TimeSaver – Virtual Worlds and Active Workflows to Deliver Friendly Public Services

Flavio S. Correa da Silva
LIDET
University of Sao Paulo
Rua do Matao, 1010, Sao Paulo, Brazil
fcs@ime.usp.br

Rosa Alarcon
Department of Computer Science
Catholic University of Chile
Av Vicuna Mackenna 4860, Santiago, Chile
ralarcon@ing.puc.cl

Claudia J. A. de Araujo
LIDET
University of Sao Paulo
Rua do Matao, 1010, Sao Paulo, Brazil
claudiaabrao@uol.com.br

Carla Vairetti
Department of Computer Science
Catholic University of Chile
Av Vicuna Mackenna 4860, Santiago, Chile
cmvairet@cm.cl

Lisney Alberti
Department of Computer Science
Catholic University of Chile
Av Vicuna Mackenna 4860, Santiago, Chile
lmalbert@uc.cl

Jesus Bellido
Department of Computer Science
Catholic University of Chile
Av Vicuna Mackenna 4860, Santiago, Chile
jebellid@uc.cl

February 24, 2010

Abstract

The effective delivery of services in electronic government is a critical feature and also a difficult challenge. Contrasting with commercial systems, electronic government systems must be directed to the broadest possible spectrum of consumers, lest their value as tools to improve and modernise democracies can be hindered. Furthermore, since in the vast majority of cases such systems have to deal with legacy bureacratic organisations, which nevertheless deal with security sensitive information and provide services whose permanent availability can be an issue of national security in many cases, their design tools must contain resources to analyse and improve legacy systems whilst minimising the interference on existing workflows, in order to improve the quality of services without causing disruptions in their availability. In the present article we introduce TimeSaver, which aims at assembling a toolkit for the development of systems to support the electronic government whose development and features have catered for all issues referred to above. In order to exhibit how TimeSaver can be used, we illustrate its use to design a user friendly system to provide cross borders services and information – in our case about visa issuing for citizens to visit foreign countries for specified reasons – thus interfacing different national bureacracies to deal with similar problems[1].

[1]This work has received partial financial support from LACCIR – the Latin American and Caribbean Collaborative ICT Research Institute

1 Introduction

As well documented in several existing articles (see e.g. [8, 6]), there is a global pressing demand for the modernisation and improvement of quality of public administration and the governmental machinery as a whole. As summarized by Okot-Uma [8], this demand amounts to turning governmental systems more *participatory*, *transparent* and *accountable*.

Electronic government refers to the utilisation of information technology, in its broadest sense, to support the modernisation and improvement of quality of public administration and the government as a whole. Most often than not, it refers to the utilisation of computer-based information technology.

Among several resources that can be employed to improve the quality of government services, *interoperability* has been pointed out as an important feature [9]. As pointed out by Pardo and Burke, however,

> *Interoperability, like technology, is not an end but a means to an end. Citizens do not demand interoperability; rather, systems must be interoperable in most cases for governments to deliver what citizens do demand* [9].

Interoperability is a multifaceted concept. Essentially, it is about putting independent resources to communicate and exchange messages and services, and it can be considered in different levels of abstraction, e.g. technical (related, for example, to IT infrastructure and the effective exchange of data packets and web services), semantic (related to the effective exchange of services and data that are *meaningful* to all parties and have their interpretations aligned according to common norms and regulations, in such way that their utilisation is coherent across all parties) and organisational (related to the alignment and orchestration of rules and regulations that govern each party that is willing to interoperate). If we take the notion of *bureaucracy* in its original, Weberian sense [7], interoperability can be a means to solve some of the distortions that deviate bureaucratic organisations from their tenets of rationality and efficiency.

One concrete metric (among others) to assess the quality of electronic government initiatives is the extent to which the services and informations provided/requested by governmental systems are friendly. Friendliness for electronic government services can be a broader concept than for commercial software services, as, by definition, electronic government should be about delivering services to the largest possible spectrum of citizens, irrespective of their cultural or educational background, ethnicity, gender, age etc.

In a rather specific technical sense, this metric is about organisational interoperability, as it correlates with the extent to which different groups of individuals can interact with information systems and, through them, with different organisations that provide public services or are interested in collecting relevant information in order to operate.

A successful system in the capability to interact effectively with the population at large should expose how information is treated internally at public organisations, in order to gain on reliability. This feature can, as a corollary, build opportunities to improve the workflows that are employed in those organisations, especially in what relates to how they interface with each other and with external users.

To take a concrete example, let us consider the problem of issuing special purpose visas for foreigners to enter a country. Even though, based on diplomatic reciprocity rules, the regulations and requirements for a citizen from country A to go to country B for a specific purpose are similar to those required for a citizen from country B to go to country A for the same purpose, internal national rules and bureaucracies can bring about subtle yet significant differences in procedures.

Let us consider, for example, what occurs when a Brazilian citizen wishes to go to Chile as a student. This individual has to go to a Chilean diplomatic authority in Brazil and follow the required procedures to obtain his/her visa. It is likely that the Brazilian citizen may find Chilean bureaucratic rules, presented according to Chilean terminology and requirements, at the Chilean consulate in Brazil. It may also happen that some of these requirements and terminology are hard to follow for the Brazilian individual.

Hence, supporting information systems that presented the necessary information according to the appropriate local terminology, as well as adapted to the local regulations and standards, can be very helpful. A fair goal in the design of such systems would be to ensure high usability levels by communicating well with users, together with high reliability levels by informing about how information is consumed and processed in order to provide the required services.

In this specific case, as in a large portion of the relevant cases for electronic government, the fundamental tool to achieve this goal can be an environment to specify and execute interaction protocols, in such way that users can interact effectively with information systems, which also provide their users with internal information about how they treat information internally.

In the present article we introduce the TimeSaver initiative, which has been designed precisely to achieve this goal. TimeSaver aims at the assembling of an environment to build organisational interoperability protocols, and is based on JamSession [4], which is a general purpose environment to specify and build interaction protocols for multiagent systems and intelligent interactive environments.

Our goal is to be able to specify systems aiming precisely at optimising the organisational interoperability as depicted above. In section 2 we highlight some related work that has inspired our work on JamSession and, more specifically, on its utilisation within the TimeSaver initiative. In section 3 we briefly review the JamSession language, and discuss how it has been used in the TimeSaver initiative. In section 4 we show how TimeSaver could be used to build an information system to support Brazilian citizens wishing to study in Chile. Finally, in section 5 we draw some conclusions and discuss planned future work.

2 Related Projects

The development of JamSession, and its utilisation in the TimeSaver initiative, have been inspired by existing research initiatives which account for specific subsets of the expected features referred to in the previous section.

A conceptual architecture that bears several similarities with our initiative is that of Electronic Institutions [5]. Electronic Institutions are comprised by agents, roles, normative rules that characterise the institutions and scenes in which regulated dialogues occur involving agents. For an agent to participate in a dialogue, it must assume a specific role that is compatible with that dialogue, and in order to assume a specific role the agent must ensure that it has the necessary capabilities required for that role. Once all roles required for a dialogue are fulfilled by agents, the dialogue can start and, if all norms are followed, the dialogue can run to its end, thus performing a structured and controlled sequence of linguistic actions.

In JamSession, the concept of role is replaced by an equivalent notion of location. Agents have to move to specific locations in order to participate in collaborative actions, as some of their capabilities becomes available only in those locations.

Indeed, an interesting exercise to be developed in future work shall be the construction of mappings between Electronic Institutions and JamSession. A further point of connection between Electronic Institutions and JamSession is that both have evolved to build user friendly systems, based on user interfaces that resort to interactive animations as means to bring to users the experience of interacting with services as if interacting with human clerks in a simulated virtual environment [2].

Norm-mediated interactions have been developed in a variety of initiatives. In these initiatives (see e.g. [3, 11]) interaction protocols are grounded by normative rules that specify permissions and prohibitions to which agents must conform. In JamSession, this concept is simulated as conditional rules attached to specified locations.

The Lightweight Coordination Calculus (LCC) [10] is a language to build executable specifications akin to algebraic specification languages such as the Ambient Calculus. It has been proved to have similar expressive power to Electronic Institutions, and it has been used extensively in a variety of applications.

The main difference between the LCC and JamSession is that the former does not account explicitly and directly for the notion of mobility and situatedness. LCC has indeed been extended to account for situatedness explicitly, thus resulting in a language coined Ambient LCC, which however seems not to have used in practice, perhaps because it has turned to be a little unfriendly and difficult to implement. JamSession has been designed specifically to simplify the specification of interactions in this situation.

In a slightly different vein, the Multilayered Multiagent Situated Systems architecture (MMASS) [12] considers mobility as the central notion for situatedness and interactions in multiagent systems. In MMASS we find the notion of locations, which are structured by pathways forming a graph of sites through which agents can move to look for specific resources to accomplish their goals. We have borrowed these concepts for JamSession.

In the TimeSaver initiative, we have employed JamSession as the platform to build concrete systems in which organisational interoperability is emphasised, especially in what regards user friendliness and publication of workflows, focusing primarily on applications related to electronic government.

3 The JamSession Language

A fundamental notion in JamSession is the concept of *location*. Intuitively, a JamSession system is a collection of services that are offered to users. These services are organised in locations, in such way that services are blocked or unblocked for utilisation, depending on the location of agents. In JamSession parlance, agents are triggers that unlock specified services by moving to the appropriate locations. Whenever we need a specific service, we must make sure that the agent that has the capability of furnishing that service in a specific location has actually moved to that location.

Formally, we have a directed graph to specify locations and their connections. The nodes of the graph are the locations, and the arrows characterise the admissible transitions that agents can perform to move about locations.

Agents are entities that inhabit locations. An agent stays in a location until it receives an order to move to a different location.

In JamSession, we employ a four valued lattice of truth values known as *Belnap four* [1]. In this lattice, we have the truth values `undefined`, `true`, `false` and `inconsistent`. The truth value `undefined` stands for a statement whose truth value cannot be determined. The truth values `true` and `false` correspond to the usual (classical) notions of truth and falsity. The truth value *inconsistent* stands for a statement whose truth value has been determined as both true and false. The order relations in the Belnap four lattice are as follows, in which $\leq$ stands for the usual ordering relation:

- `undefined` $\leq$ `true`.
- `undefined` $\leq$ `false`.
- `true` $\leq$ `inconsistent`.
- `false` $\leq$ `inconsistent`.

This order is used in JamSession to monitor the execution of interaction protocols and to ensure that, after agents interact and migrate along locations – and hence provide services to users – no exceptional state (e.g. in which an `inconsistent` truth value is generated) has been reached.

JamSession can be envisaged as an orchestrator of external resources. The constructs in this language have been designed to provide the appropriate means to coordinate and to regulate the triggering of available resources. The conceptual tool to perform this coordination and control of resources is the movement of agents through locations.

External resources are named *predicates*. Each predicate is associated to a pair $[Agent, Location]$. Predicates also have parameters, which are formed as classical first-order terms. A predicate can be triggered, i.e. there can be an attempt to evaluate it. Most typically, predicates are used to consume external resources and call external services. The predicate parameters are passed to these external resources and services, and if the resources and services are successful they are expected to send back appropriate instantiations of the parameters.

When a predicate is triggered, three situations can occur:

1. The predicate is blocked. This can occur because the agent *Agent* is not present in the location *Location*, or because the attempt to trigger the predicate is ill specified, e.g. the requested predicate is not actually associated to the given pair $[Agent, Location]$, or the number of parameters that has been passed to the predicate does not correspond to its definition. In this case, no services or resources are consumed and the predicate is evaluated as `undefined`.

2. The predicate is triggered accordingly, and the external corresponding services or resources run successfully. In this case, the parameters are instantiated and the predicate is evaluated as `true`. In this case, there is also consumption of resources and/or services.

3. The predicate is triggered accordingly, but the external corresponding services or resources fail. In this case, the predicate is evaluated to `false`. Notice that, in this case, since an attempt to execute the corresponding external services and/or consume external resources has actually occurred, the predicate parameters may be instantiated to values corresponding to the partial execution of services and utilisation of external resources. In other words, when a predicate fails, there can occur side effects as a consequence of its evaluation.

The explicit resource we have in JamSession for the orcherstration of resources is the construction of *protocols*. A protocol is a structured chain of events, and it specifies who should come after who. Events can be of four types:

1. Orders for agents to move.

2. Triggering of predicates.

3. Triggering of auxiliary protocols.

4. Combination of previous events by means of connectives.

Protocols are linked to locations. A request to trigger a protocol can result in the following alternative situations:

1. The requested protocol is not actually defined for the specified location. In this case, the obtained truth value is `undefined`.

2. The requested protocol is defined for the specified location. In this case, the specification of the protocol is retrieved and evaluated, based on the algebraic rules that govern the behaviour of the connectives that are used in the specification of the protocol. The result of the evaluation determines the truth value that shall be assigned to the protocol.

For the specification of protocols, we employ four binary connectives, whose behaviour is as follows:

1. Classical conjunction ($\wedge$): classical conjunction builds the greatest lower bound of the truth values of its operands: given two (structures of) events α and β, and assuming that we can assess the truth values associated to each of them, then the truth value of $\alpha \wedge \beta$ shall be the greatest lower bound for the truth values of α and β.

2. Classical disjunction ($\vee$): classical disjunction builds the least upper bound of the truth values of its operands: given two (structures of) events α and β, and assuming that we can assess the truth values associated to each of them, then the truth value of $\alpha \vee \beta$ shall be the least upper bound for the truth values of α and β.

3. Sequential conjunction ($\sqcap$): sequential conjunction evaluates left to right, and employs different rules depending on the event to the right. The basic idea is to regulate the evaluation of chains of events, by taking into account the success or failure of orders for agents to move:

   (a) If the event to the right is an order to move, then the truth value of the event to the left is expected to be `true`. In other words, an order to move is expected to be triggered only if its antecedents have been successful. Intuitively, this resource has been implemented in Jam Session as a tool to regulate the global behaviour and security of systems. When an agent moves to a new location, previously available services and resources are locked up, and new services and resources are released. The sequential conjunction provides the means, in Jam Session, to implement conditional rules for the switching of available services and resources.

   (b) If the event to the right is any other event different from an order to move, then the truth evaluation follows the rules of classical conjunction.

4. Sequential disjunction($\sqcup$): sequential disjunction evaluates left to right, and employs different rules depending on the event to the left. The basic idea here is to attempt different alternative solutions only if necessary. When a satisfactory truth value is achieved, the evaluation is resumed.

The specification of a protocol is a chain of events linked by connectives, thus forming nested pairs of the form $(\alpha_1\ conn_1\ \alpha_2)\ conn_2\ \alpha_3)...)\ conn_u\ \alpha_{u+1})$, in which α_i are events and $conn_j$ are connectives.

Sequential conjunction has priority over classical conjunction; both conjunctions have priority over sequential disjunction; all these three connectives have priority over classical disjunction.

Protocols can be threaded. This means that more than one single protocol may be running at any time. It shall be left to the programmer to ensure that the resulting concurrent groups of threaded protocols generate behaviours that are in accordance with the existing systems requirements for each application.

4 TimeSaver at Work: Student Visas to Chile

In this section we put JamSession to work, in the context of the TimeSaver initiative. Our goal is to show how JamSession can be used to orchestrate services in order to compose complex systems.

It is well known that the requests for special purpose visas for individuals to visit foreign countries can be quite complicated. Significant portions of the workflow to obtain a visa could be automated and presented to citizens through the web, thus creating opportunities to simplify the corresponding procedures, as well as to make the system more user friendly and cost effective. Furthermore, as mentioned in previous sections, the web presentation of services and resources related to electronic government should always be highly user friendly and directed to the broadest possible spectrum of individuals.

We have implemented a simple prototype to inform Brazilian citizens about the required procedures to obtain student visas to study in Chile. In this prototype, we have five locations and one agent. Depending on what location the agent is, different predicates are available, which trigger external resources that provide speech-based information about the procedures to follow, as well as capture information from the user.

The system can be envisaged as mimicking the actual physical interactions a user could go through while at the Chilean consulate in Brazil. Each location can be seen as the simulation of a desk, a clerk or a specific acitivity developed by a clerk at the consulate, and when the agent moves to a different location the user could have a sense of having moved to a different step in the process to obtain the visa.

Our prototype can only provide useful information to citizens. It would actually be simple to extend it to perform services that could support the provision of visas to citizens. All that would be necessary would be to replace certain services – which at the moment only provide relevant information to citizens – by services which could effectively perform parts of the required workflow in order to simplify the issuing of visas.

A prototype for JamSession has been implemented in PROLOG. We have employed this prototype to build our information system. In this system, the user is advised about what to do by an animated cartoon character. The messages that are spoken by the animated character are pre-recorded messages[2].

[2] In order to create the spoken messages from written text, we have employed a commercial product called CrazyTalk©.

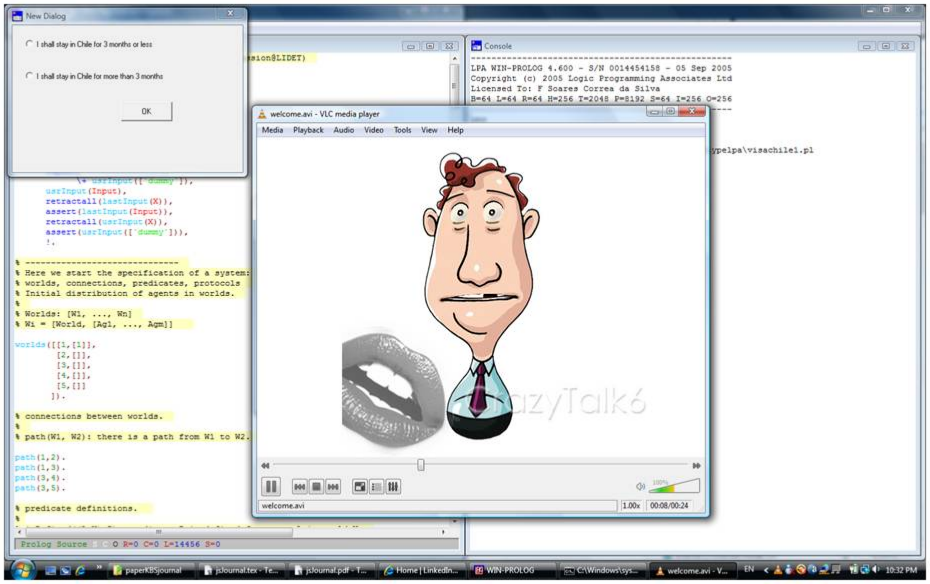

Figure 1: Screenshot of the demo system to inform about how to obtain student visas to Chile

A screenshot of the demo, in which the animated cartoon appears providing information to the user, is presented in Figure 1.

The five locations are organised as a tree, as depicted in Figure 2.

Location 1 is the entrance door to the system (and to the virtual environment that provides information to the user). When a user starts to use the system, the agent is placed in location 1, which also hosts a predicate that can trigger an external resource which salutes the user, provides information about how the system works and asks for some basic information.

Depending on what the user replies, the agent is directed to either location 2 or location 3. Once in one of these new locations, the agent can trigger the predicates that are hosted by that location, which shall provide the user with additional information and ask for additional answers. This way, as the agent hops from location to location, different system states are characterised,

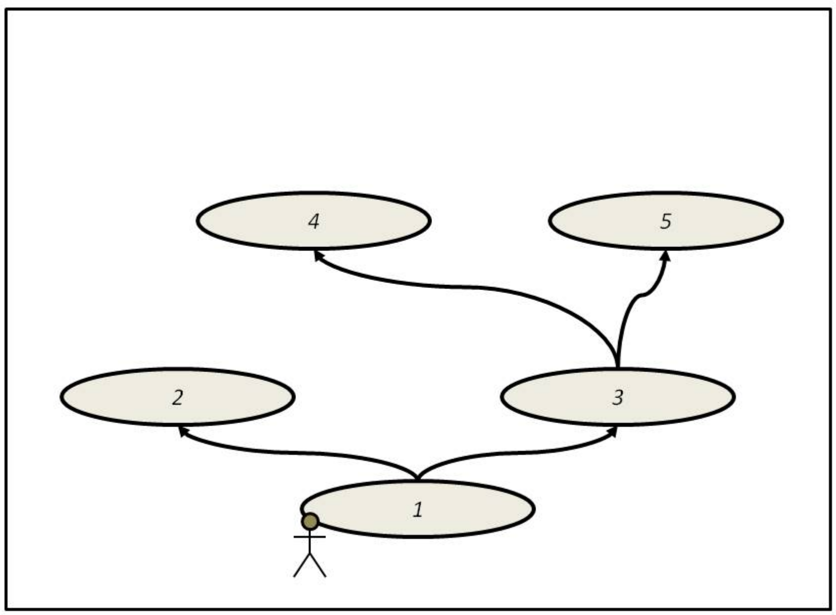

Figure 2: The structure of locations in the demo

different resources are released, and different behaviours of the system can be perceived by the user.

Evidently, for a simple system as the one depicted in this demo, much simpler solutions could be developed. We have used this example only as an illustration of how JamSession works and can be used in a context such as the TimeSaver initiative. The advantages of JamSession over more traditional development tools becomes more evident as we scale up the complexity of interactions in the intelligent interactive environment that is being developed.

5 Conclusion and Future Work

In the present work we have identified a special class of problems which are particularly relevant for electronic government initiatives, in which organisational interoperability is a relevant feature, especially in what regards the construction of friendly interaction for a broad spectrum of users. We have also reviewed the JamSession language, which can provide the means to design and implement system solutions based on the notion of intelligent interactive environments, and illustrated how JamSession could be used for such class of problems.

In future work we shall develop further the analysis of JamSession as a tool to develop systems targetting at organisational interoperability, through the development of more sophisticated prototypes of systems to support electronic government initiatives.

References

[1] Belnap, N.D.: A useful four-valued logic. Modern Uses of Multiple-Valued Logic. (Dunn, J. M., Epstein, G. - eds.). Dordrecht: Reidel (1977) 8-37

[2] Bogdanovich, A., Esteva, M., Simoff, S., Sierra, C., Berger, H.: A methodology for 3D electronic institutions. 6th International Joint Conference on Autonomous Agents and Multiagent Systems. USA (2007) 358-360

[3] Correa da Silva, F. S, Vasconcelos, W.: Rule schemata for game artificial intelligence. International Joint Conference IBERAMIA/SBIA. Springer-Verlag Lecture Notes in Artificial Intelligence 4140 (2006) 451–461

[4] Correa da Silva, F. S.: JamSession - Knowledge Based Interaction Protocols for Intelligent Interactive Environments. Technical Report, Department of Computer Science, University of Sao Paulo. Brazil (2010)

[5] Esteva, M., Rodriguez-Aguilar, J. A., Sierra, C., Garcia, P., Arcos, J. L.: On the formal specification of electronic institutions. Agent-

mediated electronic commerce. Dignum, F., Sierra, C. (eds.). Springer-Verlag Lecture Notes in Artificial Intelligence 1991 (2001) 126–147

[6] Finger, M., Pecoud, G.: From e-Government to e-Governance? Towards a model of e-Governance. Electronic Journal of Electronig Government, v. 1(1) (2003) 1-10

[7] Jain, A.: Using the lens of Max Weber's Theory of Bureaucracy to examine E-Government Research. 37th Hawaii International Conference on System Sciences. USA (2004)

[8] Okot-Uma, R. W.: Electronic Governance: Re-inventing Good Governance. Online publication. *http://www.electronicgov.net/pubs/index.shtml.*

[9] Pardo, T. A., Burke, G. B.: IT Governance Capability: Laying the foundation for government interoperability. Center for Technology in Government Technical Report (2009)

[10] Robertson, D.: Declarative Agent Languages and Technologies. Leite, J. A., Omicini, A., Torroni, P., Yolum, P. (eds.). (2004) 236–249

[11] Vasconcelos, W., Kollingbaum, M. J., Norman, T. J.: Normative conflict-resolution in multi-agent systems. Autonomous Agents and Multiagent Systems, v. 19(2) (2009)

[12] Vizzari, G.: Dynamic interaction spaces and situated multi-agent systems: from a multi-layered model to a distributed architecture. PhD thesis. University of Milano-Bicocca. Italy (2004)