specification. The other two rooms represent the transitions connecting these scenes in the specification. The rooms are separated by doors, where doors are either locked or open for a participant depending on its role and the state of the institution.

The Virtual World for this institution is automatically generated from the Electronic Institution specification. In the Visual Interaction Layer the Electronic Institution specification determines the skeleton of the Virtual World. In the Normative Control Layer the specification serves as the basis for the execution of the infrastructure, enforcing the in-
terational constraints, controlling the state of conversations, and providing permissions for different roles.

The performative structure forms the basis for the map of the institution and the 3D model of the institutional building. Each of the scenes (“Registration Room”, “Meeting Room” and “Trade Room” in this case) lay the foundations for the generation of the 3D models of the rooms. The size of each room is determined by the maximal number of participants specified inside each scene. The root and exit scenes do not have any visual representation. Transitions are transformed into special types of rooms (corridors) connecting the scenes. Connections (lines connecting the elements of the performative structure) are represented as doors. Each door is initially locked, and will be opened as soon as the participant is granted the permission to enter the corresponding scene or transition by the institution.

After the automatic generation the rooms in the newly created Virtual Institution are not furnished, however, the doors, transitions and door labels are present. This institution is fully functional, which means that all the security issues of the institution will be imposed (e.g. permissions, protocols, obligations). The agents are able to freely interact and take part in conversations; the consistency of those conversations and interactions with the institutional rules is guaranteed by the infrastructure and the possibility to split into alteroids is also granted. To make the institutional building visually appealing the textures and additional objects are added at a later stage.

Each participant enters the system appearing as an avatar in the garden. The initial role given to each of the participants is “Guest”. As defined in the specification any “Guest” can enter the institution. Therefore, the entrance door of the institutional building is always open for all avatars. Entering the institutional building for an avatar means entering the Registration Room. The “root” scene and the transition connecting it to the “Registration” scene are not visualized. After successful registration in the Registration Room the participant’s role changes and, depending on this new role, appropriate doors are open. Opening of the doors happens independently for each particular avatar, and the same doors may be locked for a different avatar if the permission is not granted.

Once inside the institutional building a participant moves throughout its different rooms. The Registration Room serves as a reception desk. Once the desk is approached a participant is welcomed into the institution by the Receptionist agent and asked to enter login and password details to be able to proceed further. Figure 3.3 demonstrates the Registration Room.

From the Registration Room every participant can either leave the institutional building by walking back through the entrance door into the garden or enter the corridor (transition) behind the registration room, which connects it to the Meeting Room. The door that connects the corridor with the Meeting Room is initially locked. Only when
the user is correctly identified as a “Buyer” this door will be unlocked. If the user is identified as a “Seller” the door will not be open as Sellers are not permitted to enter the Meeting Room. The exit door from the corridor in this case will contain a number of buttons on it (as in an elevator). Each button is marked with the name of the auction currently conducted inside the Trading Room. Pressing a button results in teleporting the Seller into the selected instance of the Trading Room.

Figure 3.3: Registration Room Inside the Trading Institution

The view inside the Meeting Room is presented in Figure 3.4. To match the scenario presented above the room is decorated as a graffiti poster gallery. The posters are presented in a conventional way (hanging on the walls) as it is usually done in real world galleries. The participants of the gallery are represented as avatars. The avatar controlled by an autonomous agent has a robot-like appearance. All the other avatars are human participants with customizable appearances.

Additionally to the main 3D part, the Virtual World is enhanced with some 2-dimensional elements. The chat window serves the purpose of the communicator between participants. To reduce the information overload and engage people into spatial interactions the maximum audibility distance is specified. Only the avatars within the audibility distance can participate in a conversation. The avatars that are not within this distance can not “hear” the conversation. Due to this fact, the female avatar present in the left part of the window is not disturbed with the conversation between the artist and the businessman. This approach also provides means to address privacy issues: humans can clearly observe the participants of each conversation and may not give away secure information
if there are undesirable participants present.

Another important 2D part of the interface is the map of the institution. It is only visible if the mouse pointer is moved to the right border of the screen. The large rectangular blocks represent rooms and the smaller ones correspond to transitions. The solid figure with the arrow on top of it displays current location of the human driven avatar within the institution. The non-solid figures represent the autonomous agents that are in the participant’s subordination. As it was already mentioned, Electronic Institutions permit situations where a participant can split himself/herself into a number of alteroids. Only one of the alteroids can be controlled by a human and all the others are controlled by autonomous agents. Autonomous agents act autonomously trying to fulfill the task specified by a human. They may move around and even walk between different rooms. If in some situation an autonomous agent is unable to proceed with the given task due to the lack of intelligence, the figure representing this agent on the map starts to blink attracting the attention of the human.

The human is able to control any of the alteroids at any time by clicking the corresponding figure on the map. This will lead to switching to a different view (determined by the position and head rotation of the alteroid). The control over the avatar that the human was controlling before is automatically passed on to an autonomous agent and the appearance of this avatar is changed to the default appearance of an autonomous agent (robot look). In the given scenario the autonomous agent (non-solid figure) represents the businessman in the fish market auction, while the businessman drives the avatar (solid figure) through the graffiti auction room.
The Backpack with obligations is another 2-dimensional element of the user interface. It helps the human to remember the obligations towards the institution that have to be fulfilled. The backpack automatically opens and the pending obligation is displayed if the situation of not fulfilling an obligation makes it impossible to proceed to another scene or state in the institution. The participant can also see the obligations on demand by clicking on the backpack icon.

Figure 3.5: Trading Room Inside the Trading Institution

As it is expressed in the institutional specification, the Trading Room is allowed to have multiple executions – it is represented as a number of similar rooms placed on top of each other. Inside each of these rooms a different type of auction is conducted. One such auction is outlined in Figure 3.5. This room instance is currently functioning as a Fish Auction. A seller is conducting the auction with lobsters being the current offer. The buyers surrounded the seller’s desk and wait for an acceptable price. The red robot on the left hand side is the businessman’s autonomous agent from our scenario.

Once the offer is announced, the big screen behind the seller is updated to show the picture of the current product, the remaining time left for bidding and current price. As specified in ISLANDER the auction follows the Dutch auction protocol. In this protocol the auction is held for a given period of time and the initial price for a good is set to be slightly higher that the desired price of the seller. Within the given time-frame for a given auction round the price decreases to the minimal acceptable price. As soon as the price drops to the point when one of the buyers is ready to accept it – the buyer raises his hand to notify the auctioneer that he wants to purchase the advertised product. The auctioneer immediately announces the winner and if there are still products left to be
sold – then continues with the next round. After a successful purchase the buyer who
won the previous round is required to approach the seller and finalize the purchase. The
obligation to pay for the good is assigned to the buyer by the institution and can be
observed through the buyer’s backpack. Until this obligation is fulfilled this buyer will
be unable to leave the institution. The product that was not sold during the given time
frame is withdrawn from the auction.

3.6 Formalizing the Concept using Z Specification
Language

In the previous section we described the concept of Virtual Institutions. Here we for-
malize the concept using Z specification language (see [123] for details). The main task
behind the formalization is to create precise requirements for implementation of Virtual
Institutions and their deployment. For providing such requirements, each of the con-
cepts for both Visual Interaction and Normative Control layers is formalized and some
additional components are introduced.

Z is a formal specification language that is widely used for creating precise specifica-
tion of computer systems. It utilizes standard mathematical notations used in set theory,
lambda calculus, and first-order predicate logic. The main reason for selecting Z for Vir-
tual Institutions is that Z notation used in this language is commonly preferred by many
researchers and the language itself is quite successful in communicating general ideas
about software architectures from researchers to developers. Formal Z specifications us-
ing mathematical notations help to precisely describe the properties which an information
system must have, without unduly constraining the way in which these properties are to
be achieved [190]. Z specification helps to describe what the system must do without
saying how it is to be done. This abstraction makes formal specifications useful in the
process of developing a computer system, because they allow questions about what the
system does to be answered confidently, without the need to disentangle the information
from a mass of detailed program code, or to speculate about the meaning of phrases in
an imprecisely-worded prose description [190].

As the result of applying this technique, the Z-specification of Virtual Institutions is
quite general and independent of a particular technology. Despite that generality, it pro-
vides enough information for system developers to be able to gain overall understanding
of the computer system and implement it.

In general, a Z specification of a system consists of the following steps [190, 123]:

• Defining the terms, constants and variables. Here constants, global variables and
  compositional types used throughout the specification are defined. The variables
here should be seen as mathematical data types rather than variables seen in pro-
gramming languages.

- Defining the basic schemas. Z schemas are small pieces of specification in math-
ematical form linked together with text commentaries. They help to decompose
the formalization of the system into basic components. In our case, each of the
schemas corresponds to a basic concept from either the Normative Control Layer
or Visual Interaction Layers.

- Defining the runtime state. This dimension of the formalization defines how the
schemas defined on the previous step should evolve at runtime.

- Defining the state initialization mechanisms. Here the initialization procedures for
each of the schemas are described.

- Defining the operations allowed at runtime. The schemas represent the key archi-
tectural components and the operations define possible actions that can change the
state of the system.

All of the above steps were followed to create the Z-specification of Virtual Institu-
tions. Before presenting a detailed overview of these steps, we will outline the key
components that were identified on each of the conceptual levels. Figure 3.6 shows the
two layers and each of them includes the logical components that correspond to this
layer. The Normative Control Layer contains only one specification element EISpec
as the top level abstraction of the institutional rules. The reason for this abstraction is that
the formalization of the Normative Control Layer was completed by the developers of
Electronic Institutions and, therefore, we do not present it here. Interested readers will
find the specification of the EISpec in Appendix A.

The Visual Interaction Layer consists of 3D World - corresponding to 3D Interaction
Space; 3DBuilding - the institutional building; Garden - the part of the 3D Interaction
space surrounding the institutional buildings; Map - the schematic representation of an
institutional building; Room - a room inside an institutional building; Door - a door
connecting two rooms; Obj - an object located anywhere inside the 3D Interaction Space;
Action/Message Table - an abstraction required to achieve mapping between the actions
inside an institutional building and corresponding messages sent inside the Normative
Control Layer. Action represents any change of the 3D Interaction Space that forces the
change in the Normative Control Layer, while Event corresponds to a request to conduct
an action inside an institutional building.

The extended version of this diagram is presented in Figure 3.7. There the relations-
ships between the presented components are outlined and the basic Z-schemas used in
the formalization are introduced. Each of the schemas are represented by a rectangle separated into 3 sections.

1. the name of the schema;
2. the names of other schemas that a given schema depends on;
3. the operations defined for the schema.

Readers familiar with object oriented programming should consider an explanation of the specification in terms of the class metaphor. The name of the schema can be seen as a name of the class. The names of the dependent schemas correspond to member variables. The operations can be treated as member functions. The connecting arrows presented in the picture represent the inheritance, and dotted lines show the use of one class by another.

Next we present the details of each of the Z specification steps.
3.6.1 Virtual Institution Terms

Variables

We use different sets of variables for different classes of elements that we wish to represent.

$$[\text{RoomVar, DoorVar, AvatarVar, ObjectVar, TimeVar, ActionVar, EventVar}]$$

Constants

The constants are the identifiers associated with avatars, rooms, doors, objects and natural numbers relating to time and a constant to specify the minimum avatar space in a room.

Figure 3.7: Schema Relationship Diagram for Z-Specification.
Chapter 4

Approach and Methodology

The implementation requirements expressed in the previous chapter provide the ground for the implementation of Virtual Institutions. They specify the components required for both Visual Interaction and Normative Control layers as well as describe how the system should be deployed.

In this chapter we demonstrate the proof of Virtual Institutions concept by presenting our proposal for implementation.

We start with introducing the Virtual Institutions Methodology, which is quite generic and can be applied to the development of any Virtual Institution. Next we present the technical details behind some of the steps of this methodology and demonstrate how existing technologies can assist in completing them. Towards the end of the chapter we demonstrate the generic architecture that should be used for deployment of Virtual Institutions and outline the technological solution that was used.

4.1 Virtual Institutions Methodology

For building Virtual Institutions we propose using Virtual Institutions methodology outlined in Figure 4.1. This methodology covers the whole development process and is also supplied with the tools for deployment of Virtual Institutions. This methodology should be employed by system architects and software developers (both are further called users). In general, applying Virtual Institutions methodology requires 7 steps to be accomplished:


2. Specification of an Electronic Institution.

3. Verification of the specification.

4. Automatic Generation of the corresponding 3D environment (if needed).
5. Annotation of the Electronic Institution specification with components of the 3D Virtual World.

6. Integrating the 3D Virtual World into the institutional infrastructure.

7. Enabling Implicit Training

Figure 4.1: Methodology steps.

Completing the above steps will result in defining both Normative Control Layer and Visual Interaction Layer of the corresponding Virtual Institution. As shown in Figure 4.1,
the specification requirements for the Normative Control Layer are derived on Step 1 of the methodology. The Normative Control Layer is created on Step 2 and Step 3. The development of the Visual Interaction Layer is completed after applying Step 4, Step 5 and Step 6. In case a system designer wishes to enable programming agents through implicit training mechanisms Step 7 should also be completed. Next we present a detailed overview of each of these steps.

**Step 1. Eliciting Specification Requirements.** The initial step of the methodology is the analysis of the application domain by system architects with the goal to elicit specification requirements. This step should be conducted by system architects and normally results in the creation of the Software Requirements Document. In this document the key activities, roles of the participants and basic scenarios are outlined. The suggested methods for eliciting system requirements are interviews, questionnaires as well as other means of applied exploratory and descriptive research. Chapter 6 (Section 6.2.2) presents an example of how this step of the methodology could be performed. Once the specification requirements are established, Step 2 should be used to formalize them.

**Step 2. Specification.** This step is conceptually the same as specifying the institution in the Electronic Institutions methodology [7] and should also be executed by the system architects. It establishes the regulations that govern the behavior of the participants. This process is supported by ISLANDER [66] which permits to specify most of the components graphically, hiding the details of the formal specification language and making the specification task transparent. As it was described in Chapter 2 the specification is determined by the three types of conventions:

- **Conventions on language, the *Dialogical Framework*.** Here the roles the participants are allowed to play, the relationships amongst these roles and common language are specified.

- **Conventions on activities, the *Performative Structure*.** This dimension determines in which types of dialogues the users can be engaged. First the key activities (scenes) are identified. Next the role flow between different scenes is set. Each of the scenes is further associated with an interaction protocol, where a protocol is defined as a final state machine.

- **Conventions on behavior, the *Norms*.** Norms determine which consequences the user actions in different scenes have. These consequences are regarded as commitments that participants acquire while acting in the environments and have to fulfill later on. These commitments restrict future activities of the users. They may limit
the possible scenes that can be entered by the users, and possible actions that can henceforth be executed.

- Interaction protocols, the *Scenes*. Scenes define conversation protocols for a group of roles. Each scene requires a definition of its participating roles and their population, its conversation protocol, and the states at which the agents can either leave or enter the conversation. The scenes of an institution define the valid interactions that the agents are allowed to have and set the context wherein the exchanged illocutions amongst the agents must be interpreted.

If during formalization it became evident that the system requirements elicited on step 1 are insufficient or if for some other reason their evolution is required, the system architect has a choice to return to step 1 for refining the system requirements. If no requirements evolution is necessary and the specification of the system is complete, its validity should be ensured on Step 3.

**Step 3. Verification.** One of the advantages of the formal nature of the methodology is that the specification produced on the previous step can be automatically verified for correctness by ISLANDER. The tool verifies the scene protocols, the role flow among different scenes and the correctness of norms. This verification is static in nature, meaning that the specification has to be finalized before the verification can take place (in contrast to trying to verify on the fly during the specification process). The verification starts with the validation of the correctness of the protocol defined by each scene. This includes checking that the state graph of each scene is connected, that each state is reachable from the initial state and that there is a path from each state to a final state. It is also ensured that the messages associated to the arcs of the state graph are correct with respect to the Dialogical Framework.

The Performative Structure establishes how the participants can legally move among the different scenes. As we do not want them to get blocked in any scene or transition it is verified that from each scene and transition the users always have a path to follow, and that from any scene or transition there is a path to the final scene (so that all participants can leave the institution).

Finally, ISLANDER checks whether the norms are correctly specified and whether the participants can fulfill their commitments. As commitments are expressed in terms of actions that users have to carry out in the future, it is verified that those actions can be performed. For instance, if there is a norm that defines that a user has to pay for acquired products, it is checked that from the scene where the products are purchased there is a path that allows reaching the payment scene.
The verification permits to detect errors in the institutional rules before starting the design and development of the 3D visualization. If such errors are found, the developers should go back to step 2 to correct those. If the specification contains no errors there are two options how to proceed. If the 3D Visualization of the environment is already created (reuse of the existing design) then the developers may skip the next two steps and continue with Step 6. Otherwise, the generation step (Step 4) should be executed.

**Step 4. Automatic Generation.** On the generation step the 3D Virtual World and its floor plan are created in a fully automatic way. Not only the institutional specification defines the rules of the interactions, but also helps to understand which visualization facilities are required for participants to operate in the institution. Some elements of the specification have conceptual similarities with building blocks in 3D Virtual Worlds, which makes it possible to create an automatic mapping between those. On this step of the methodology the Electronic Institution Specification is automatically transformed into the skeleton of the Virtual World.

The scenes and transitions are transformed into 3D rooms, connections become doors, and the number of participants allowed to be in a scene determines the size of each room. The generation can function in two different modes: Euclidean and non-Euclidean. In the first case the rooms on the generated floor plan are positioned so that each room and transition connected in the Performative Structure graph by an arc are physically placed next to each other (connected via a door). In the non-Euclidean case the rooms may be located anywhere and are not necessarily involved in any sort of spatial relationship. The movement between connected rooms in the non-Euclidean Virtual World will then be done using teleportation.

Next, all the rooms are resized to be able to include the maximum number of participants allowed in the corresponding scene. Another outcome of this step is the schematic plan (map) of the institution.

After finishing this step the generated visualization has to be annotated on Step 5.

**Step 5. Annotation.** The Electronic Institutions metaphor is quite generic and can be applied for solving a wide range of problems. Due to this generality the Electronic Institutions Specification was not originally thought to be visualized in 3D Virtual Worlds. It only defines the rules of the interaction and has nothing to say about the appearance of the specified elements. Therefore, a fully immersive and visually rich Virtual World can not be automatically created just on the basis of this specification. In order to make the generated skeleton appealing it has to be enriched with additional graphical elements on the annotation step. These additional elements include textures and 3D Objects like plants, furniture elements etc.
Many components defined in the specification are ignored on the generation step, although some of them have the corresponding objects in the resulting Virtual World. For example, data types defined in ontology can be represented by 3-dimensional objects. If this is the case, on the annotation step a system designer should create 3D models for these objects and manually insert them into the corresponding rooms.

One of the central ideas behind the Virtual Institutions methodology is to facilitate the development of 3D environments so that most of the tasks can be achieved automatically or semi-automatically. The absence of the visualization related information inside the Electronic Institutions specification decreases the degree of automation that can be achieved. Therefore, certain trade offs between full automation and the variety of possible visualizations should be made depending on a particular situation.

Full automation of the annotation step can only be achieved under the condition that the target domain for the application of Virtual Institutions technology is very limited. For example, if the methodology is considered to be applied only to the development of virtual shopping malls, only then it is suggested to automate the annotation step. Such automation can be achieved by limiting the range of possible designs and employing a technology (i.e. design grammars [83]) for selection of an appropriate design for a particular specification element).

If the Virtual Institutions methodology is required to be applied for a very wide range of problems – fully automatic generation is impossible to achieve. In such a case the annotation should be done manually. Further we show how this manual annotation can still be done quite efficiently with the help of the AtmoKits tool.

This step of the methodology does not usually require the involvement of the system architects and should rather be executed by software developers. After this step the user can return to Steps 1 and 2 to refine the specification requirements or the specification itself or can continue with Step 6.

**Step 6. Integration.** On the integration step the execution state related components are specified. This includes the creation of the set of scripts that control the modification of the states of the 3D Virtual Worlds and mapping of those scripts to the messages, which change the state of the Electronic Institution. Firstly, the scripts that correspond to the messages from the agent/institution protocol need to be defined. These include entering or leaving a scene or transition, entering or leaving an institution, etc. Next, the scripts that correspond to the specific messages defined in the ontology on the specification step must be created.

As most of these scripts are required to be present in every Virtual Institution the standard supply of them is provided for every visualization platform. However, in some specific situations these scripts may have to be modified, which should be done on the
integration step.

In case there were any 3D objects representing the ontology data types, the actions upon which require validation - the mapping between these objects and the corresponding data types in the ontology has to be established. At the end, the correspondences between the messages and scripts (actions) are created by filling in the Action/Message table. In case the previous step of the methodology was automatically completed through the application of design grammars, these objects may have already been introduced and there is no further integration necessary. The Action/Message Table in this case should be automatically filled.

Making the integration a separate step of the methodology stimulates the development of the scripts in the form of design patterns, that are generic enough to be reused in other systems.

After accomplishing this step the generated 3D Virtual World is ready to be visualized and the Virtual Institutions infrastructure will be executed to take care of the validity of interactions between participants, verify the permissions of participants to access different scenes and make sure that all the institutional norms and obligations are imposed. This step is particularly important for the case when the system requires to use an already existing design. For existing designs integration can not happen automatically and manual integration is the only possible way to enforce the technological connection between the Visual Interaction and Normative Control layers.

Similar to the previous step, integration should be conducted by software developers. After this step the user can return to Steps 1 and 2 to refine the specification requirements or the specification itself or can continue with Step 7.

**Step 7. Enabling Implicit Training.** Virtual Institutions provide unique facilities for development of autonomous agents. We propose the method, called Implicit Training, which is suggested to be one of the central technologies behind the decision making of the autonomous agents in Virtual Institutions. With the help of Implicit Training the agents can learn sophisticated human-like behaviors from observation of human actions in the 3D environment. Following this approach in many cases it becomes much more efficient to train the autonomous agents than to program them. For some application domains like E-Commerce, where the demand for assistant agents is quite high, Implicit Training is particularly important. It is suggested that the assistant agents for such applications would constantly observe the behavior of human assistants and learn from them how to answer the customer enquiries (without any explicit training efforts required from the humans). In contrast to programming, such approach is much less resource consuming and a lot more flexible.
In case of Implicit Training the Normative Layer of Virtual Institutions forms the basis for the decision tree of the agent, where possible illocutions become the nodes of this tree. For each of those nodes it is possible to specify whether implicit training is conducted or not. This process is completed on the Enabling Implicit Training step of the methodology. The remaining details of the implicit training are presented in Chapter 5.

4.2 Technological Solution

The methodology presented above should be used for generating the technological infrastructure of Virtual Institutions. The specification produced on Step 2 and verified on Step 3 forms the Normative Control Layer. The technology for achieving those steps is already available. The ISLANDER [66] tool supplied with Electronic Institutions Development Environment [7] is a graphical editor used for creating the specification and verifying it for correctness. The completed specification can be stored in an .xml file. XML format is generic enough and there are many parsers available for it. Therefore, for Step 2 and Step 3 of the Virtual Institutions Methodology we suggest utilizing ISLANDER without any modifications.

The steps of the methodology that correspond to the development of Visual Interaction Layer as well as enabling implicit training are not supported with existing tools. Further in this section we propose a technological solution for supporting Steps 4, 5, 6 of the methodology and the details regarding the Step 7 are given in Chapter 5.

Before continuing with the presentation we find it necessary to discuss some details of the implementation. The Virtual Institutions Methodology assumes that the Electronic Institutions specification covers all of the aspects of the Normative Control Layer. In particular, it is assumed that one specification can include the formalization of a number of institutions. Unfortunately, existing versions of ISLANDER and AMELI are unable to cope with multiple institutions defined inside one specification (although, the developers are currently working on the implementation of this feature). Because of this, the technology we propose here only applies to the development of one institutional building.

Another remark we would like to make is that creating a fully functional framework covering all the steps of the Virtual Institutions Methodology was beyond the scope of our research. Instead, we only focused on creating a set of working prototypes that are capable of demonstrating how each of the steps could be facilitated through various existing technologies and provide a proof of concept. We also tried, where possible, to reuse existing solutions and collaborate with other research groups to share their expertise. Therefore, some of the tools described here were adapted from (or fully created by) other collaborators.
Ultimately we aim at providing a technology independent technological solution, however, for the purpose of quick prototyping we had to select a particular technology to be used for visualization of 3D Virtual Worlds. The technology chosen for this task was Adobe Atmosphere visualization platform. In particular, the Trading Institution prototype presented throughout the thesis was implemented using this software. Adobe Atmosphere was selected for quick prototyping purposes based on the comparison presented in Table 2.1. It has extraordinary rendering capabilities, comprehensive object builder and allows for import from major 3D formats (for the case if existing design is required to be reused). No additional software has to be explicitly installed for using Adobe Atmosphere; the necessary plug-in will be installed automatically once the user opens the corresponding page with the web browser. We also required a high degree of control, so that the rules of the institution can be strictly enforced and Adobe Atmosphere was capable of providing such control.

To show that the approach we took is general enough and to a high degree independent of a particular visualization platform we also implemented some of the prototypes using other technologies.

Next we outline our technological solution that aims at facilitating efficient application of Steps 4–6 of the methodology.

### 4.2.1 Step 4. Automatic Generation of Virtual Institutions

To make the visualization of Virtual Institutions more efficient we propose to transform the Performative Structure graph into the corresponding 3D Virtual World in a fully automatic way. There are two possible solutions for achieving this automation. The first approach is the non-Euclidean visualization, where the resulting 3D Virtual World does not have an Euclidean representation. This means that the rooms present in the Virtual World are not involved into any kind of spatial relationship with each other and are independent objets, loaded on demand when a user needs to enter them (through a door). Second approach is to make the visualization fully Euclidean, meaning that whole institution is located within some virtual space, which is abide by the Euclidean laws. The connected rooms in this case are physically located next to each other and separated by doors inside this space.

#### Non-Euclidean Generation

Automatic generation of a Virtual Institution using the non-Euclidean approach is technically simpler. The institutional building and each of the rooms are represented as 3D models each stored in a separate file. The associations between these files are created
with the help of the door objects. Colliding or clicking on such a door will result in uploading the 3D model of the room the user is trying to enter into and positioning the user inside this room.

The non-Euclidean automatic generation method was implemented and tested in the lab by Simon Biber. The “World Generator” software he produced takes an .xml description of an Electronic Institutions specification and transforms each of the scenes and transitions, present there, into 3-dimensional room objects that can be further used by the Half Life 2 game engine. Figure 4.2 outlines the Trading Room of the Trading Institution generated using this mechanism.

In the generated Virtual Institution the movement between rooms can only be achieved by teleports. Each room is constrained by the walls from every side and to leave it a user is required to collide with one of the doors present there.

As the notion of distance and size is not relevant for the non-Euclidean spaces the creation of a schematic plan of the Virtual World is quite a difficult task. However, without such a plan it will be very hard for participants to navigate the Virtual World. Therefore, the Performative Structure of the underlying Electronic Institution was used as the plan (map) of the institution.

---

1 http://half-life2.com
2 3-dimensional objects that force teleportation to a given position inside the Virtual World once collided with.
Even with the Performative Structure acting as the map we found it quite problematic to navigate a non-Euclidean Virtual World. Next we present some research that supports our concerns.

**Importance of Euclidian Representation**

Navigation is an important issue in the design of 3D Virtual Worlds [54, 87, 202, 26]. Navigational problems may break the immersive experience and lead to the rejection of the system by end users [39, 43]. Humans live in an Euclidian world: distances and angles help humans to navigate it. Some researchers [26] hypothesize that the better a Virtual World imitates the real world, the more potential it has to offer consistent experience and support social factors like communication and collaboration.

3D Virtual Worlds could certainly be programmed without an Euclidian model in mind. For instance, we could create a series of rooms interconnected by teleportation. In this case the distance between each two connected rooms will be equal to zero. Technically, such a solution poses much less problems for development, as there is no need to control the positioning of every room while generating the building layout. Instead, each of the rooms can simply be uploaded on demand from a separate file.

Despite the technological simplicity of the non-Euclidean way of representing an institution as a Virtual World, such an approach may cause navigational problems for the participants inside. Moreover, having no Euclidian representation of the Virtual Institution may have negative effect on learning of the institutional structure by the participants.

To demonstrate the confusion a participant may experience navigating a non-Euclidean Virtual World consider the situation (Figure 4.3) where a participant walks from Room 1 to Room 4 following the arrows. In Room 4 the human should correctly expect that Room 1 is behind Door 4, otherwise the believability of the interface will be lost. This expectation of Room 1 is based on both the consistency of the navigational layout and intuitive feeling about the size of visited rooms. In a non-Euclidean case the human can not expect that after entering Door 4 he/she will appear inside Room 1.

Despite the fact that the problems of Euclidean and non-Euclidean visualizations of Virtual Worlds are understudied, there is some research evidence in favor of our initial hypothesis that motion techniques which instantly teleport users to new locations are correlated with increased user disorientation. In [26] the authors present the results of their user study, where one of the questions was whether teleportation can cause navigational problems. The study clearly shows that teleportation (or jumping techniques, as it is called in the paper) can reduce the user’s spatial awareness. With teleportation there is no sensation of motion, only that the world has somehow changed around the user. It is a technique whose motion has no analog in the physical world. Moreover, authors
came to a conclusion that frequent teleportation may even reduce the sense of presence in a Virtual World, which would eliminate one of the most important benefits of the 3D technology.

Another research [169] provides some support in favor of the assumption behind the example presented on Figure 4.3. This user study analyzed the navigational efficiency of participants in the overlapping Virtual Worlds. The results suggest that some users had great difficulties navigating them. Even the experimenter, who informally observed participants as they travelled through the Virtual Worlds, was often unsure of which door to go through to enter a particular room.

Additionally to the above presented evidence in favor of fully Euclidean Virtual Worlds there is another important issue. Many popular Virtual Worlds platforms simply do not allow for non-Euclidean models. Engines like Second Life make profit from selling virtual land, the price of which depends on the physical area of the purchased space. Uploading of additional models on demand from separate files is not supported there. It is not in the interest of the platform developers to facilitate purchasing a minimal space required to include the 3D Model of the building and then infinitely extend this space by uploading the rooms on demand. Due to the growing popularity of Second Life and alikes we can not ignore the issue of automatic generation of Euclidean spaces. Moreover, it is highly desirable to minimize the space occupied by the institutional building to be able to minimize the price of the land purchases.
Euclidean Generation

One of the existing approaches that can be used to achieve automatic transformation of the Performative Structure graph of the existing Electronic Institution Specification into a space optimal building inside a Virtual World is the rectangular dualization.

Rectangular dualization method is successfully employed by Massimo Anacona and Sara Drago from the University of Genoa, Italy [59] to achieve space optimal transformation of graphs into Euclidean maps. Through our collaboration with Massimo’s group their OCoRD tool that is used for generation of rectangular duals of the input graphs was further extended with the possibility to parse an .xml specification of an Electronic Institution and use it to create a graph that satisfies the conditions expressed by their method. As an output of OCoRD a 2-dimensional map of the institution is produced. This map can then be further used by Simon by the World Generator (created by Simon Biber) to generate the 3-dimensional representation of the institutional building.

Although most of the work presented further in this section was conducted by Sara Drago and Massimo Anacona [59], we find it important to present the details of the rectangular dualization method here. The presentation is based on [59].

Rectangular Dualization

Rectangular Dualization was originally used to generate rectangular topologies for floor planning of integrated circuits: by a floor plan, we partition a rectangular chip area into rectilinear polygons corresponding to the relative location of functional entities of the circuit [91]. In spite of the specialized problems that motivated its origin, rectangular dualization contributes to the resolution of many other visualization problems having in common with circuits the condition that objects and their interoccurring relations are represented by means of a planar graph. An example is given by network configuration issues, when human interventions of design or topology adjustment are needed and a physical or logical layout representation becomes essential for the human operator.

In fact, a very serious problem to cope with in graph drawing is how to represent edges in such a way that they do not appear too close together. The aim is to enhance the readability of the drawing, making easier to find out which nodes are connected by an edge. The very first solution to this problem is to avoid edge crossings, and this motivates the interest for planar graphs, that are precisely those graphs that can be drawn in the plane with no edge crossings. The choice for planar graphs is not only a representation facility but is primarily validated by real-world examples where the presence of crossing links may produce technical drawbacks.

Further on, since a major optical effort is encountered in the proximity of vertices, where adjacent edges need to meet in a point, several studies have been focused on de-
vising drawing algorithms capable of maximizing angular resolution, i.e. the smallest angle between adjacent edges, in such a way that lines representing connections are kept as separate as possible; rectangular dualization results to be an effective visualization method since only orthogonal lines occur.

Definitions

A rectangular dual of a planar graph $G = (V, E)$ is a rectangle $R$ with a partition of $R$ into a set $\Gamma = R_1, \ldots, R_n$ of non-overlapping rectangles such that:

- no four rectangles meet at the same point;
- there is a one-to-one correspondence $f : V \rightarrow \Gamma$ such that two vertices $u$ and $v$ are adjacent in $G$ if and only if their corresponding rectangles $f(u)$ and $f(v)$ share a common boundary.

It is easy to see that if a graph admits a rectangular dual, it may not be unique (see Figure 4.4).

On the other side, some graphs do not admit rectangular dual. Kozminski and Kinnen present necessary and sufficient conditions under which a plane graph $G$ has a rectangular dual [112]: the most important point is the absence of separating triangles (i.e. 3-vertex cycles with at least one vertex in their interior), a condition that in planar triangulations is equivalent to 4-connectivity whose meaning is that the removal of any set of 3 vertices leaves the remainder of $G$ connected. A matching in $G$ is a subset $M$ of edges such that for every vertex $v$, at most one edge $e$ covers $v$, that is $v$ is an endvertex of $e$. A graph is $k$-regular if every vertex has degree $k$, that is $k$ incident edges. A maximum matching is a matching with largest possible cardinality. If the graph is weighted, we may even consider a maximum weight matching. A bridge is an edge whose removal...
disconnects \( G \). Whenever we speak of a planar graph, we assume that some planar embedding has been fixed, which corresponds to the idea of depicting an existent physical connection among real objects (in this perspective, it would be more accurate to speak of \emph{plane graphs}, i.e. planar graphs with a fixed embedding in the plane). A \emph{structured graph} is a form of abstraction applied to a large graph in order to make it modular and more manageable. The abstraction consists in collapsing a subgraph to a single vertex (called a \emph{macrovertex}), or to a single link (called a \emph{macrolink}) thus obtaining a simpler and hierarchically described graph. The structuring operation is usually iterated recursively until a large graph is decomposed into relatively small and manageable components and sub-components defined at several levels of nesting (see Figure 4.5) adopting a methodology that is usually applied to every large project (software and hardware design) involving hundreds or thousands of components: i.e. “modularity”.

**OCoRD Software**

The OCoRD software is a tool aiming at the solution of floorplanning problems and at the orthogonal drawing of planar networks. Given a planar embedding of a graph, it accepts a numeric adjacency lists and, if the graph admits a rectangular dual, it returns its coordinates in the plane and a drawing in \texttt{fig} format. Moreover, OCoRD transforms a graph not admitting a rectangular dual into a 4-connected supergraph satisfying Kozminski and Kinnen criterion. In fact, it is possible to eliminate separating triangles by adding crossover vertices on an edge of each separating triangle [116]. The implemented breaking method is optimal, in other words it only adds a minimum number of such crossover vertices and inserts them in strategical positions, such that a single vertex may break two
triangles or more. The method, described in [3], performs this transformation in the following steps:

1. four external vertices are added according to the construction presented in [107];
2. the geometrical dual of the graph is computed and faces belonging to a separating triangle are detected and clustered together in a single macrovertex;
3. a covering affecting macrovertices is computed; the effect is that all separating triangles are optimally broken by inserting new vertices in some strategical places along some of their constituting edges;
4. the resulting graph is triangulated with the algorithm described in [21].

We have so far obtained a graph satisfying triangularity and four-connectivity. Faces belonging to a separating triangle are detected in linear time [40]. Separating triangles are broken by solving a minimum unweighted macro-covering problem on the geometrical dual of the input graph. The macro-covering can be computed by solving a sequence of minimum weighted edge covering problems on each simple graph of the structured dual. In [150], Parekh showed how to reduce a minimum weighted edge cover of a specified subset of the vertices of $G$ to a maximum weighted $b$-matching, a well solved problem [62] that is worked out by implementing the $O(mn \log n)$ algorithm presented in [71]. The input graph is biconnected and it is described through adjacency lists satisfying the following properties (see Figure 4.6):

- vertices are indexed by non negative integers $(1, \ldots, n)$;

![Figure 4.6: A planar embedding of a graph with its input file.](image)

140
• for each internal vertex, its adjacencies are clockwise listed, according to a fixed planar embedding of the graph, starting from the vertex having the lowest index;

• for each vertex belonging to the graph contour, its adjacencies are clockwise listed, starting from the vertex on the contour which precedes it in a counterclockwise direction with respect to the contour.

We note that in the modification perspective, also input graphs that are not under the biconnected constraint may be processed: a preliminary step can be implemented to provide this degree of connectivity [165]. Figure 4.7 gives an overview of the implemented rectangular dual construction method:

<table>
<thead>
<tr>
<th>Algorithm: Rectangular Dual Construction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong></td>
</tr>
<tr>
<td><strong>Output</strong></td>
</tr>
<tr>
<td><strong>Add four external vertices</strong></td>
</tr>
<tr>
<td><strong>Compute the geometrical dual graph G</strong></td>
</tr>
<tr>
<td><strong>Detect faces belonging to a separating triangle, for any</strong></td>
</tr>
<tr>
<td><strong>Collapse them into a macrovertex in G</strong></td>
</tr>
<tr>
<td><strong>Solve the macro-covering problem in G</strong></td>
</tr>
<tr>
<td><strong>Add new vertices in G</strong></td>
</tr>
<tr>
<td><strong>Triangulate G [21]</strong></td>
</tr>
<tr>
<td><strong>Compute a REL</strong></td>
</tr>
<tr>
<td><strong>Compute the rectangular dual coordinates</strong></td>
</tr>
<tr>
<td><strong>Delete external vertices and draw</strong></td>
</tr>
</tbody>
</table>

Figure 4.7: Algorithm: Rectangular Dual Construction

As a refinement of the described method, we may say that the logarithmic cost of the method is due to the fact that the aforecited matching algorithm holds for general graphs, a much wider class of graphs than the one we deal with in our planarity assumption. Instead, a matching in a 3-regular bridgeless graph can be found in linear time [20]. Since the collection of all planar 3-regular bridgeless graphs is exactly the collection of duals of planar triangulations where the outside face is a triangle, we may tighten the bound by solving the matching problem on the dual of a planar triangulation and this can be obtained by producing a triangular outer boundary and by running the algorithm [21] (which triangulates without adding new separating triangles) before the computation of the geometrical dual.

**Adaptation of OCoRD to Virtual Institutions**

The algorithm presented in Figure 4.7 as well as the OCoRD software are further adapted for the visualization of Virtual Institutions.
From the Performative Structure graph of the institution both the 3D representation of the institutional building and the map of the institution are created. The process of automatic generation of the layout of the institutional building as well as the corresponding map from this graph is depicted in Figure 4.8.

Figure 4.8: Generating the 3D representation of a Performative Structure graph.

In the upper left corner in Figure 4.8 the source Performative Structure graph is presented. This graph corresponds to the Trading Institution explained in Section 3.2. Rectangular shapes represent scenes, triangular shapes are transitions and directed arrows (arcs connecting nodes) are connections.

The automatic generation is done in 3 steps. On the first step the Performative Structure graph is transformed into the format understandable by OCoRD software and the unnecessary information is filtered out. If some nodes of the graph are connected with more than one arc, only one randomly selected arc is left and all the others are deleted.

After the transformation is completed the step 2 is executed, on which the OCoRD
software reads the transformed graph and creates its rectangular dual. The rectangles, which were introduced because of the breaking points, are removed and the adjacent rooms are reshaped to the size of removed rectangles. The Root and Exit scenes are always present in the performative structure graph. Moreover, the root scene is not permitted to have incoming arcs and the exit scene doesn’t have any outgoing arcs. As those scenes are not visualized, the corresponding graph nodes are ideal candidates to be two of the four external vertexes. In this way the garden will be automatically created as the rectangle surrounding the graphs rectangular dual. When the rectangular dual is produced, the only thing that is left is placing the doors between connected rooms. The outcome of this step is the map of the institution, which is presented in the lower left corner of Figure 4.8.

Step 3, transforming a 2D map of the institution into a 3D Virtual World, is the task of the World Generator software. The coordinates of the map are first transformed into the coordinates of the 3D Virtual World. Then, every room is scaled so that it can physically contain the maximum number of participants that is defined for it. Later on, the corresponding 3D objects are reshaped and put inside the 3D Virtual World. The result will look similar to the bottom right part of the Figure 4.8.

4.2.2 Step 5. Annotation

In order to make the generated institutional building appealing, it has to be annotated with additional visual elements. As it was mentioned before, if a system architect plans to employ the Virtual Institutions methodology in a limited domain, it is possible to have the annotation happening automatically through the utilization of design grammars [168]. The essence of this approach is to capture the style of the facade and rooms inside the institutional building by so-called design rules. Design rules illustrate and describe the forms and functions of each design element. These rules are represented by different combinations of the basic primitives existing in the 3D Virtual Worlds.

One of the most famous types of design grammars are shape grammars [195]. Shape grammars are a specific class of production systems that generate geometric shapes. A shape grammar consists of shape rules and a generation engine that selects and processes them. A shape rule defines how an existing shape can be transformed. It consists of two parts separated by an arrow pointing from left to right (i.e. \( A \rightarrow A + B \)). The left hand side (LHS) of the rule depicts a condition for room execution in terms of a shape. The right hand side (RHS) determines how the LHS shape should be transformed and outlines the result of the room execution [195, 168].

To give an impression of what shape rules look like, Figure 4.9 presents a simplified version of the rule for positioning a table inside a room.
Shape grammars can be successfully utilized to generate suitable 3D designs for the institutional rooms. They provide a linkage that translates the “specification language” of the institution into the “design language” of 3D Virtual Worlds.

The shape grammar approach has a number of advantages. Firstly, the changes of the specification and of the runtime environment can be automatically addressed through the grammar application (by alternating the choices and order of the design rules during the application). Secondly, the use of the shape rules makes the system platform independent. Using a different 3D Virtual World platform for design implementation will only require a simple re-mapping between the elements of the database and basic primitives of the Virtual World, without undergoing any major system changes.

A successful attempt to use shape grammars for enhancing the appearance of Virtual Institutions was made by Owen Macindoe and Ning Gu from the University of Sydney [84]. Figure 4.10 outlines the visualization of the Trading Institution they created following the shape grammar approach [83]. The resulting design is rendered by the Second Life visualization platform. Each of the rectangular spaces connected to the poll on this picture represents one of the scenes in the Trading Institution.

Automatic designs are, obviously, not always suitable for developers. There are situations when a particular design solution is required for a particular institution. In such a situation each (or some) of the rooms generated on the Automatic Generation step of the methodology will be originally empty and the design of the rooms will have to be enriched manually (annotation phase).

Even for this kind of situations there are tools available on the market that could significantly improve the speed of the annotation. One of such tools is AtmoKits software. This software is compatible with the Adobe Atmosphere platform that was initially used for the visualization of the 3D Virtual World. It is capable of uploading the model of the Virtual World (produced on the generation step), enriching it with additional elements and saving the result in Adobe Atmosphere compatible format.

AtmoKits is supplied with the predefined sets of objects, but also supports including

---

Figure 4.9: An Example of a Shape Rule.
new objects into existing sets. The most popular standard set of objects is AtmoKits Household Kit. It can be successfully utilized to annotate the majority of commercially oriented institutions. The set contains textures, furniture pieces, plants and household objects. Figure 4.11 gives a rough idea about how the designs of the rooms can be changed via Atmokits and illustrates some of the objects available in the Household Kit.

The upper left side of the picture shows the currently selected set of objects (Household Kit) and supports the navigation through different types of objects available there. Further below the schematic plan of the uploaded institution is outlined. The bottom part of the interface is used for selecting an appropriate object from the list, moving it, rotating and scaling. The movement, rotation and scaling of the objects is done by clicking the corresponding arrow buttons. The status line at the very bottom of the window shows the detailed information about the currently selected object. The “Remove Model” and “Lock Model in Place” buttons support erasing the current object from the room or locking it in the current position. The right hand side of the interface shows the created design in real time. All the objects transformations and position changes are immediately reflected onto the interface. The system designer is represented as an avatar with default appearance. It is possible to fully explore every room and move between different rooms by changing the position of this avatar.

Another very useful feature of AtmoKits software is the possibility of collaborative room annotation. Each of the designers in this case is visualized as an avatar, so that a number of designers can work on the design simultaneously, observe the design evolution in real time and communicate with each other through their avatars.
4.2.3 Step 6. Integration

The Virtual Institutions model proposed in this thesis assumes that there is a tight connection between the Visual Interaction Layer and Normative Control Layer. To make this connection happen every action of the Visual Interaction Layer that requires verification has to be validated by the Normative Control Layer. Most of the 3D visualization platforms (including Adobe Atmosphere that is currently being employed) support representing these actions in terms of scripts. In case of Adobe Atmosphere these scripts are blocks of text structured following the Java Script notation\(^4\). Such a view permits to express every action of the Virtual World as a text block uniquely identifiable by its name. Adobe Atmosphere offers straightforward mechanisms to execute such a script if its name is provided.

An efficient way to link the two layers is to create a mapping between the actions (scripts) that change the state of the Visual Interaction Layer and the actions (illocutions) that change the state of the Normative Control Layer. In the current implementation this mapping is achieved by employing the Action/Message table, where each script from the

\(^4\)http://www.javascript.com
Visual Interaction Layer is assigned with a corresponding message from the Normative Control Layer (where a message can be either a part of the illocution sent by the agent to the institution or a part of the institutional response).

The actual methodology employed for conducting this step may be different and depends on the visualization platform being used. Therefore, it is not the goal of this dissertation to provide specific technical details as to executing this step. As a general recommendation we suggest employing the Action/Message table approach described above.

Table 4.1 presents a simplified fragment of the Action/Message table for the trading institution. The table specifies that “addNewAvatar” script should be executed as the result of receiving the “EnteredAgentInstitution” message from the institutional infrastructure, which corresponds to entering the institutional building. The “leftSceneRoom” script will be launched before the institution notifies the agent that it has successfully exited a scene. The “enteredSceneRoom” script will be started when the institution updates its state letting the agent to move inside a scene, etc.

<table>
<thead>
<tr>
<th>Action</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>addNewAvatar</td>
<td>EnteredAgentInstitution</td>
</tr>
<tr>
<td>leftSceneRoom</td>
<td>ExitedScene</td>
</tr>
<tr>
<td>enteredSceneRoom</td>
<td>MovedToScenes</td>
</tr>
<tr>
<td>leftTransitionRoom</td>
<td>ExitedTransition</td>
</tr>
<tr>
<td>enteredTransitionRoom</td>
<td>MovedToTransition</td>
</tr>
<tr>
<td>raiseHand</td>
<td>Bid</td>
</tr>
<tr>
<td>removeAvatar</td>
<td>ExitedAgentInstitution</td>
</tr>
</tbody>
</table>

All of the institutional messages shown in the table represent the responses of the institution to the agent requesting to perform an action. For example, the “MovedToScenes” message will be sent as the result of the institutional infrastructure receiving an “EnterScene” message from the agent. This message will be sent back to the agent by the institution after the “EnterScene” message is validated and accepted by the institution, and the state of the institution is updated to reflect the fact that the agent entered the desired scene.

At the current stage of the implementation the script creation and population of the Action/Message Table with data has to be done manually by a system designer (although some standard script templates are already available). However, we plan to automate some stages of this process in the future.

The creation of the Action/Message table is the main task that should be completed on the integration step. The population of the table with data should proceed as follows.
First, the name of each of the scripts from the Visual Interaction Layer that requires institutional verification has to be inserted into the table and the corresponding message has to be assigned to it. Next, each of the message names from the agent-governor protocol has to be inserted into and associated with the corresponding script name.

During the integration process it is also required to explicitly specify which scenes or transitions in the Normative Control Layer correspond to which rooms in the Visual Interaction layer and which doors should be opened in order to access a desired scene or transition. This is necessary to control the avatar movement inside the Visual Interaction Layer and to determine whether an avatar is trying to enter or leave a scene or transition (so that the state of the Normative Control Layer can be updated accordingly). It is also important to have this information to be able to force the movement of an avatar into a particular scene or transition in case this movement is requested in the Normative Control Layer.

In case the 3D representation of the institutional building was automatically generated from the specification the information about the mapping between scenes and transitions to rooms and connections to doors is readily available and there is no need for explicit manual efforts from the system designer. Otherwise, if the existing design is reused for the 3D model of the institutional building the automation of this process is not possible. The aforementioned mapping has to be conducted manually. For this purpose the following two tables have to be populated with data.

The first table specifies the details of the scene/room mapping in the Virtual Institution. An example of such a table created for the Trading Institution is presented in Table 4.2. The Name column determines the Name of a scene as set in the specification of the Normative Control Layer. The Bounds Object column specifies the name of the Visual Interaction Layer object that should be used to determine the bounds of the scene. The Entrance Door sets the name of the door objects from the Visual Interaction Layer. The bounds of the room objects (the coordinates of the imaginary cube surrounding the room) in Adobe Atmosphere can be easily determined by calling the “roomName.bounds()” function. In case another visualization platform is used and such a function is not available it may be required to populate the table with actual coordinates of the vertexes of the cube rather than the names of the corresponding objects.

<table>
<thead>
<tr>
<th>Name</th>
<th>Bounds Object</th>
<th>Entrance Door</th>
</tr>
</thead>
<tbody>
<tr>
<td>meetingRoom</td>
<td>Scene/ceiling</td>
<td>advDoor</td>
</tr>
<tr>
<td>registrationRoom</td>
<td>Scene/registration/registrationRoom</td>
<td>entrDoor</td>
</tr>
<tr>
<td>tradeRoom</td>
<td>Scene/tradingfloor/ceiling</td>
<td>alleyDoor</td>
</tr>
</tbody>
</table>

Table 4.2: Scene Bounds.
Table 4.3 outlines the details of the transition/room mapping in the Trading Institution. The columns in this table have the same meaning as in Table 4.2.

<table>
<thead>
<tr>
<th>Name</th>
<th>Bounds Object</th>
<th>Entrance Door</th>
</tr>
</thead>
<tbody>
<tr>
<td>toMeetingRoom</td>
<td>Scene/main hall/reg2Adv</td>
<td>regDoor</td>
</tr>
<tr>
<td>toTradeRoom</td>
<td>Scene/main hall/Adv2Trading</td>
<td>tradingDoor</td>
</tr>
<tr>
<td>toOutputFromTradeRoom</td>
<td>Scene/infoRoom/info</td>
<td>exitDoor</td>
</tr>
</tbody>
</table>

As a good practice of Virtual Institutions development we propose that every institution accepts the entrance of “guest” agents into the registration room where they will be able to request the information about the activities conducted inside and receive authorization to enter other rooms. To be able for a participant to proceed to further rooms he/she should be assigned with an acceptable role. For secure role assignment we propose using the identification of each user by entering login and password details. We see this process being a common practice and, therefore, each institution is provided with the authorization mechanism. The registration details for different users in the institution are stored in the permissions table. Table 4.4 presents a fragment of this table for Trading Institution.

<table>
<thead>
<tr>
<th>User Name</th>
<th>Password</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>testBuyer</td>
<td>bdPUEWiFycSfg</td>
<td>Buyer</td>
</tr>
<tr>
<td>testSeller</td>
<td>A8c/KTq5QcktA</td>
<td>Seller</td>
</tr>
<tr>
<td>roomMgr</td>
<td>n55uVIwxxkbHew</td>
<td>roomManager</td>
</tr>
<tr>
<td>janedoe</td>
<td>N/70mMPI3B/3g</td>
<td>receptionist</td>
</tr>
</tbody>
</table>

The User Name column stores a unique identifier of a user. The Password field stores the hashed value of the password for each of the participants. The Role field contains the role that should be assigned to a participant in the Normative Control Layer after successful completion of the registration process.

Depending on the situation the permissions table can either be manually filled on the integration step or populated with data during deployment.

Populating the aforementioned tables with data creates a link between the Visual Interaction and Normative Control layers and makes it possible to maintain the tight layer connection at run time and be certain that the Virtual Institution is enacted as expected. In the next section we present the technology that was used to deploy the system.
4.3 Deployment

Applying the Virtual Institutions Methodology described in the previous section results in the creation of the Electronic Institution Specification (the central component of the Normative Control Layer) and the 3D model of the Virtual World (containing the institutional building that represents this specification). The logical connection between the two layers is achieved on the integration step. In order to maintain this connection and make the Virtual Institution functional some additional efforts are required. In this section we present the result of these efforts - the architecture for deployment of Virtual Institutions.

From the software engineering prospective, the deployment of Virtual Institutions can be achieved by designing a single system, where Normative Control Layer and Visual Interaction Layer will remain logically separated, but technologically united. Alternatively, it is possible to make the technological separation and try to reuse existing technologies for implementation of each of the layers.

The reality imposes second choice. On the one hand, such approach is justifiable from the perspective of saving programming resources. Implementing a new architecture for each of the layers would cost significant programming effort, which in case of reusing existing technological solutions can be saved. On the other hand, this technological separation seems to be the most efficient way to deal with rapid development of visualization platform for Virtual Worlds. As it was shown in Chapter 2 it is often the case that one technology used for the visualization of 3D Virtual Worlds quickly changes another and the former eventually disappears. Relying on a particular technology for deployment of Virtual Institutions may have a consequence that the potential users may abandon them. Additional benefit of having the two layers technologically separated is the fact that the Normative Control Layer is quite general and it is not necessary for it to be used only with conjunction with Virtual Worlds. It should be possible for other technologies to access it directly and bypass the Visual Interaction Layer. Such an approach, for example, can be useful for connecting a Virtual World and the real world. Finally, technological layer separation is more flexible in terms of development and allows for distributed programming.

In the architecture we propose the technological separation of the two layers is supported through the introduction of the middle layer (Communication Layer) that is used for technological connection of the Normative Control and Visual Interaction layers. This 3-layered infrastructure is illustrated in Figure 4.12. First layer here is the Normative Control Layer. It uses the AMELI system [7] to regulate the interactions of participants by enforcing the institutional rules established on the specification step. AMELI maintains the execution state of the institution and uses it along with the specification to guarantee that participants’ actions do not violate any of the institutional constraints.
Second layer is the *Communication Layer*. Its task is to causally connect the institutional infrastructure with the visualization system and transform the actions of the visualization system into messages, understandable by the institutional infrastructure and the other way around. This causal connection is achieved via the Causal Connection Server, which uses the Action-Message table created on the integration step to establish the mapping between actions of the Visual Interaction Layer and messages of the Normative Control Layer. The causal connection is happening in the following way: an action executed in the 3D Virtual World (that requires institutional verification) results in a change of the institutional state in the AMELI layer, as well as every change of the institutional state is reflected onto the 3D Virtual World and changes its state.

The third layer is called *Visual Interaction Layer*. It is used to visualize the 3D Virtual World for the users. The advantages of separating of the runtime architecture into three different layers are listed below:

1. The interactions inside the 3D Virtual World become structured, secure and predictable, as everything that needs control is verified by AMELI and will happen as specified.

2. The Visual Interaction Layer can be easily replaced (i.e. when a more advanced visualization platform appears on the market) with minimal changes required for the rest of the system.
3. The changes in the Normative Control Layer can be automatically reflected onto the Visualization layer or will require minimal manual adjustment.

4. A number of different visualization platforms (possibly implemented via different technologies) can be simultaneously connected to the Causal Connection Server and share the same institution.

5. Some participants (i.e. software agents) can bypass the 3D Virtual World and directly connect to the institution via the Normative Control Layer, while other participants (humans) will be able to observe their presence and actions in the 3D Virtual World.

It is important to understand that Virtual Institutions change the philosophy behind the security control initially introduced by Electronic Institutions. Direct connection to AMELI system restricts the actions of the user to those that are allowed in the current state of the institution for the particular role a user is playing, nothing else can be done. Having 3D Virtual World as a part of the system creates the possibility to have an opposite view. The institutional infrastructure does not anymore dictate what the users are allowed to do in the system but rather controls some critical issues and permits everything else. A 3D Virtual World client usually supports some standard actions: walking, chatting, turning around, jumping etc. Having no institutional rules present in the system means that everyone is allowed to do everything, because nothing prohibits users to execute any of those actions. The Electronic Institutions Specification enforces the rules of the interactions, which should be obeyed where present.

Next we present a detailed explanation of the technological solution used for deployment of each of the three presented layers.

### 4.3.1 Normative Control Layer

The Normative Control Layer is Deployed using the AMELI tool provided with the Electronic Institutions Development Environment [7]. AMELI is capable of reading the .xml file with the institutional specification and enforcing the institutional rules specified there. After reading the specification it opens a socket connection for participants and for every new connection done through this socket assigns a new governor agent, which will further control the actions of the connected participant.

The AMELI system maintains the runtime state of the institution and updates it as the result of the illocutions sent by a participant to the corresponding governor agent.
4.3.2 Visual Interaction Layer

The Visual Interaction Layer is currently supported by Adobe Atmosphere player embedded into an HTML page accessible through a web interface by web browsers. Each of the participants initially connects to the system by accessing a web form, where a user is requested to type in the identification details (user name and password). After pressing the “Submit” button the connection request is sent to the Communication Layer. The Communication Layer creates a corresponding agent and sends the “EnterAgentInstitution” request to the Normative Control Layer. If the permission to enter the institution is granted - the Web interface is notified and as the result the web browser loads the corresponding .aer file with the 3D model of the Virtual World.

The Atmosphere Player is unable to directly communicate with the Causal Connection Server, therefore, the web interface is supplied with a Java Applet. The Java Applet connects to it using the socket connection.

The .aer file, which the Atmosphere Player operates with, stores the model of the Virtual World produced as the result of applying the Virtual Institutions Methodology. This model is represented by the set of mathematical equations associated with each object present in the Virtual World. To visualize it the Atmosphere client applies algorithms for translation of this mathematical models into a 2-dimensional images, which are constantly updated as the result of actions of the participant associated with the client or as the result of the actions performed by other participants.

As most of the game engines or other Virtual Worlds platforms, Adobe Atmosphere has a separate component that is used to maintain the states of the participants inside the Virtual Worlds and mechanisms for changing this state. This component is called Adobe Atmosphere Community Server. Every participant entering the system establishes a two directional connection with the Community Server. Every 50 Ms the coordinates of each of the avatars and corresponding transformation vectors are communicated to the Community Server and the Community Server distributes this values to each of the connected clients. In this way each of the participants is able to observe the updated state changes requested by other participants every 50 Ms.

4.3.3 Communication Layer

The task of causally connecting the Electronic Institutions runtime environment AMELI with the Virtual World visualized by the Visual Interaction Layer is accomplished via the Communication Layer. A system is said to be “causally connected” to its representation if whenever a change is made in the representation, the system itself changes to maintain a consistent state with the changed representation, and whenever the system evolves, its representation is modified to maintain a consistent relationship [127]. Reflective systems
are a particular case in which the representation of the system is part of the system itself. The Electronic Institution (execution) has a representation of itself in the form of a 3D environment consisting of rooms, avatars, doors and other graphical elements. The 3D visualization provides a possibility for humans to interfere with the agent representing their interests (that runs over the institution infrastructure) through actions in the 3D environment. These actions and the agent’s actions have to be consistent. Thus, a causal connection between the Electronic Institution and the Virtual World seems to be a must.

To achieve the Causal Connection between the Normative Control Layer and Visual Interaction Layer each participant can connect to the system through any of these layers and the participants’ representation in another layer will be automatically created. If a human enters the system as an avatar in the Virtual World – the corresponding agent is created in the Normative Control Layer to represent the human there and communicate the actions of the human to the institutional infrastructure. On the other hand, if an autonomous agent directly connects to the Normative Control Layer – the avatar representing this agent is created in the Visual Interaction Layer.

The causal connection has to materialize in two directions. First, the state changes of the Normative Control Layer must have an immediate impact on the Visual Interaction Layer. Movements of an agent between scenes, for instance, must make the corresponding avatar “move” in the 3D Virtual World accordingly. Second, actions performed by the human in the Visual Interaction Layer are understood as made by the agent in the Normative Control Layer. The movement of the human between different rooms inside the Virtual World must result in the changes of the AMELI state so that the associated agent “moves” to the corresponding scenes or transitions. This has a consequence that the actions that the agent is not allowed to perform in the current execution state cannot be permitted in the 3D environment. For instance, if in the current state of the Normative Control Layer it is prohibited for an agent to leave a scene, opening an exit door of the corresponding room must be prohibited to the avatar in the Virtual World.

The task of causally connecting the two layers is performed by the Causal Connection Server. For every human that enters the Virtual World through the Visual Interaction Layer the Causal Connection Server sends a request to the AMELI infrastructure for creating an agent in the Normative Control Layer. It can also manipulate this agent and send messages on its behalf. The Causal Connection Server is also able to create avatars in the Visual Interaction Layer if an autonomous agent desires to directly connect to the Normative Control Layer.

Each agent that connects to the Normative Control Layer is virtually associated with a Governor inside the AMELI infrastructure. The Governor acts as the third party that communicates the desires of the agent to the institutional infrastructure and answers the requests of the agent about the institutional rules. The messages received by the Gov-
error are forwarded to AMELI for validation. More precisely, AMELI verifies whether a particular message goes in line with the specified rules or not. If a positive validation response is given by AMELI, the requested action gets the permit to be performed. The Governor sends the response of the institutional infrastructure to the Causal Connection Server. As the result of this, the corresponding action is then reflected at the Visual Interaction Layer.

Figure 4.13 outlines the mechanism for achieving the causal connection and illustrates the difference between events, actions and messages.

![Figure 4.13: Causal Connection.](image)

An event is generated as the result of a human performing an action in the physical world with the goal in mind to change the state of the Virtual World. In the particular case illustrated in Figure 4.13 this physical action is positioning the mouse pointer over the door handle and clicking the left mouse button (requesting the avatar in the Virtual World to open a door by pushing the door handle). Each event that requires institutional verification must have an associated script and the name of this script has to be present in the Action/Message table. If this is the case the name of the script is forwarded to the Communication Layer, where the Causal Connection Server consults with the Action/Message Table to find the message that represents this event in the Normative Control Layer. In case such a message is accepted in the Normative Control Layer
the response message is sent back to the Communication Layer. The Causal Connection Server again consults the Action/Message table and transforms this response into the name of the action that has to be executed in the Visual Interaction Layer and the action is performed by executing the corresponding script. In the given example this action will result in opening the door and moving the avatar through the door.

The Causal Connection Server

The development of the Causal Connection Server is one of the key technological contributions of this thesis. While for the deployment of the Visual Interaction Layer and the Normative Control Layer we adapted and employed existing technologies, the Causal Connection Server had to be developed from scratch. The architecture behind the Causal Connection Server enables the Communication Layer to be independent from the technologies used in the other two layers. As the proof of this the group of Helmut Berger in Austria developed a prototype of the Itchy Feet system [18] that was built based on the Virtual Institutions methodology and the Causal Connection Server was successfully employed to connect the AMELI to the Visual Interaction layer based on the Torque Game Engine by GarageGames. Figure 4.14 demonstrates the interface of the Itchy Feet system. This system implements a 3D online tourism portal. The institution is represented as a building located inside the garden.

Further in the thesis we show how the Causal Connection server is used in combination with AMELI and Adobe Atmosphere, however, the reader should keep in mind that these two technologies can be replaced by some other technologies and the Causal Connection Server would still be capable of performing the delegated task.

Figure 4.15 outlines the key components of the three deployment layers with the detailed focus on the architecture of the Causal Connection Server. The AMELI system represents the Normative Control Layer and is featured with two additional components: Federation Monitor and Institution Monitor. These components are used to capture the responses of all governors to all the participants connected to a particular execution of AMELI. The difference between these two components is that the Institution Monitor only controls the responses to the actions inside the institutional building, while the Federation Monitor is capable of capturing the messages that correspond to the actions in the garden, outside any of the institutional buildings. The Institution Monitor and Federation Monitor offer an interface to AMELI, allowing the observation of all messages within a single Electronic Institution platform. The Causal Connection Server is connected to AMELI through sockets provided by these monitors and collects available messages.

---

1http://ispaces.ec3.at
2http://www.garagegames.com/
The collected messages assist in maintaining the synchronized and consistent relation between the Normative Control Layer and the Visual Interaction Layer.

The Visual Interaction Layer is represented by the Atmosphere Player embedded into a Web interface. A human can only act in the Virtual Institution by controlling the avatar inside the Atmosphere Player.

The Communication Layer is represented by the Causal Connection Server and Agent Launcher. The task of the Agent Launcher is to create agents on request of the Causal Connection Server and to force these agents to send the desired messages to the corresponding AMELI governors. The Causal Connection Server consists of the number of components. First component is the Communicator. Its task is to communicate with AMELI, Agent Launcher and Web Interface. Next, there are two socket monitors, which constantly observe the sockets of the communicator for messages received from Institution and Federation monitors. To be able to communicate the messages through a socket they have to be first transformed into string form. The task of the StringToMessageProcessor is to translate them back into a special Message class, that can be used for easy access to the message parameters. As the number of messages that can be received can be quite high and all of them can not be processed in real time, all the messages are
first put into the Message queue. In Figure 4.15 two different message queues are illustrated by two directed arrows pointing in opposite directions. One queue contains the messages received from AMELI and another one – the messages that should be sent to AMELI. Two Message Monitors constantly observe these queues and make sure that all the messages are further delivered for processing.

The processing of the messages received from AMELI is done by the Action Composer. This component consults the Action/Message table to transform a message into an action, which is then further passed to the Atmosphere Player through the Communicator and the Web Interface. The Message Composer does the opposite: it transforms events received from the Atmosphere Player into messages that should be sent to AMELI for verification. In order to create the correspondence between avatars and agents, the AvatarID/AgentID table stores their identifiers and helps in making the mapping between them. Such a mapping is necessary for being able to correctly dispatch the messages and actions to the recipients.

Two types of participants are considered in Figure 4.15, namely humans and autonomous agents. Humans connect to the system via the web interface, through which a
user validates the access to the system and if admission to the institution is granted, the Adobe Atmosphere Player [4] is loaded and starts to visualize the 3D Virtual World. At the same time, a message is sent via the Causal Connection Server to the Agent Launcher that, in turn, spawns a new software agent. This software agent represents the human in the Normative Control Layer and communicates human’s requests to the corresponding Governor, which is created by the institutional infrastructure as soon as the autonomous agent requests to enter the institution.

The second type of participants are autonomous agents (marked as External Entity on the picture) that contact AMELI directly. The agents are unaware of the availability of the Visual Interaction Layer and only know how to interact with AMELI. Each such agent requests access to enter the institution and if the access is granted – communicates with the institution via a newly assigned Governor. The figure demonstrates how the actions of both types of participants are handled by the Causal Connection Server.

An arbitrary event, e.g. a mouse click on a door handle, caused by a human participant leads to a sequence of processing steps. First, the event is caught by the Atmosphere Player and transmitted in terms of a 2–tuple \(<\text{AvatarID, Event}\>\) to the Causal Connection Server. Then the event tuple is stored in the Event Queue which is observed by the Event Monitor. As soon as the Event Monitor notices the arrival, it translates the event by means of the Event/Message mapping table into the corresponding message. In analogy to that, the AvatarID is mapped onto the AgentID by means of the AvatarID/AgentID mapping table. A 2–tuple \(<\text{AgentID, Message}\>\) is composed and stored in the Message Queue. This time the Message Monitor detects the arrival and sends it to the corresponding agent via the Communicator. Finally, the agent passes the message to the corresponding governor. The governor validates whether the received message adheres to the institutional rules and generates an adequate response.

Messages originating from AMELI need to be reflected onto the Virtual World and are processed in exactly the opposite way. Every institutional message is first received by the Communicator, which puts it into the incoming message queue. When the Action Composer is ready to process it, the message is extracted from the queue and is transformed into an action (a script name together with a list of parameters). This action is forced to be executed inside the Adobe Atmosphere Player. As the result of the action the state of the Visual Interaction Layer is updated by the Atmosphere Community Server, that communicates the state update to all the connected clients.

The Figure also illustrates the types of connections that are made on each side. It is shown that the Causal Connection Server opens two server sockets for incoming communication requests from Federation Monitor and Institution Monitor. Both monitors use the client socket connection type to establish the connection with the Communicator. Each of the agents started by the Agent Launcher communicates with the server socket.
of the corresponding Governor via client socket mechanism. Finally, the Web Interface acts as a client and connects to the server socket of the Communicator component inside the Causal Connection Server.

To better explain the collaboration between different components of the runtime architecture Figure 4.16 presents the sequential view of the architecture.

![Component Interaction: Sequence Diagram](image)

Figure 4.16: Component Interaction: Sequence Diagram

The main architectural components presented here are: Web Interface, Causal Connection Server, AMELI and Governor with the actors: human and autonomous agent.

First it is illustrated how the registration of the human participant is conducted. The human registers through the Web Interface by filling in the form on the web site. As the result of pressing the “Submit” button the web interface requests AMELI to allow the hu-
man to enter the institution. AMELI validates this request and assigns the governor. The Web interface is notified about the successful institutional registration by sending it the address of the newly created governor. This address is then forwarded to the Causal Connection Server, which creates an autonomous agent for communication with the governor and establishes a socket connection with it. Next, the Causal Connection Server sends the correct URL for the Atmosphere Player and the Web interface loads the document behind this URL and displays the Virtual World.

The sequence diagram also specifies the details of the process of the human trying to execute an action in the Virtual World. As the result of the human generating an event (pressing a key on the keyboard or clicking the mouse) the action request is sent to the Atmosphere Player. The Atmosphere Player requests the Causal Connection Server to find the message that corresponds to this action. When the message is found it is forwarded to the corresponding autonomous agent, which posts this message to the corresponding governor. The message is then verified by the AMELI and the response is sent back to the agent as another message. The agent forwards the response message to the Causal Connection Server, which consults the Action/Message table to find the name of the corresponding script from the Visual Interaction Layer. The resulting script is executed by the Atmosphere Player and the action is visualized through the Web Interface.

The last example illustrated by this diagram is the case of the agent directly acting in the institution on human’s behalf. This agent may not even be aware of the existence of the Visual Interface level and may bypass the Causal Connection Server and talk directly to the AMELI. If the agent desires to execute an action in the Visual Interaction Layer it sends the corresponding request to the associated governor. The governor then validates this message with AMELI’s help. The institutional response is then sent to the agent itself and also to the Causal Connection Server. The Causal Connection Server finds the action that corresponds to the response message and then this action is performed.

### 4.4 Summary

In this chapter we introduced the Virtual Institutions methodology. Applying this methodology requires the following seven steps to be completed: eliciting specification requirements, specification of an Electronic Institution, verification of the specification, automatic generation of the corresponding 3D environment, annotation of the Electronic Institution specification with components of the 3D Virtual World, integrating the 3D Virtual World into the institutional infrastructure and enabling implicit training.

Each of the methodological steps is described in detail and the technological facilities that help automating each of the steps are outlined.
The implementation requirements expressed in the previous chapter are used for the development of the technological solution presented here. We illustrated the technologies suggested for automatic generation of Virtual Institutions together with tools facilitating annotation and integration steps of the methodology.

The major technological contribution presented in this chapter is the development of the Causal Connection Server, which is used for deployment and acts as a middle layer between Electronic Institutions and Virtual Worlds. The Causal Connection Server is capable of translating the messages of the Normative Control Layer into actions of the Visual Interaction Layer and vise-versa.

The notion of the Causal Connection Server also makes a significant conceptual contribution. It supports connecting the institutional infrastructure to a number of different visualization platforms simultaneously and can even potentially be used for merging the real world and a Virtual World.